Private Set Intersection from FHE

VISA Research

Peter Rindal

Microsoft Research

Hao Chen
Kim Laine

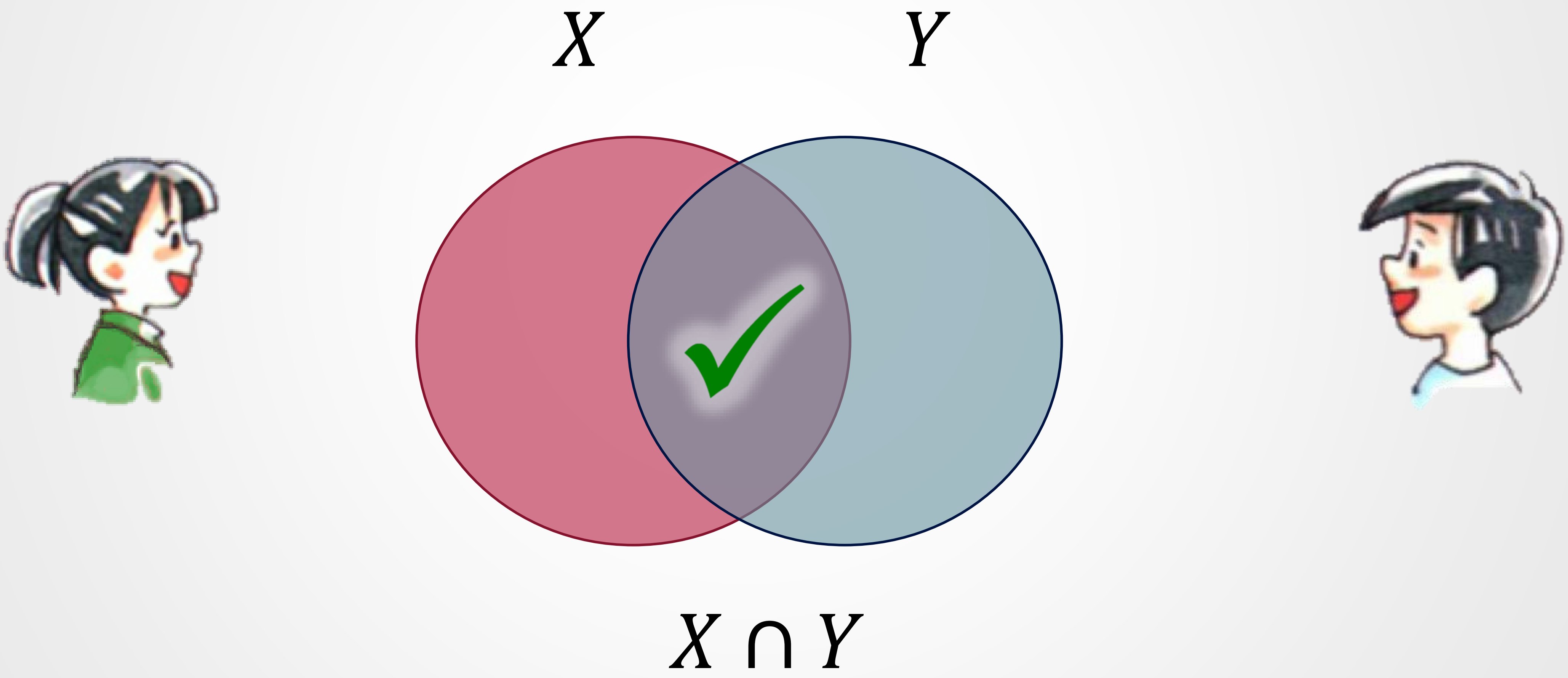
可用不可见 BLINDFOLDED COMPUTING

Zhicong Huang

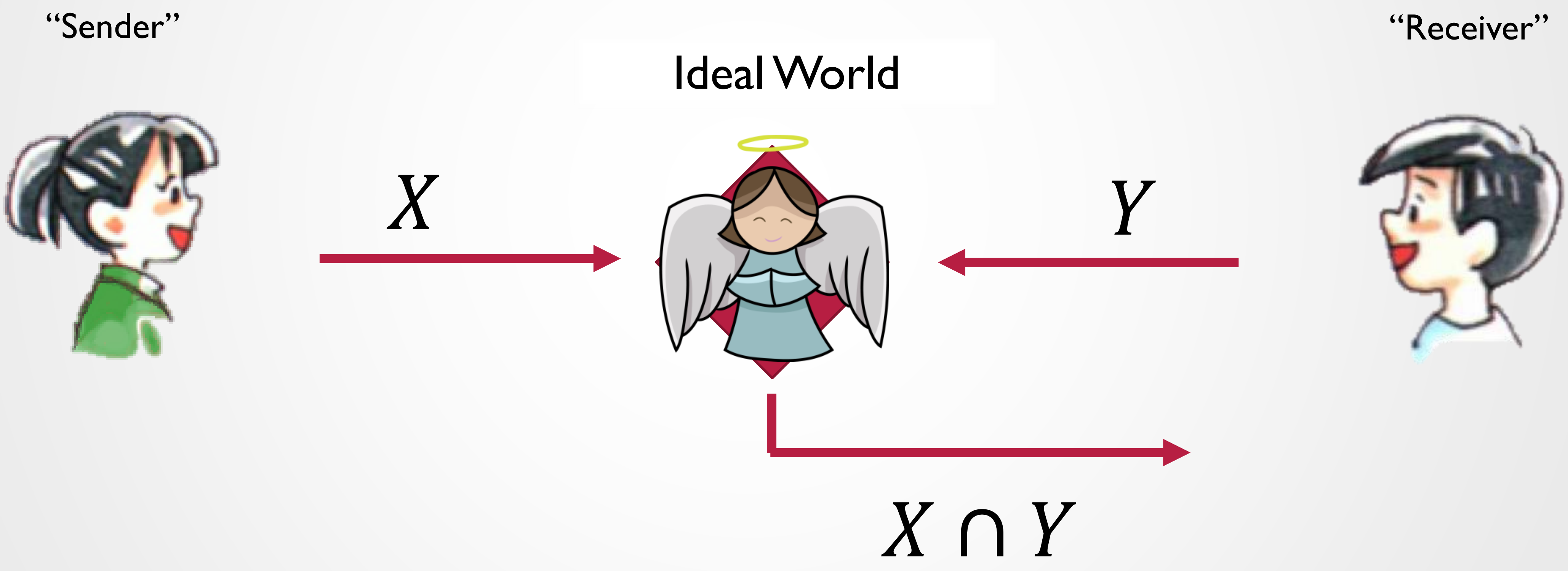
[CLR17] **FAST PRIVATE SET INTERSECTION FROM HOMOMORPHIC ENCRYPTION** - CCS 2017

