

Privacy Preserving via HE

Xianhui Lu SCRiPTS NTU

Outline

- What is Homomorphic Encryption
- Privacy Preserving via Homomorphic Encryption
 - Data Aggregation
 - Private Information Retrieval (PIR)
 - Private Set Intersection (PSI)
 - Secure Neural Networks

What is **H**omomorphic **E**ncryption?

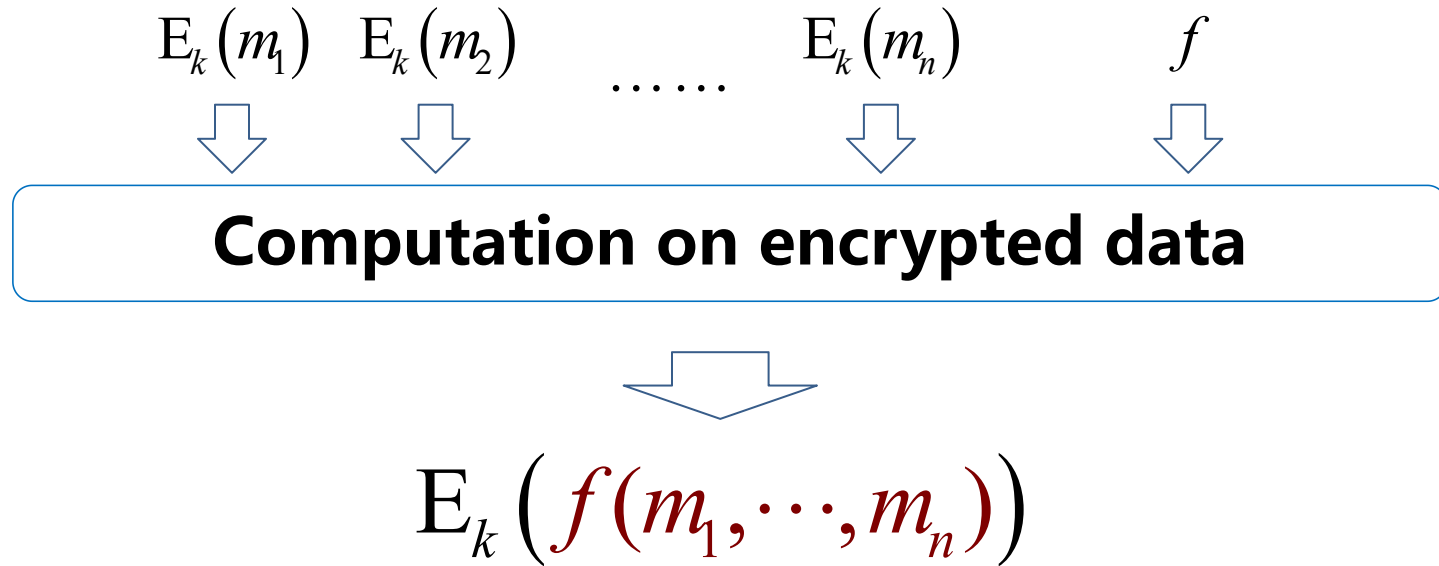
Wiki: **E**ncryption is the process of **encoding** a message in such a way that **only authorized parties can access it** and those who are not authorized cannot.

Welcome to NTU  Zhofrph wr QWX

Wiki: **H**omomorphism is a **structure-preserving map** between two algebraic structures of the same type. It comes from the ancient Greek language: ὁμός (homos) meaning "same" and μορφή (morphe) meaning "form" or "shape".

$$f (x \cdot y) = f (x) \cdot f (y)$$

What is **H**omomorphic **E**ncryption?

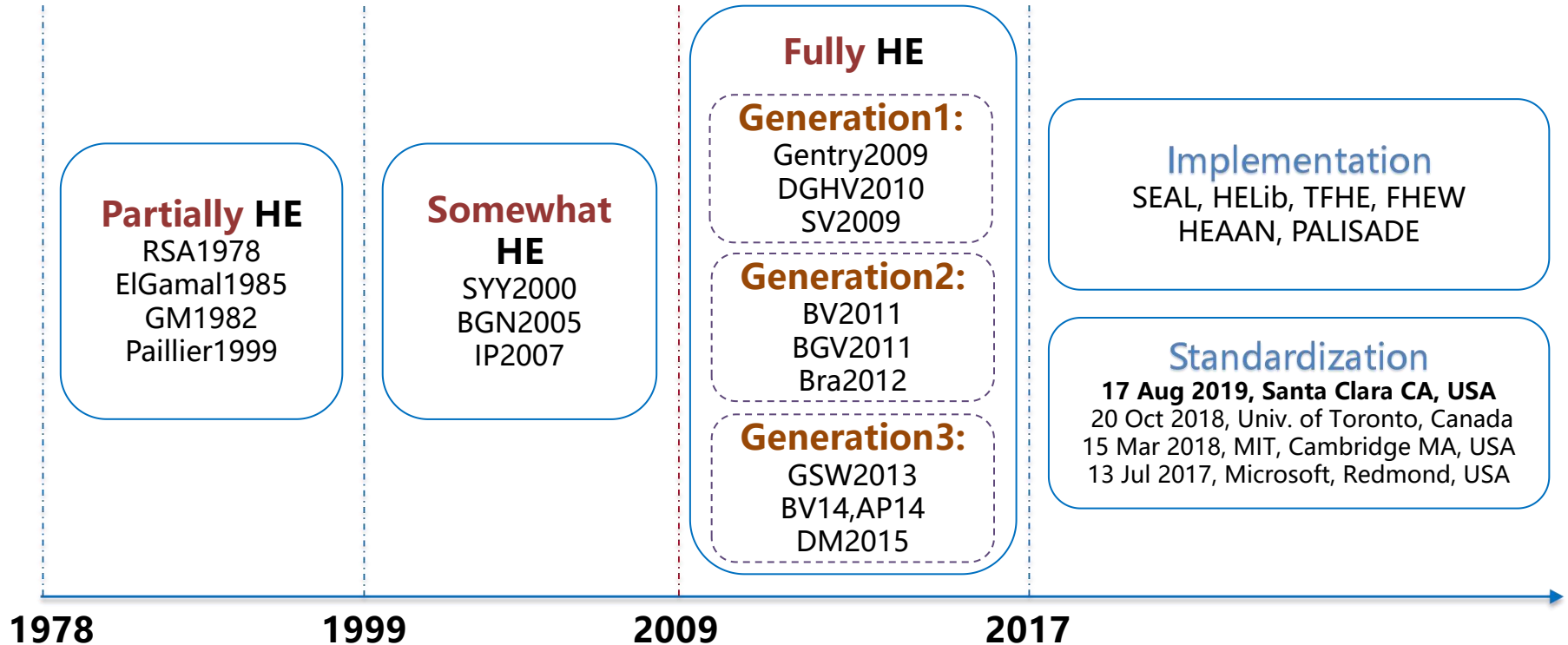


A special kind of encryption that support **Ciphertext Computation**

Types Homomorphic Encryption

Types	Function	Applications	Technical Challenge
Partial HE	Single Homomorphic Operation, Add or Mul.	Data aggregation in IoT Construction of MPC	Easy to achieve, Born with PKC algebraic structure
Somewhat HE	Limited numbers of times Homomorphic Operation, Add and Mul.	Protection of: linear regression simple neural network.	Difficult to achieve, need special structure: pairing, lattice
Fully HE	Unlimited numbers of times Homomorphic Operation, Add and Mul.	Protection of: deep neural network, approximate password,	30 years Open problem Holy Grail of Cryptography First construction in 2009

History of Homomorphic Encryption



Standardization

ISO/IEC 18033-6:2019 Preview

IT Security techniques -- Encryption algorithms -- Part 6: Homomorphic encryption

This document specifies the following mechanisms for homomorphic encryption.

- Exponential ElGamal encryption;
- Paillier encryption.

For each mechanism, this document specifies the process for:

- generating parameters and the keys of the involved entities;
- encrypting data;
- decrypting encrypted data; and
- homomorphically operating on encrypted data.

Annex A defines the object identifiers assigned to the mechanisms specified in this document. Annex B provides numerical examples.

Buy this standard

Format	Language
☒ PDF + ePub	English ▼
☐ Paper	English ▼

CHF 88

General information

Status :  Published
Publication date : 2019-05

Edition : 1
Number of pages : 17

Homomorphic Encryption Standardization

An Open Industry / Government / Academic Consortium to Advance Secure Computation

Standards Meetings

Please join us and participate in our regular standards meetings. Our next meeting is tentatively scheduled for August 17 at Santa Clara CA, after Usenix Sec and before Crypto.



- [August 17 2019, Santa Clara CA, USA](#)
- October 20 2018, Univ. of Toronto, Toronto, Canada
- [March 15-16 2018, MIT, Cambridge MA, USA](#)
- [July 13-14 2017, Microsoft Research, Redmond WA, USA](#)



Open Source Libs

Libs	Function	Owner/Contributor
SEAL	SIMD finite field vectors, fixed point numbers BFV, CKKS (no bootstrapping)	Microsoft (Kristin Lauter, Kim Laine, and Sreekanth Kannepalli)
HElib	SIMD finite field vectors, fixed point numbers BGV, CKKS	IBM (Shai Halevi, Victor Shoup)
HEAAN	SIMD fixed point numbers CKKS	Andrey Kim, Kyoohyung, Han
TFHE	Boolean logic, Comparison GSW13-AP14-DM15-TFHE	S. Carpov, I. Chillotti, N. Gama, M. Georgieva, M. Izabachene
FHEW	Boolean logic, Comparison GSW13-AP14-DM15	Léo Ducas, Daniele Micciancio
PALISADE	Library for lattice cryptography BFV, BGV, LTV, CKKS Prox Re-encryption IB-HE, GPV Signature	NJIT + Partners and Collaborators: MIT, UCSD, WPI, NUS, Sabanci U Raytheon (BBN), IBM Research, Lucent, Vencore Labs, Galois, Two Six Labs

Performance of HE

Factors		Status
Computation Mode & Performance	Integer Arithmetic (BGV&BFV)	1: Fast SIMD computations over integer vectors 2: Fast scalar multiplication 3: Slow bootstrapping, often leveled design (without bootstrapping)
	Approximate Arithmetic (CKKS)	1: Efficient SIMD computations over fixed-point number vectors 2: Fast polynomial approximation 3: Relatively fast division and Discrete Fourier Transform 4: Slow bootstrapping, often leveled design (without bootstrapping)
	Boolean Circuit (GSW & TFHE)	1: Supports arbitrary Boolean circuits (NAND) 2: Fast bootstrapping (13ms) 3: Fast number comparison

Performance: Integer Add & Mul

Candidates	Add (ms)	Mul(ms)
Exponential ElGamal	0.001	NA
Pailliar	0.008	NA
Int16AHE	0.0002	NA
SEAL (BFV)	0.006	1.186
HELib (BGV)	5.949(24)	210.843(24)
TFHE	1040	19968

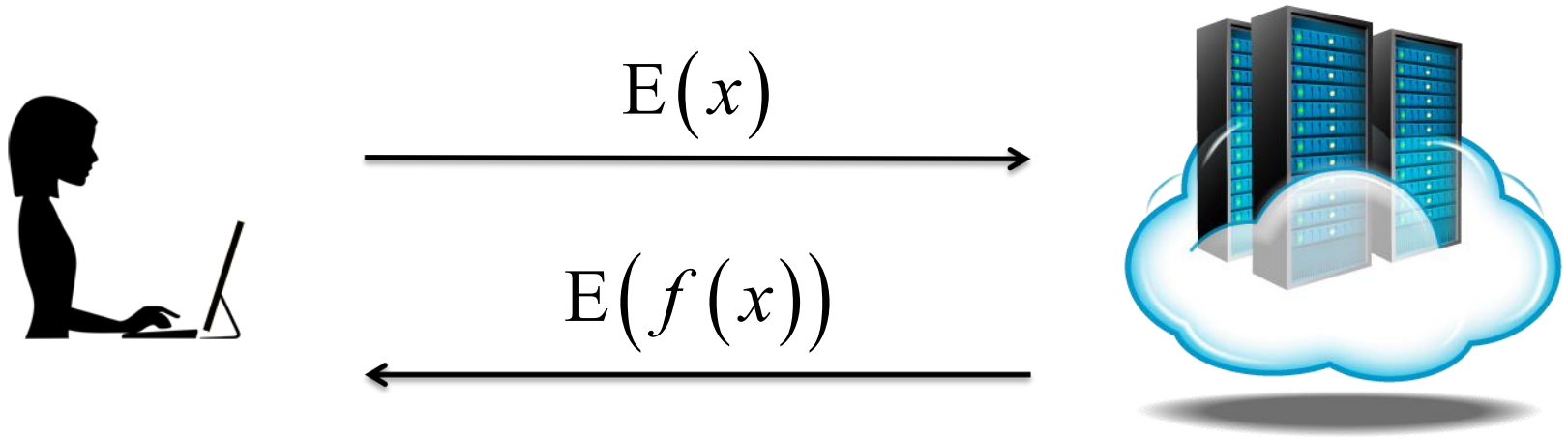
Versatile TFHE: Arithmetic Operation

Operations	Number of Gates	Time(s)
Add_Int32	$32*5=160$	2
Mul_Int32	$32*32*6=6144$	80
Div_Int32	8406	109
Add_Float32	2405	31
Sub_Float32	2325	30
Mul_Float32	5391	70
Div_Float32	7272	95
Sqrt_Float32	3896	51
Sin_Float32	6214	81
Exp_Float32	13364	174