[CLHR18] **LABELED PSI FROM FULLY HOMOMORPHIC ENCRYPTION WITH MALICIOUS SECURITY** - CCS 2018

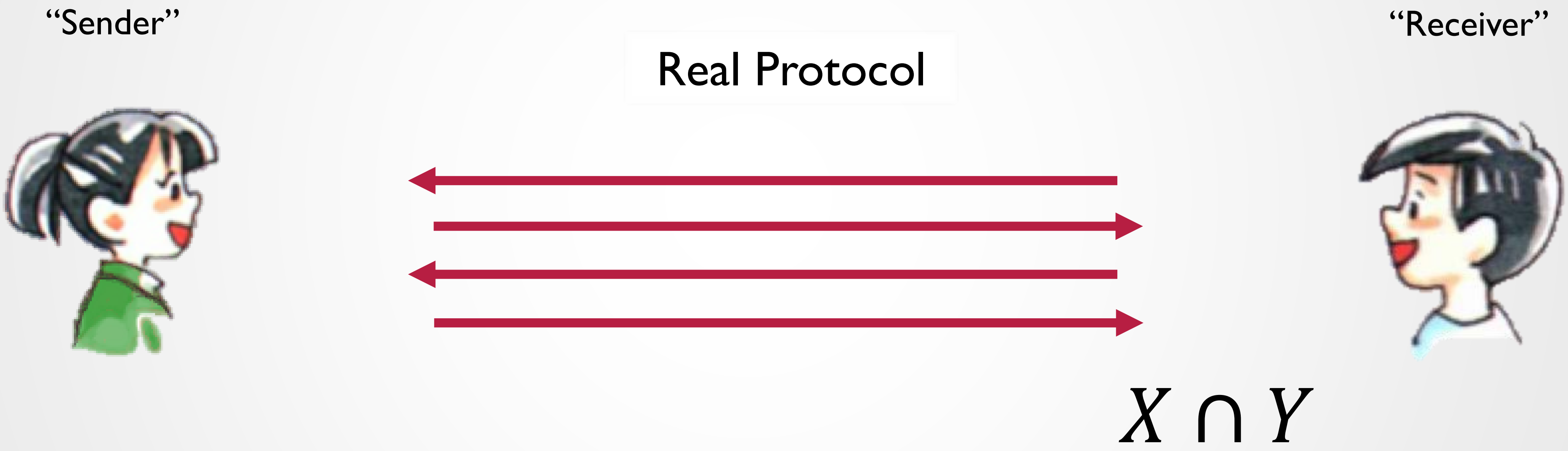
Private Set Intersection (PSI)



Private Set Intersection (PSI)

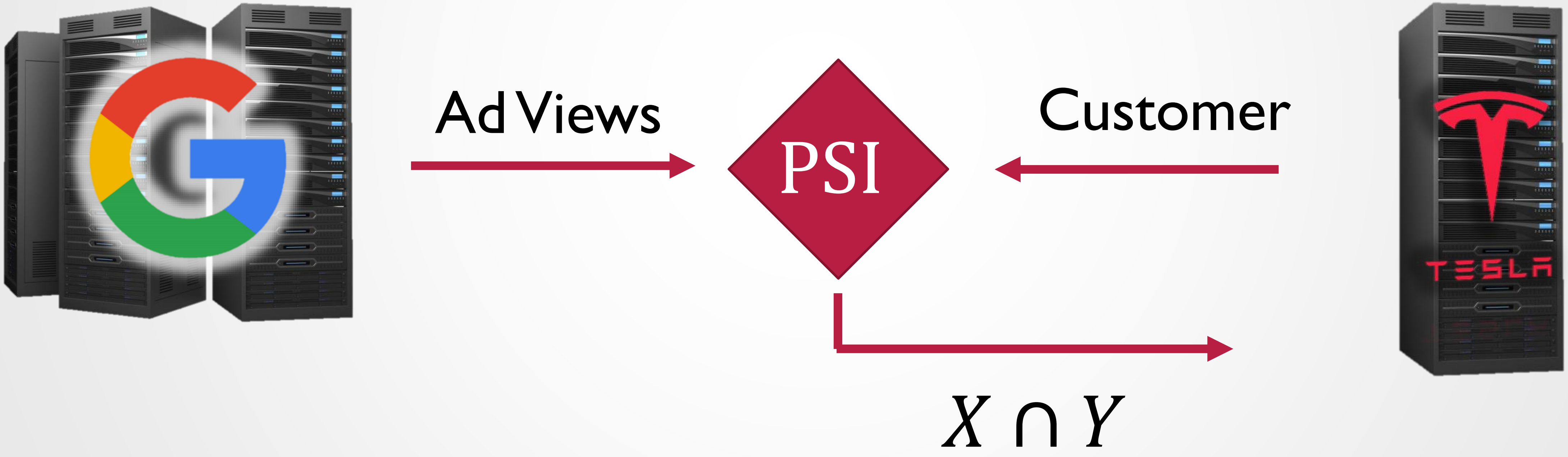


Private Set Intersection (PSI)

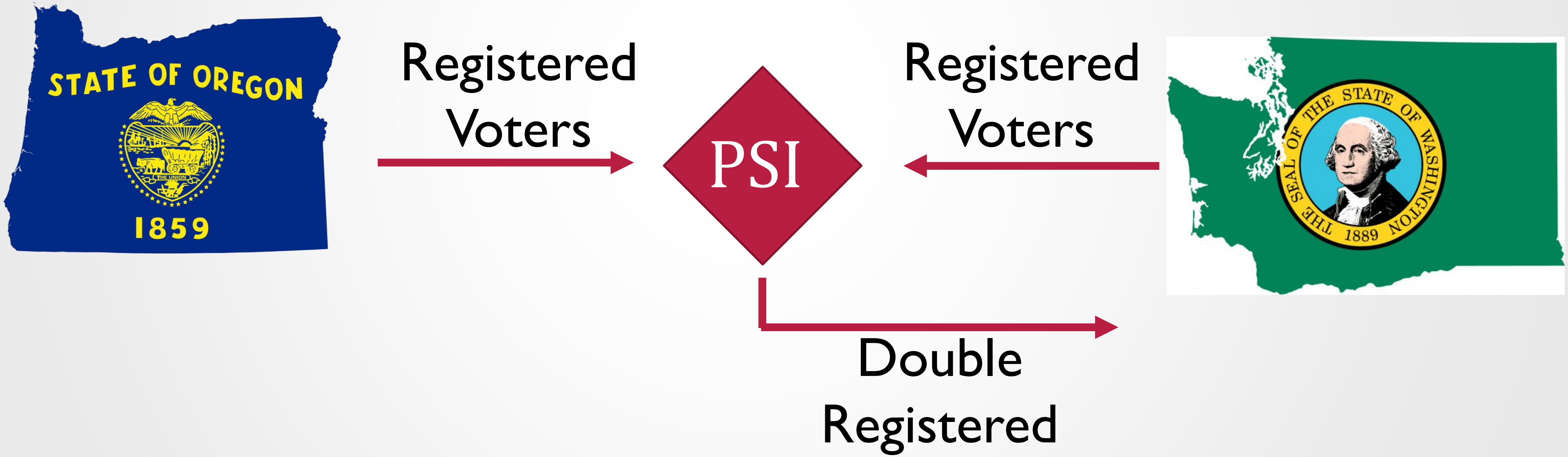


- Adversary Types:
  - Semi-honest – follows the protocol
    - Sends the correct messages
  - Malicious – may deviate from the protocol
    - Can send the incorrect