Note: Number of gates form <https://github.com/encryptogroup/ABY/tree/public/bin/circ>

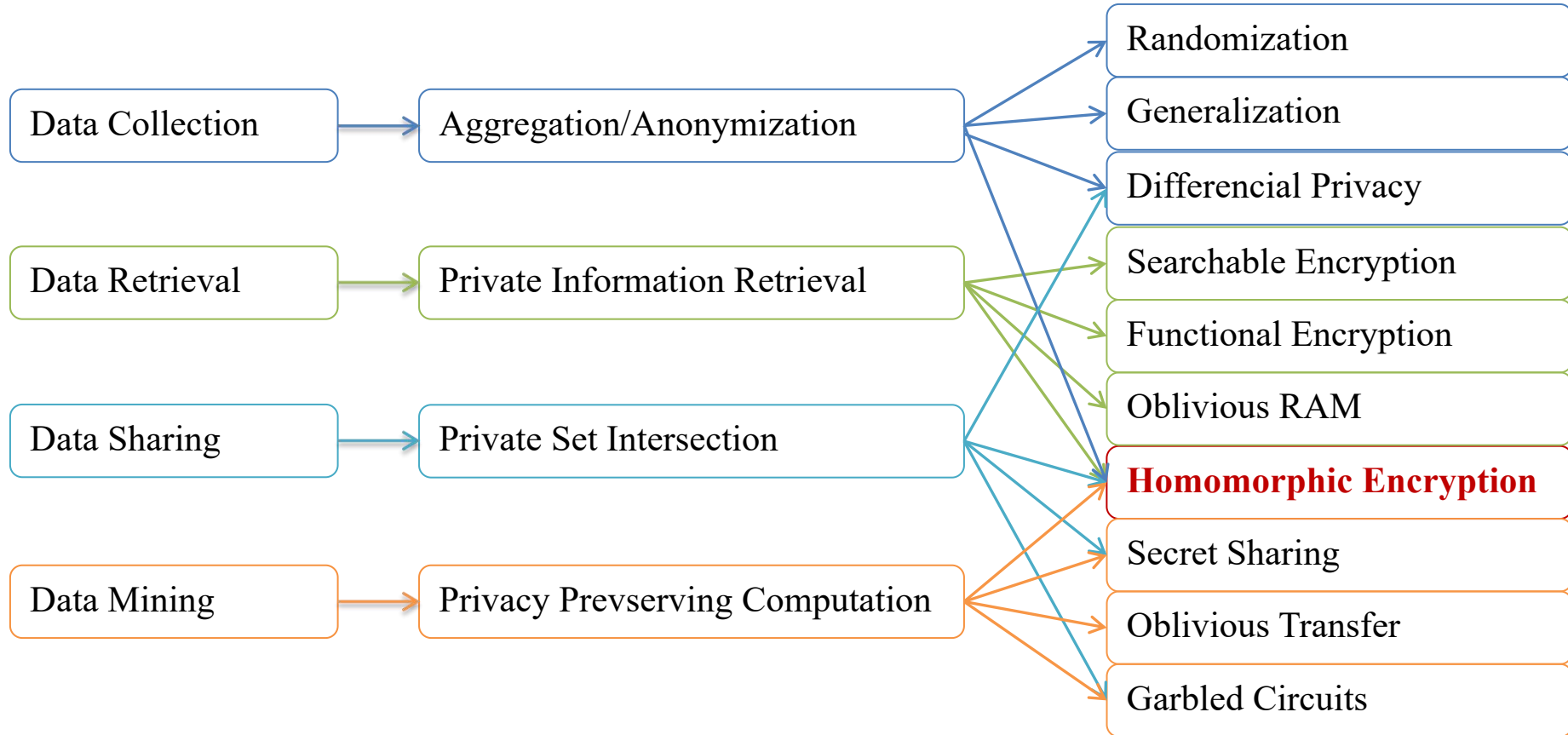


Privacy Preserving via HE

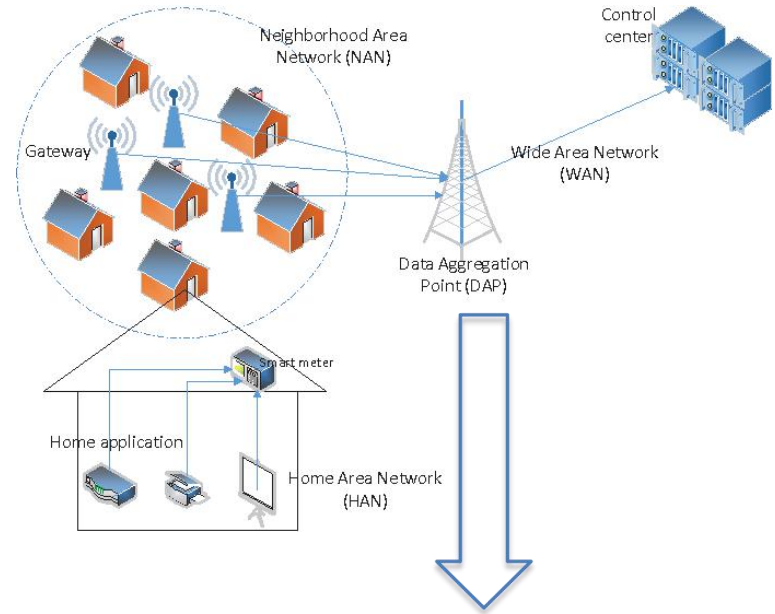
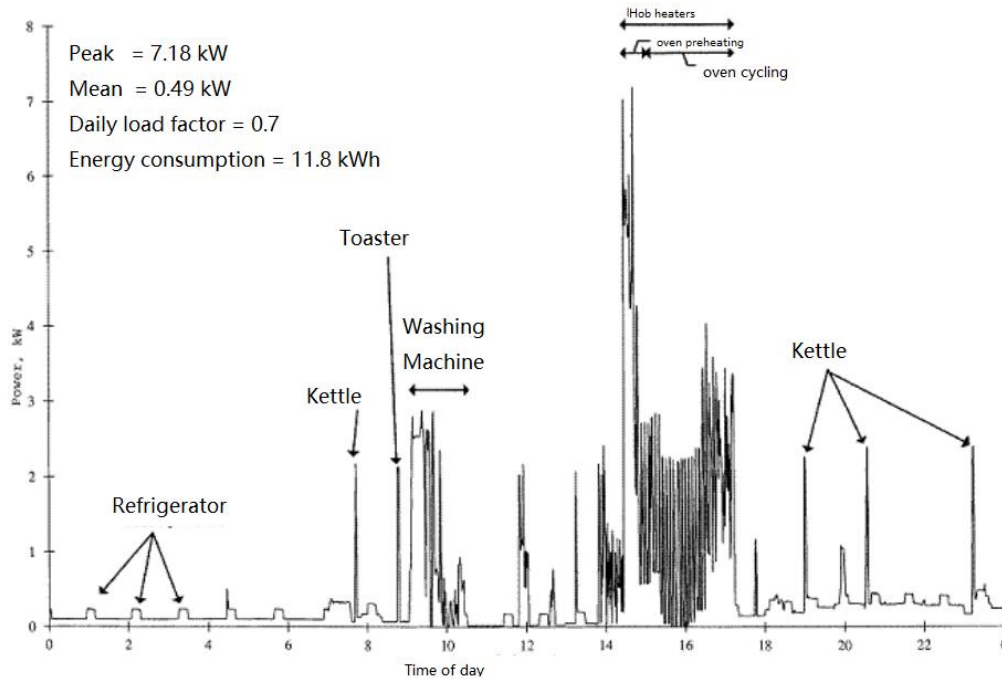


Generic solution for any application.

Privacy Preserving of Data



Privacy-preserving Data Collection

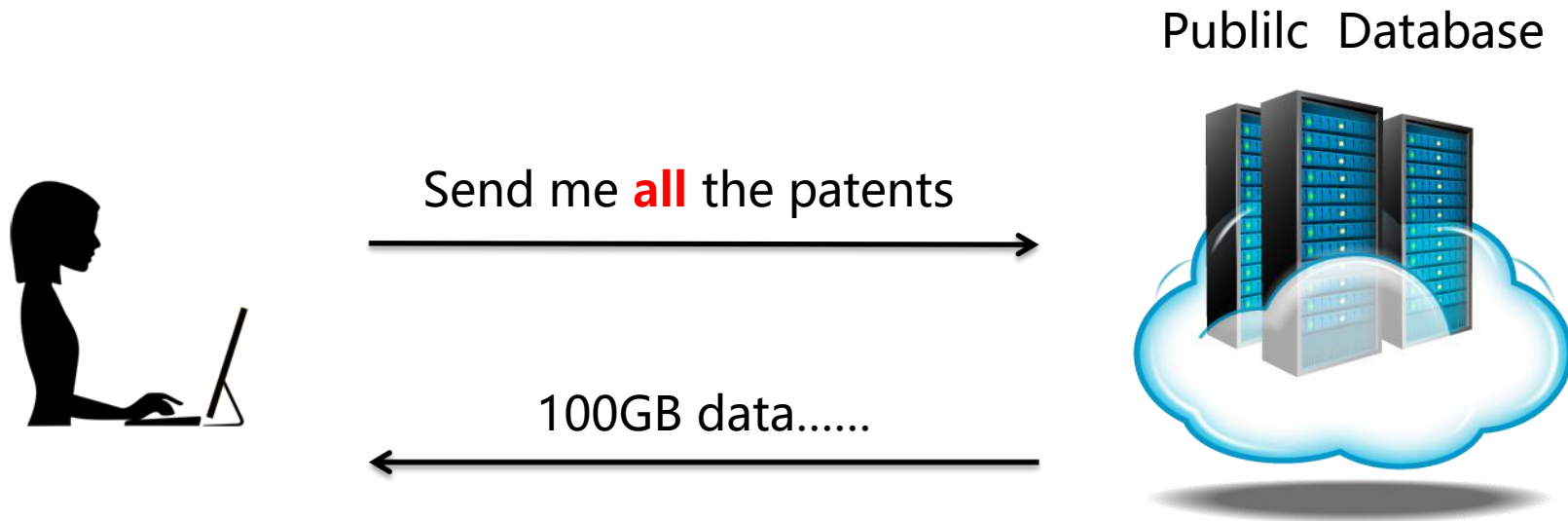


$$E_{pk}(3) + E_{pk}(7) + \dots + E_{pk}(2)$$

Aggregation based on AHE

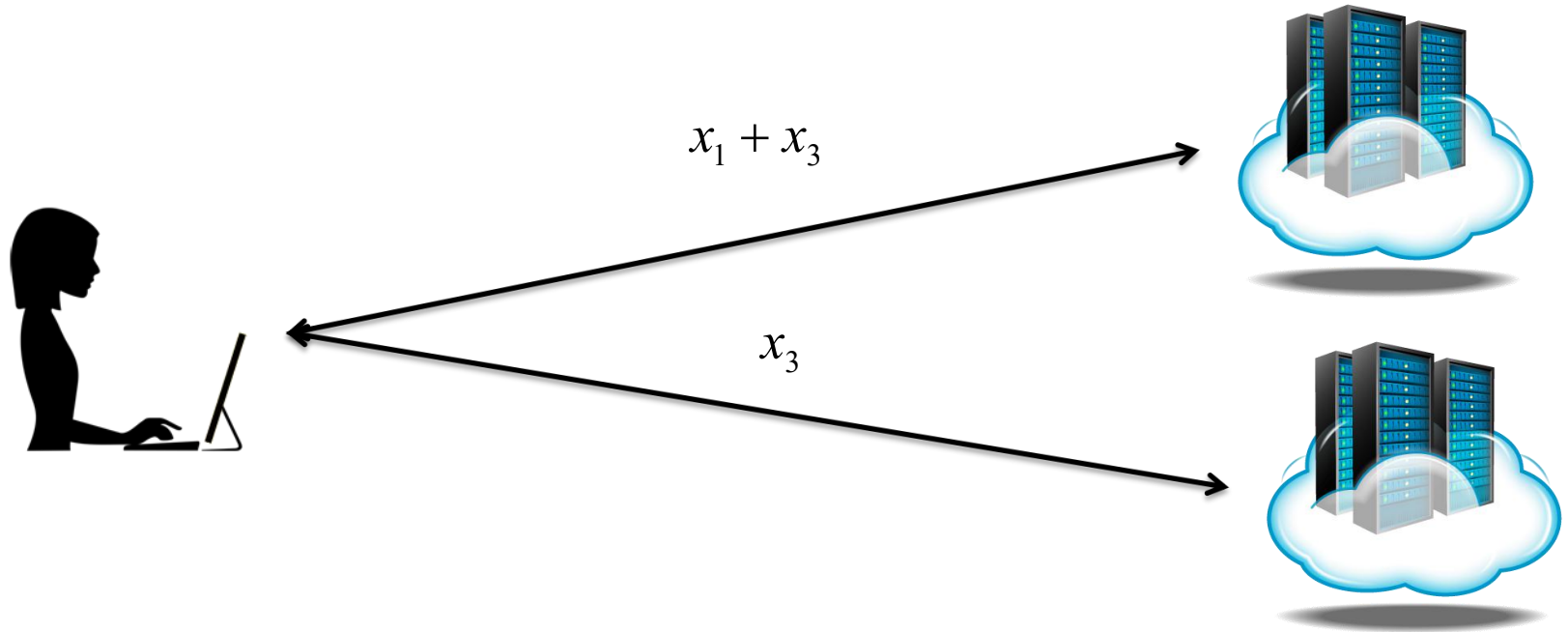
<https://csrc.nist.gov/publications/detail/nistir/7628/rev-1/final>