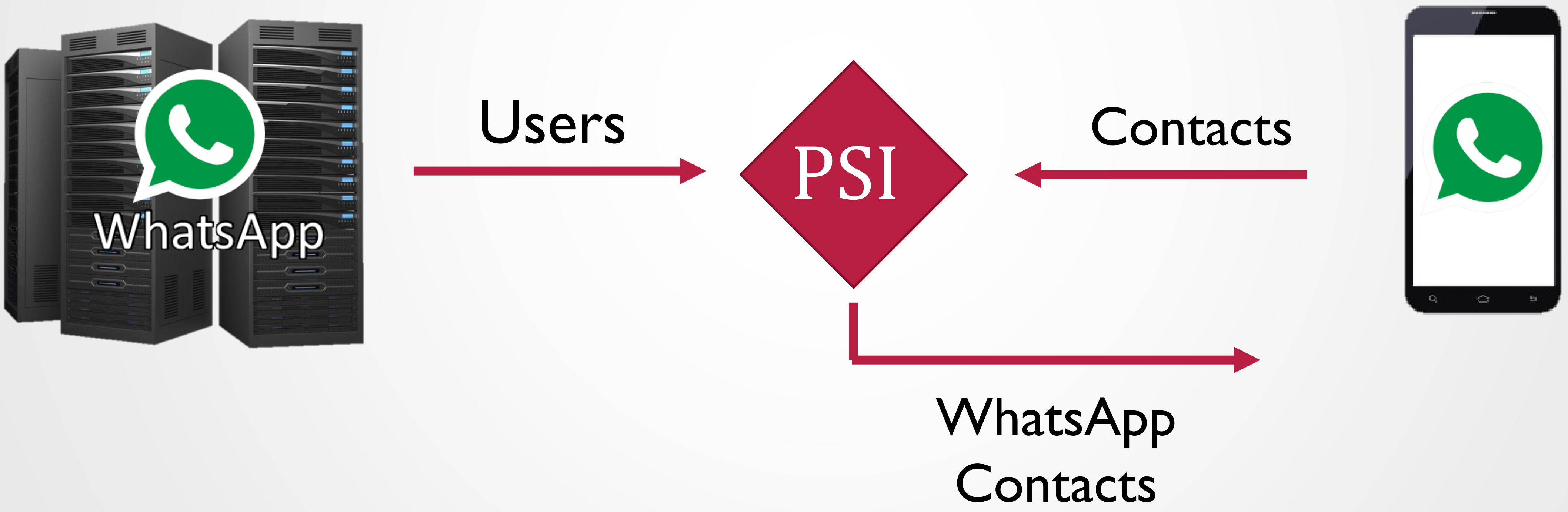
App: Ad Efficiency



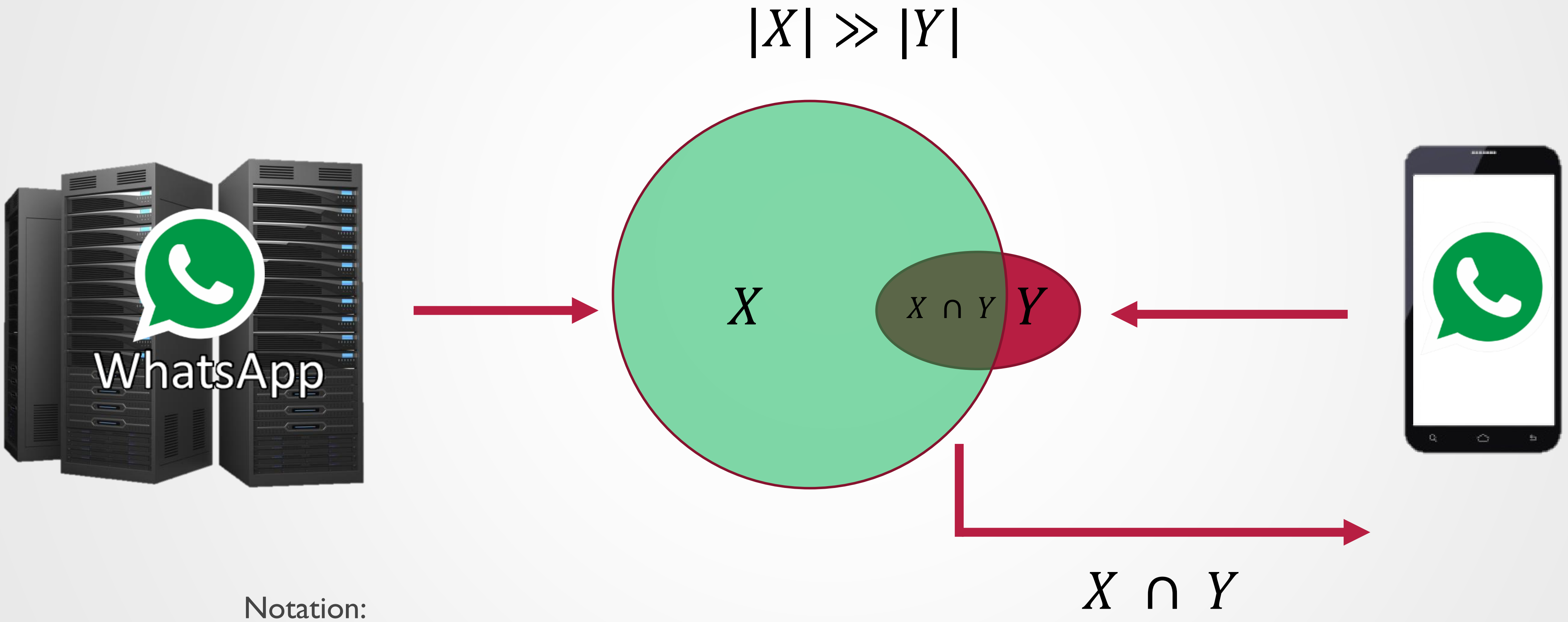
App: Voter Registration



App: Contact discovery



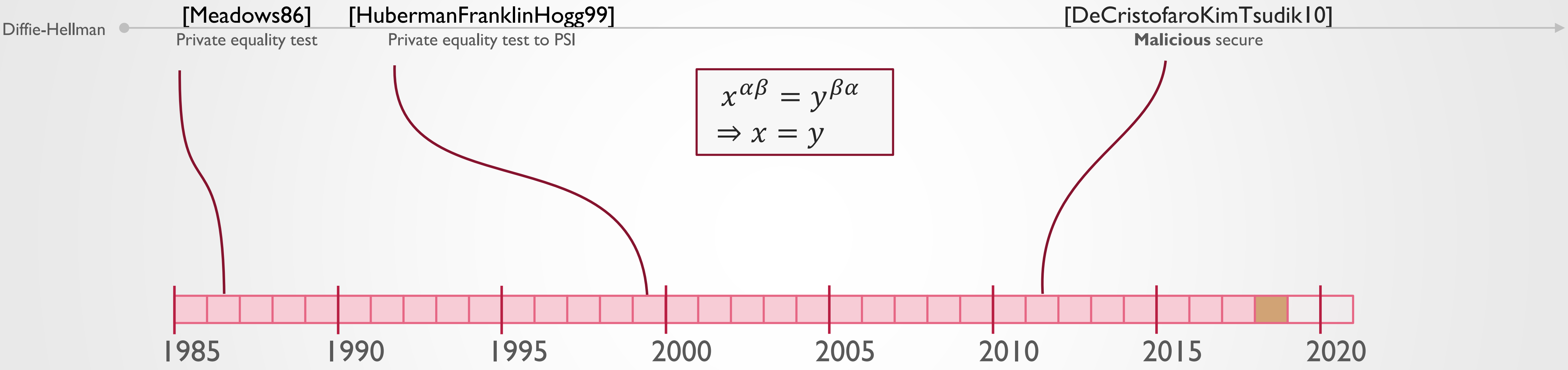
App: Contact discovery



Notation:

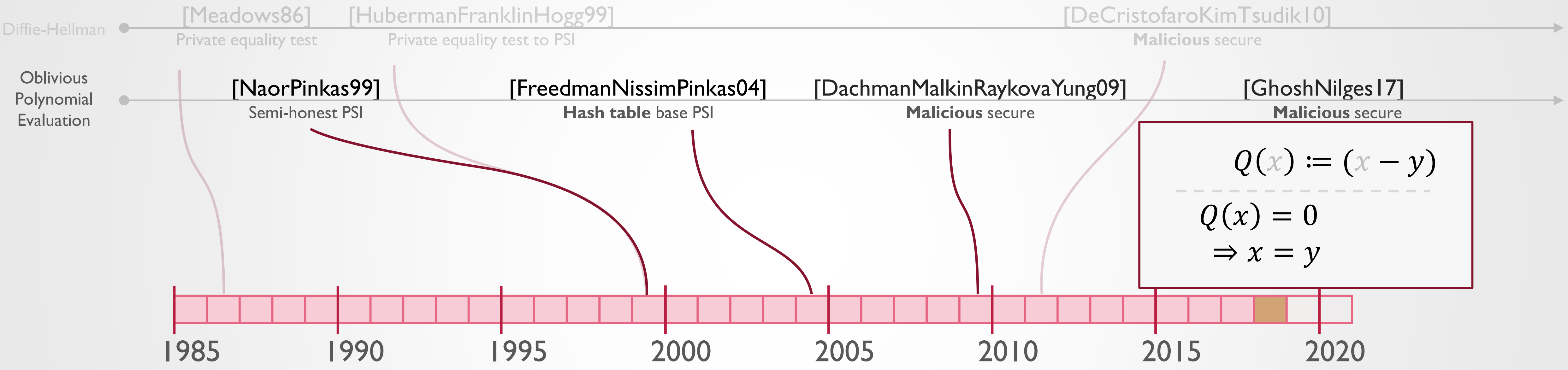
- $N = |X|$
- $n = |Y|$

A *Sampling* of PSI Over the Decades

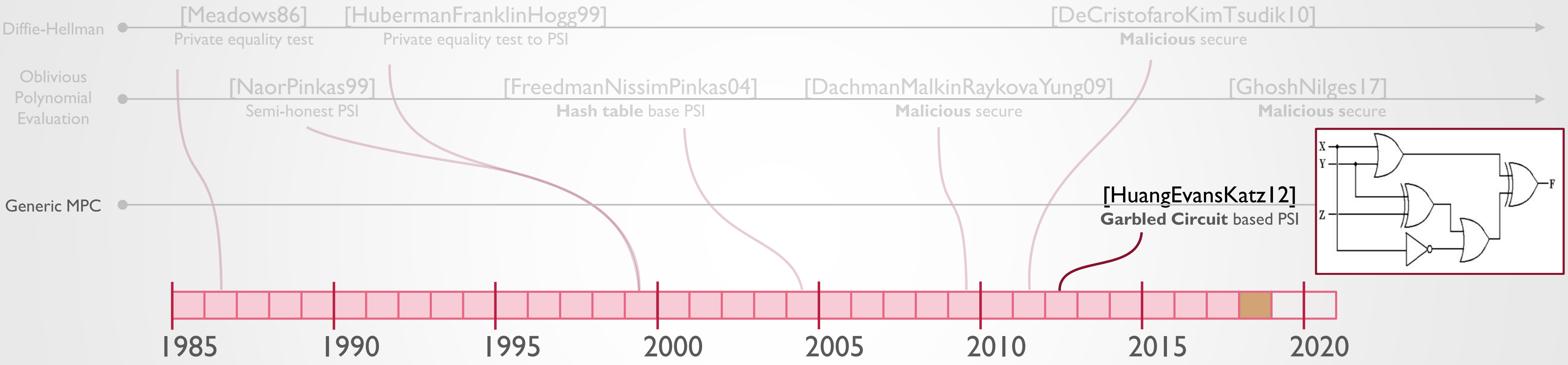


$$x^{\alpha\beta} = y^{\beta\alpha}$$
$$\Rightarrow x = y$$

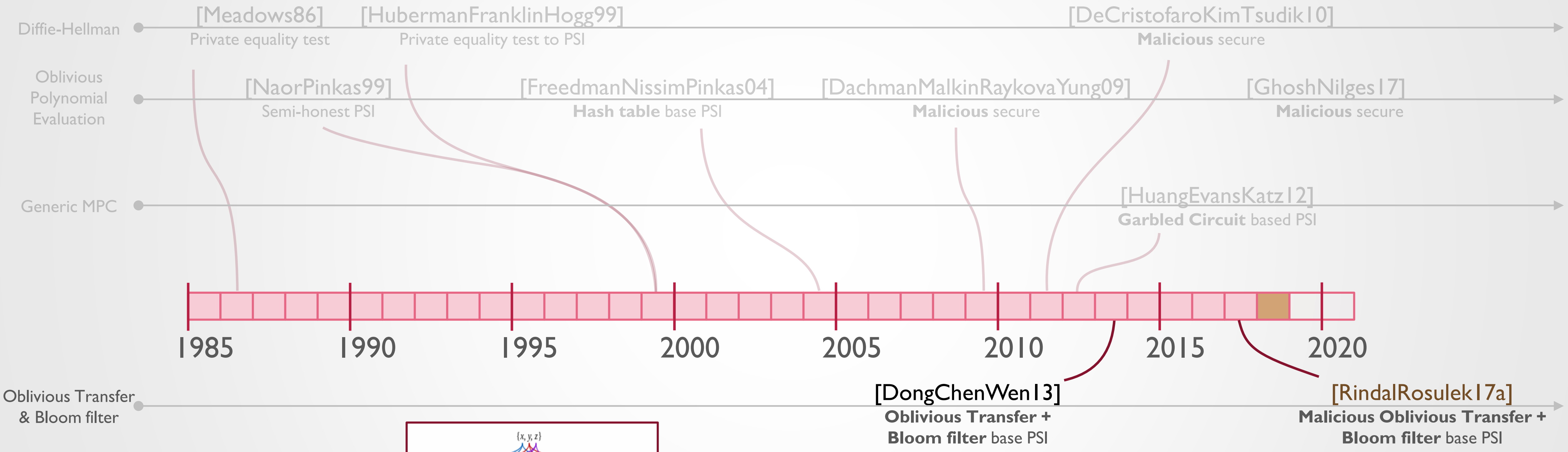
A *Sampling* of PSI Over the Decades

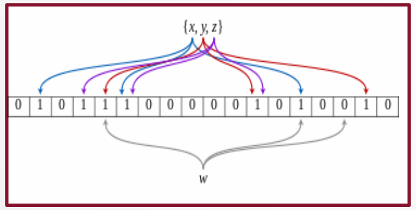