Privacy-preserving Data Retrieval



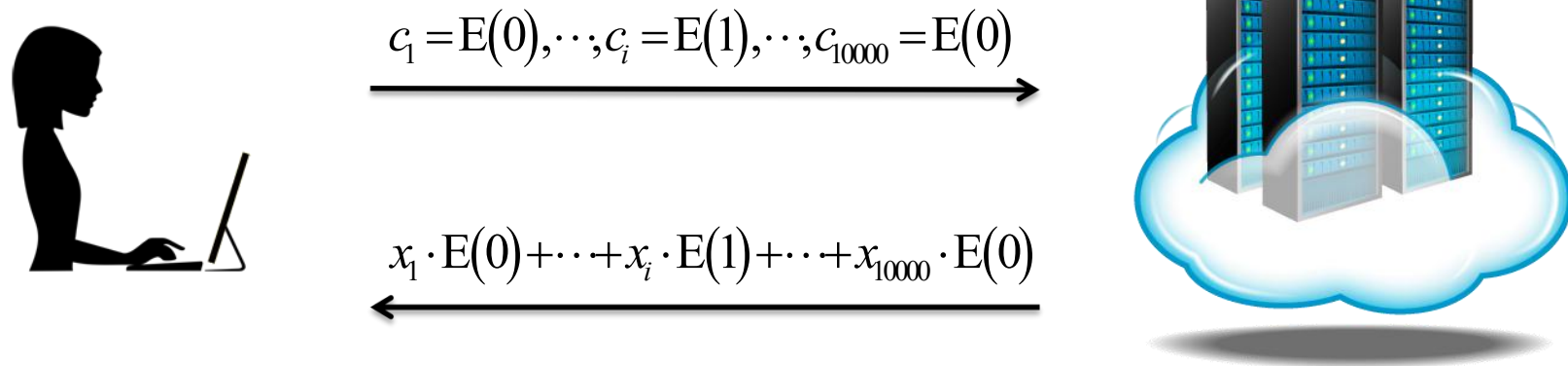
Trivial solution: just send the **whole** database

Privacy-preserving Data Retrieval



Multiple **non-colluding** servers

Privacy-preserving Data Retrieval



Add homomorphic encryption based solution

Privacy-preserving Data Retrieval



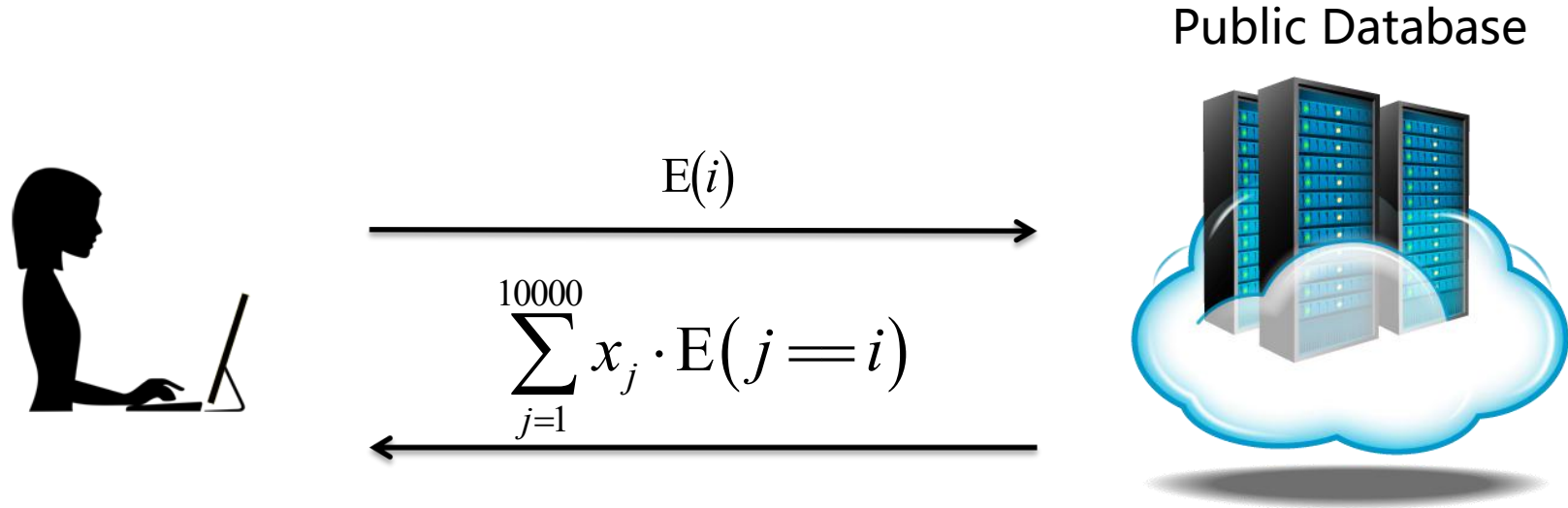
$$\begin{aligned} & \xrightarrow{c_1 = E(0), \dots; c_i = E(1), \dots; c_{100} = E(0)} \\ & \sum_{j=1}^{100} x_j \cdot c_j, \dots, \sum_{j=9901}^{10000} x_j \cdot c_j \\ & \xleftarrow{\hspace{10em}} \end{aligned}$$