A *Sampling* of PSI Over the Decades

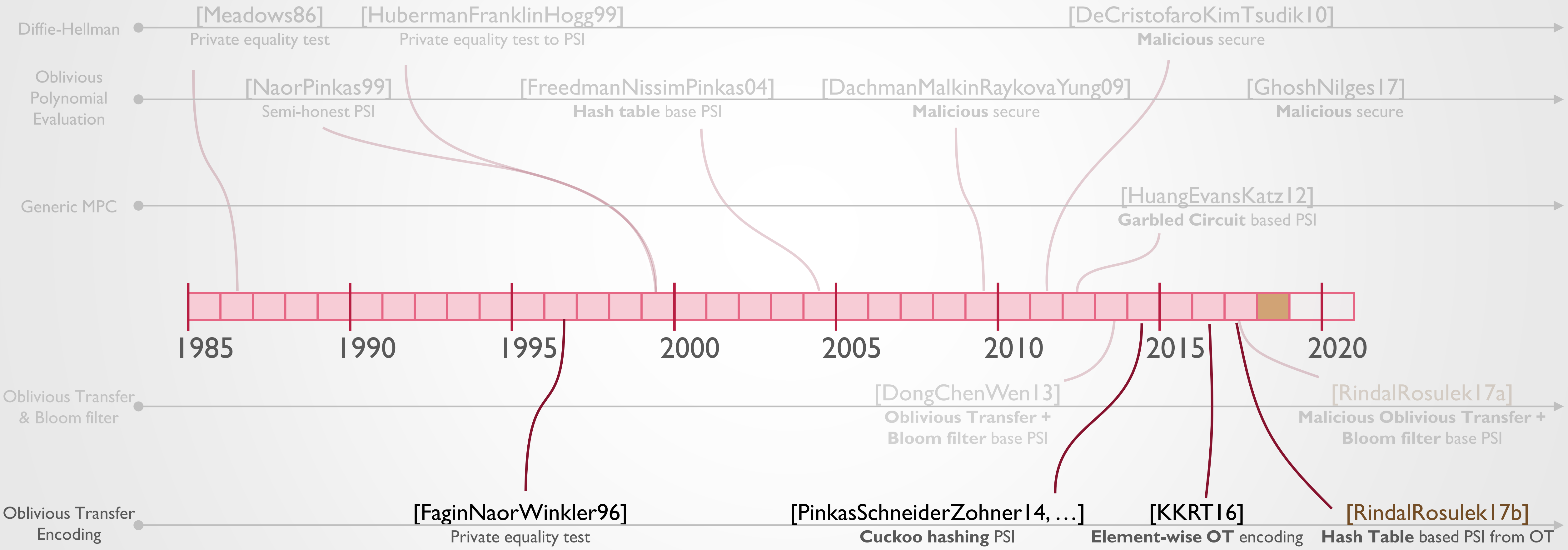


A *Sampling* of PSI Over the Decades

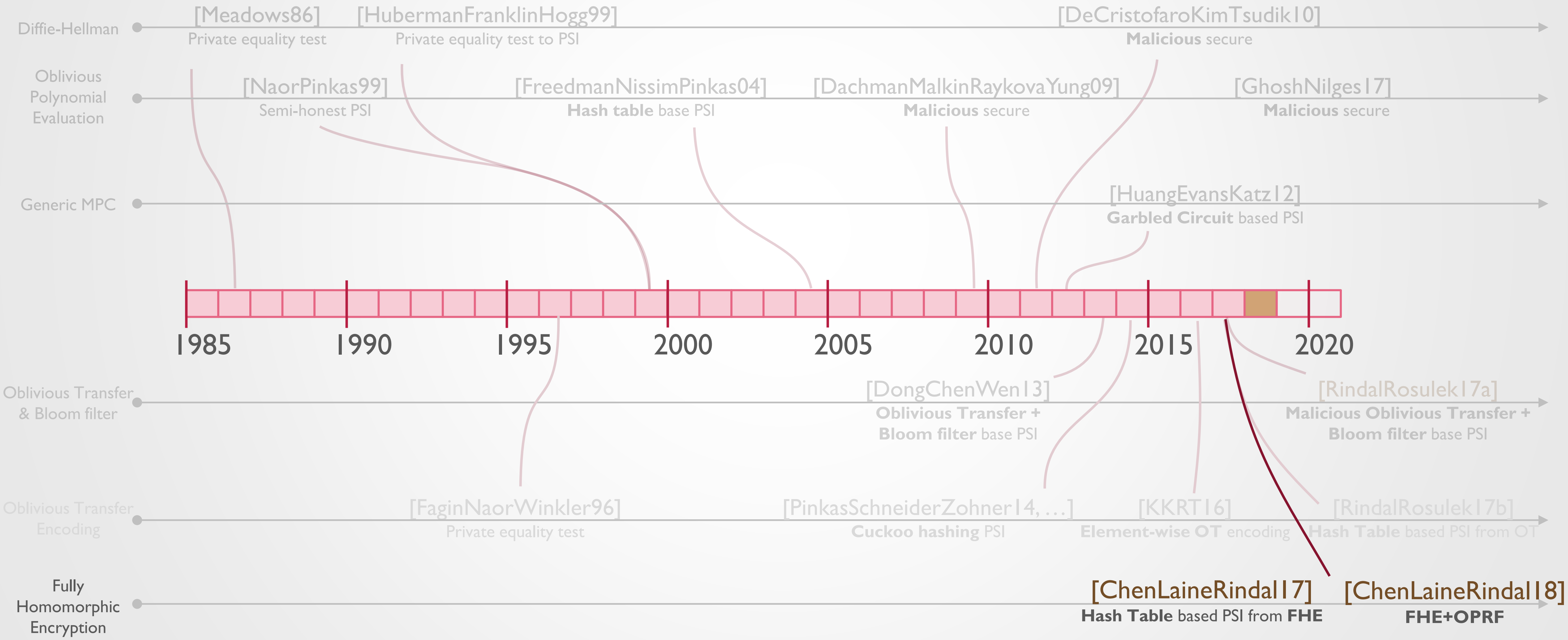




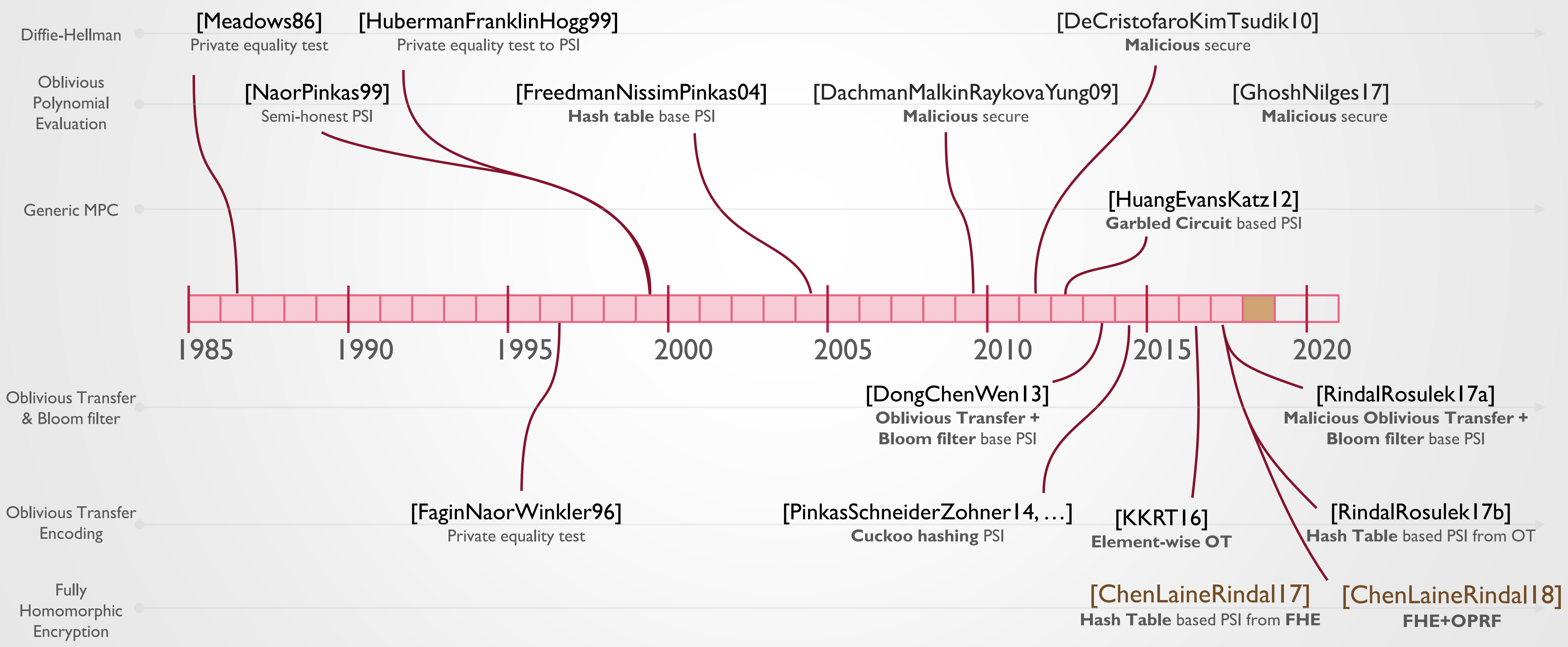
A *Sampling* of PSI Over the Decades



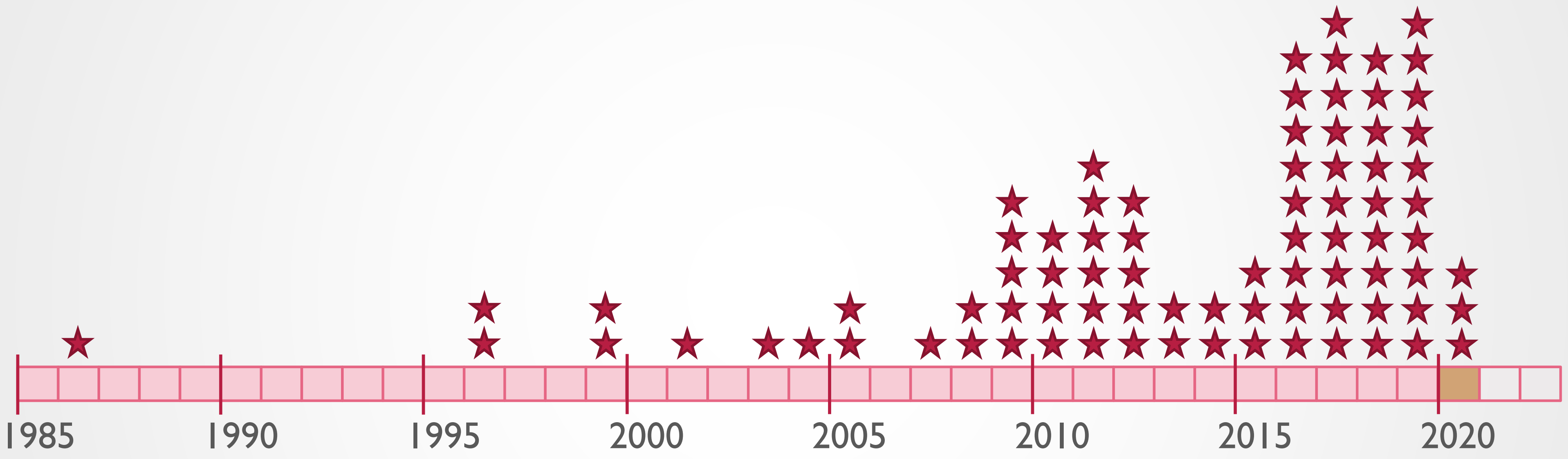
A *Sampling* of PSI Over the Decades