Public Database



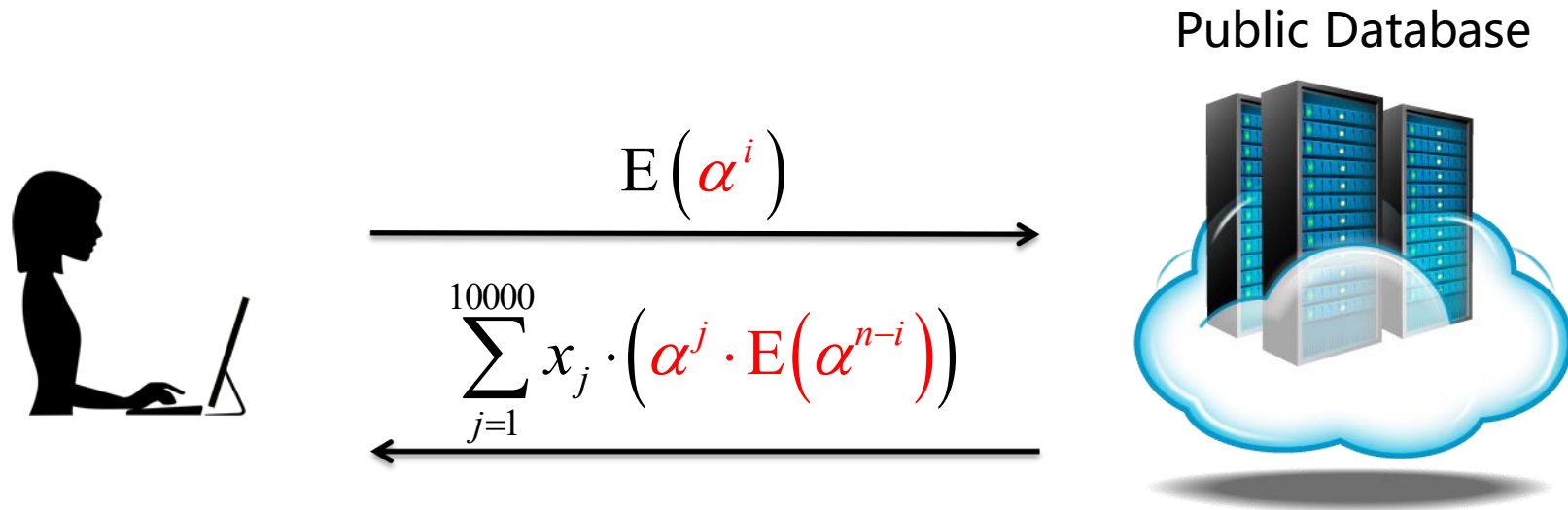
Optimized add homomorphic encryption based solution

Privacy-preserving Data Retrieval



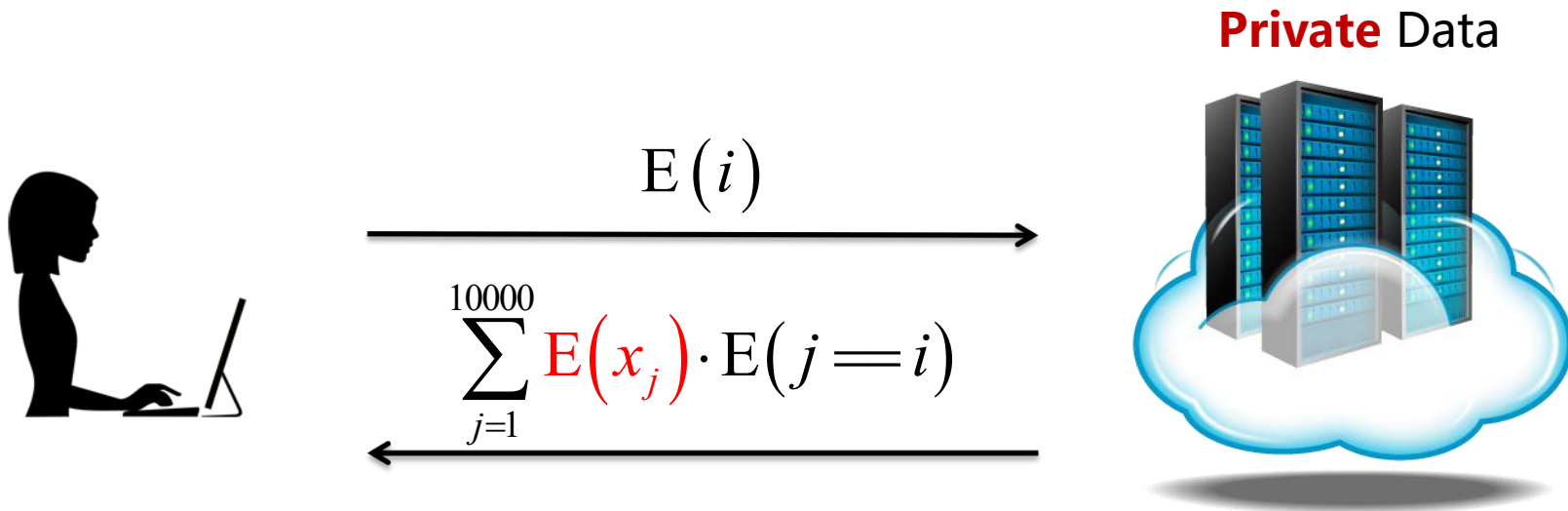
Fully homomorphic encryption based solution