A *Sampling* of PSI Over the Decades

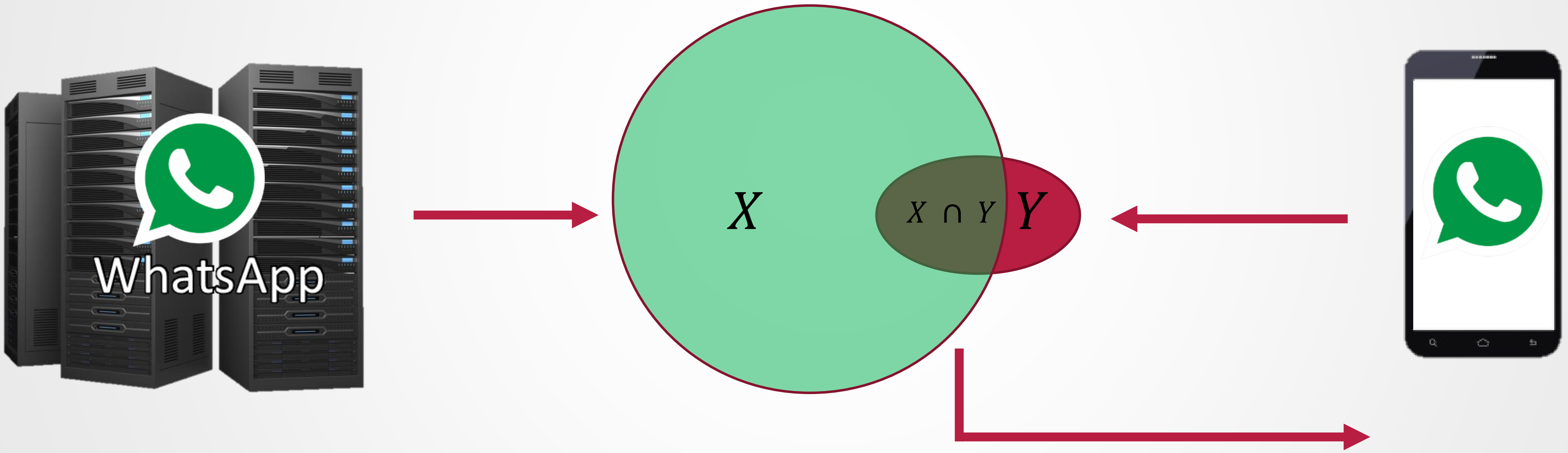


A *Sampling* of PSI Over the Decades



App: Contact discovery

$|X| \gg |Y|$



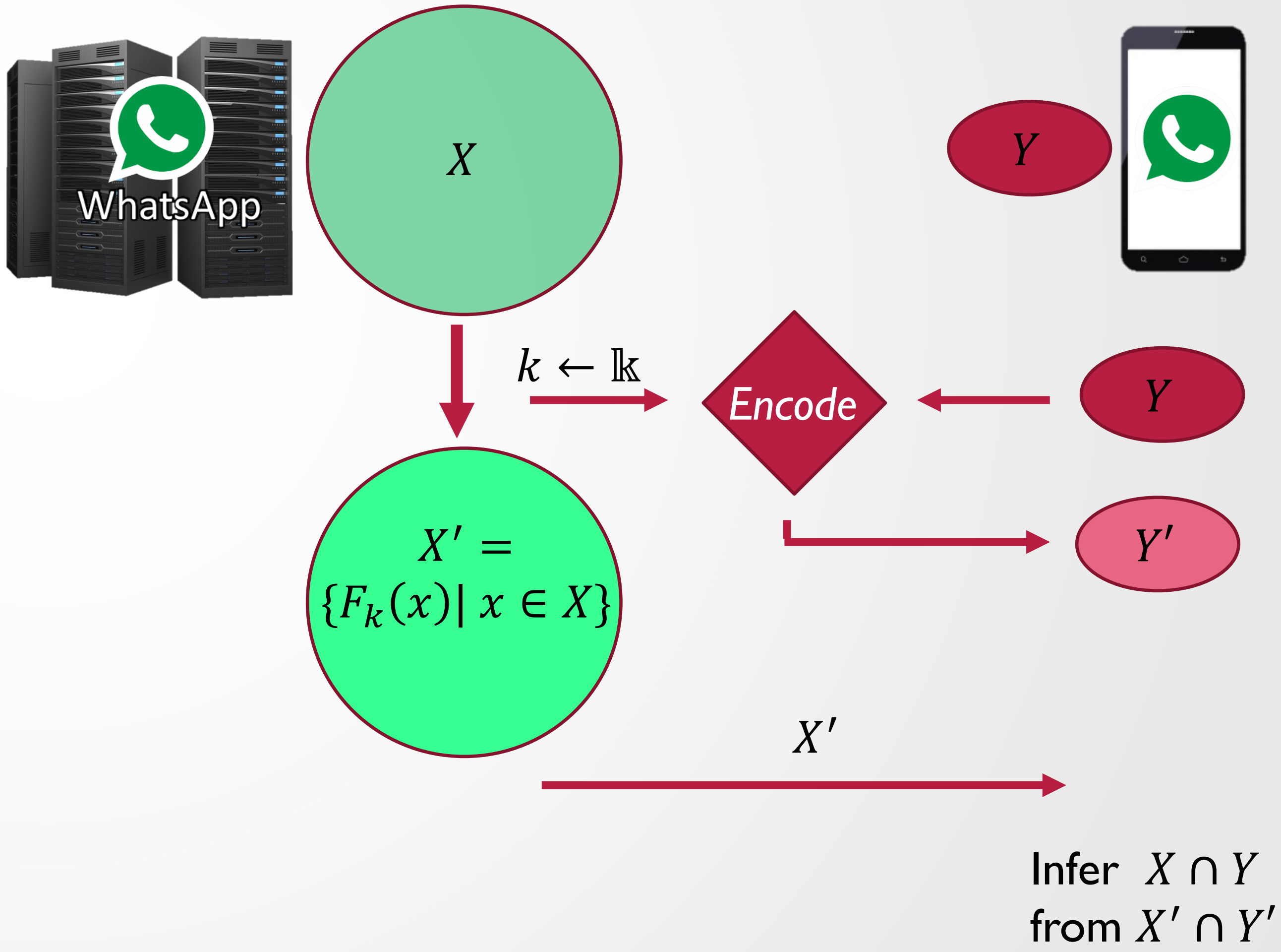
$X \cap Y$

Notation:

- $N = |X|$
- $n = |Y|$

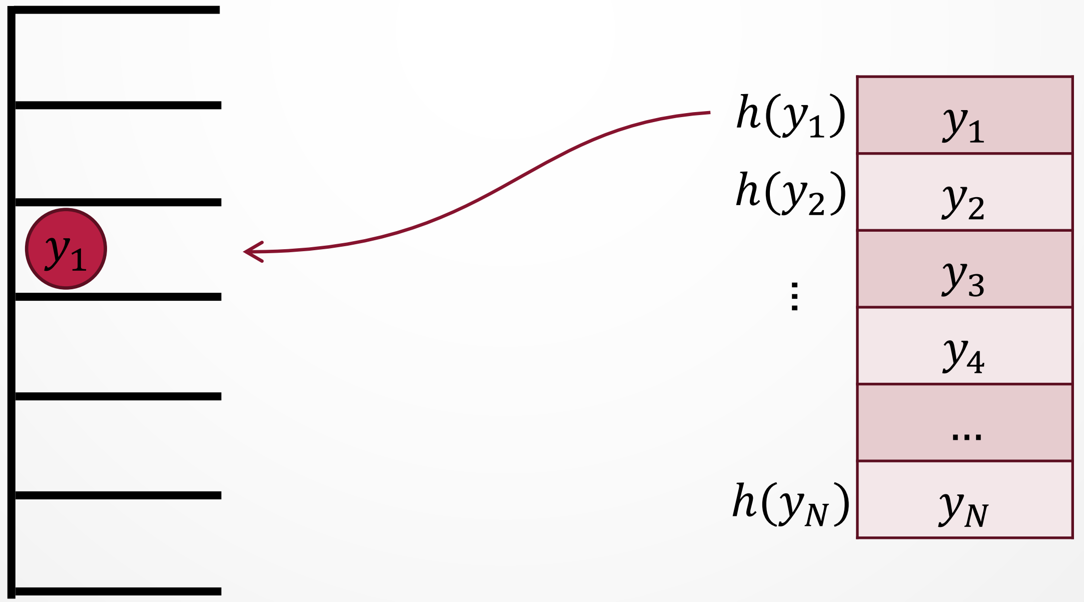
Shortcomings of Prior Work

- Communication linear in both sets