Privacy-preserving Data Retrieval



Polynomial encoding + add homomorphic encryption based solution

Privacy-preserving Data Retrieval

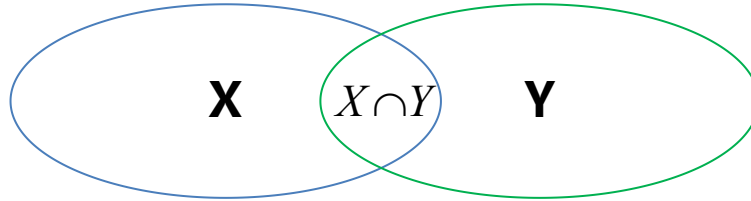


Important building block in **onion ORAM**

Privacy-preserving Data Sharing



Alice



Bob

Private Set Intersection

Privacy-preserving Data Sharing



Alice

$H(x)$



$H(y)$



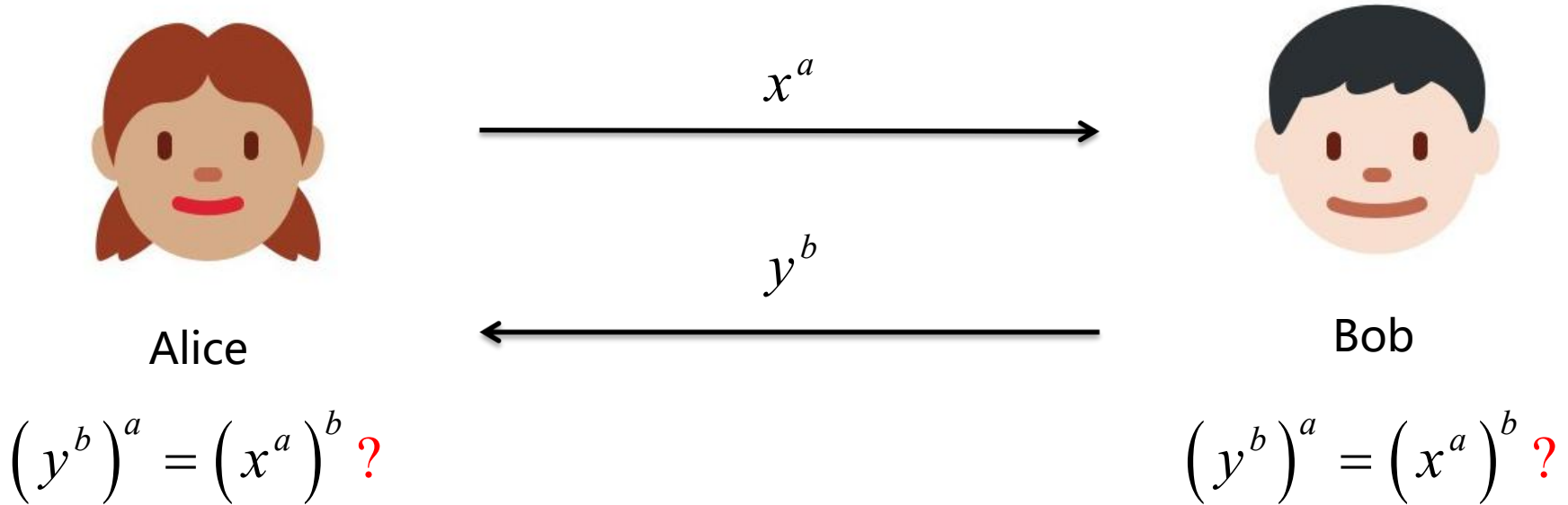
Bob

$$H(x) = H(y)?$$

$$H(x) = H(y)?$$

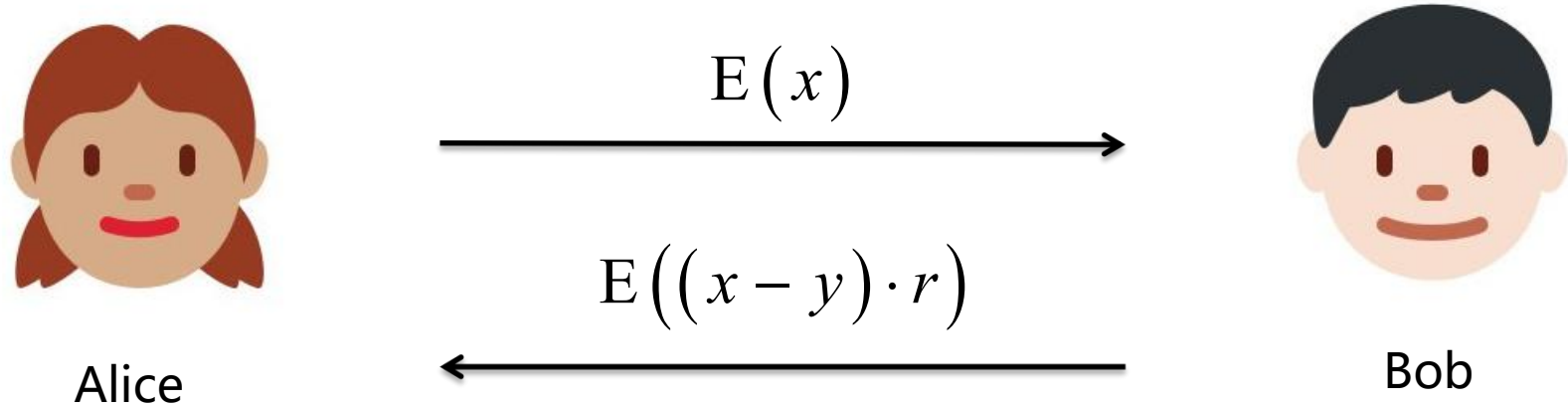
Trivial solution based on hash function

Privacy-preserving Data Sharing



Diffie-Hellman problem based solution

Privacy-preserving Data Sharing

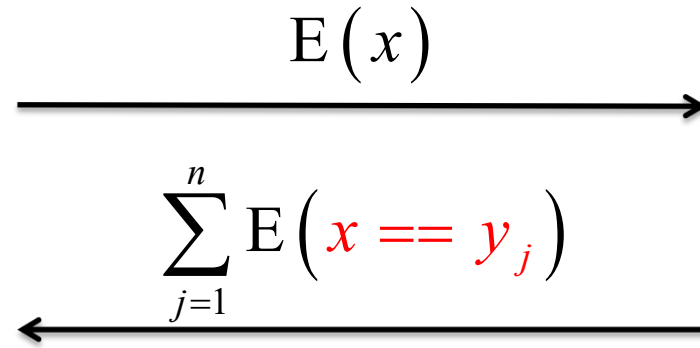


Homomorphic equality test

Privacy-preserving Data Sharing



Alice



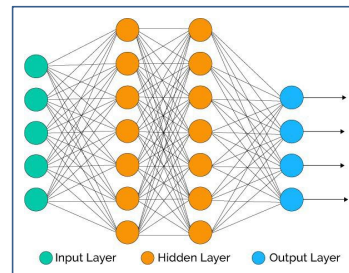
Bob

Homomorphic equality test

Privacy-preserving Data Mining

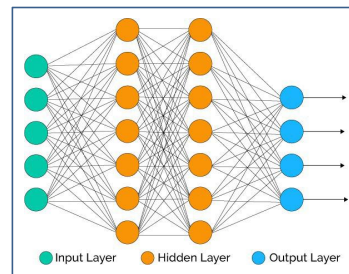
Training:

**Training
Data**



Prediction:

User Data



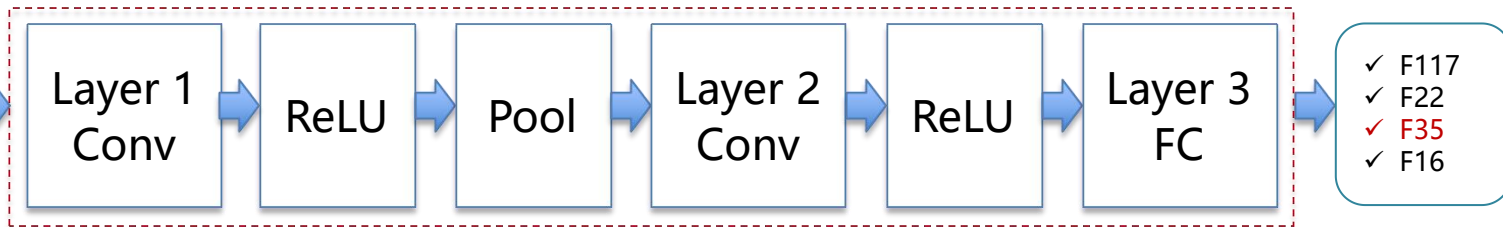
Prediction

Computation Modes of Neural Networks

Input Image

Convolution Neural Networks

Output Classes



$\mathbf{X}$

$$\mathbf{o}_1 = \mathbf{W}_1 \mathbf{x}$$

$$\mathbf{o}_2 = \begin{cases} \mathbf{o}_1 : \mathbf{o}_1 \geq 0 \\ 0 : \text{else} \end{cases}$$

$$\mathbf{o}_3 = \text{mean}(\mathbf{o}_2)$$

$$\mathbf{o}_4 = \mathbf{W}_2 \mathbf{o}_3$$

$$\mathbf{o}_5 = \begin{cases} \mathbf{o}_4 : \mathbf{o}_4 \geq 0 \\ 0 : \text{else} \end{cases}$$

$$\mathbf{y} = \mathbf{W}_3 \mathbf{o}_5$$

$\mathbf{y}$

Input **Vector**

Matrix Mul

Compare

Mean

Matrix Mul

Compare

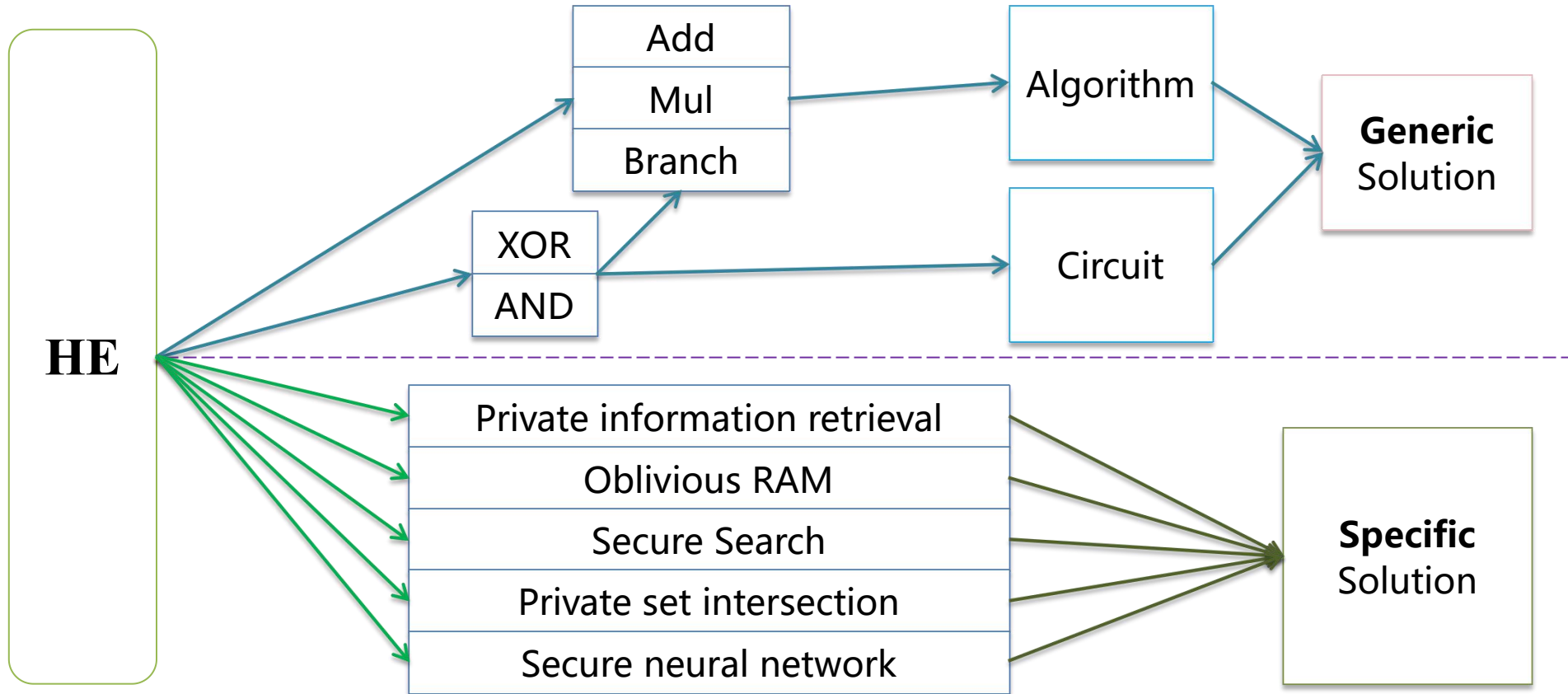
Matrix Mul

Output **Value**

Computation Cost Estimation

Operations	TFHE Based Solution	AHE Based Solution
Matrix Mul 1024*1024	$2^{20} \times 16$ times Add_Int32 about $2^{20} \times 16 \times 2$ seconds	$2^{20} \times 16$ times Add_Int32 about 1.6 seconds
Sigmoid	Exp_Float32+Div_Float32 95+174=269 seconds	
Approximate Sigmoid	2*Mul_Float32 2*70=140 seconds	
ReLU	31 times AND Gate 31*13=403 microseconds	?
Mean (16 items)	16 times Add_Float32 16*31=496 seconds	16 times Add_Int32 about 1.6 microseconds

Conclusion



Thanks