  $O(|X| + |Y|)$
  - What about $|X| \gg |Y|$?
  - Insecure solution:
    - Send small set to other party
    - Comm. = $O(\min(|X|, |Y|))$
  - Can we match this?
    - In theory: Yes
    - In practice: Almost…



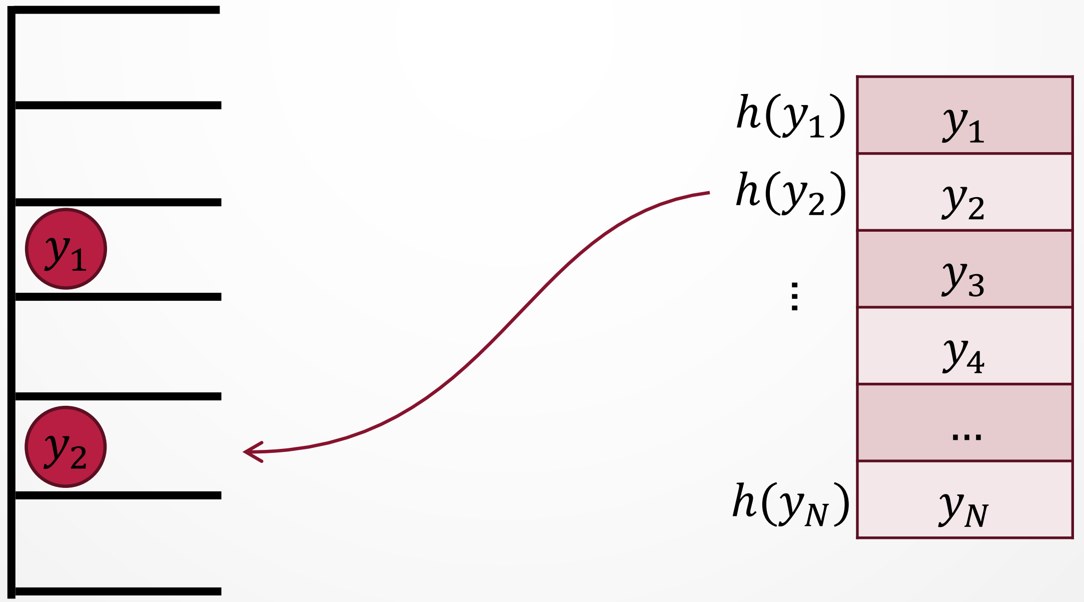
Cuckoo Hashing

- Hash table technique with $O(1)$ lookup time



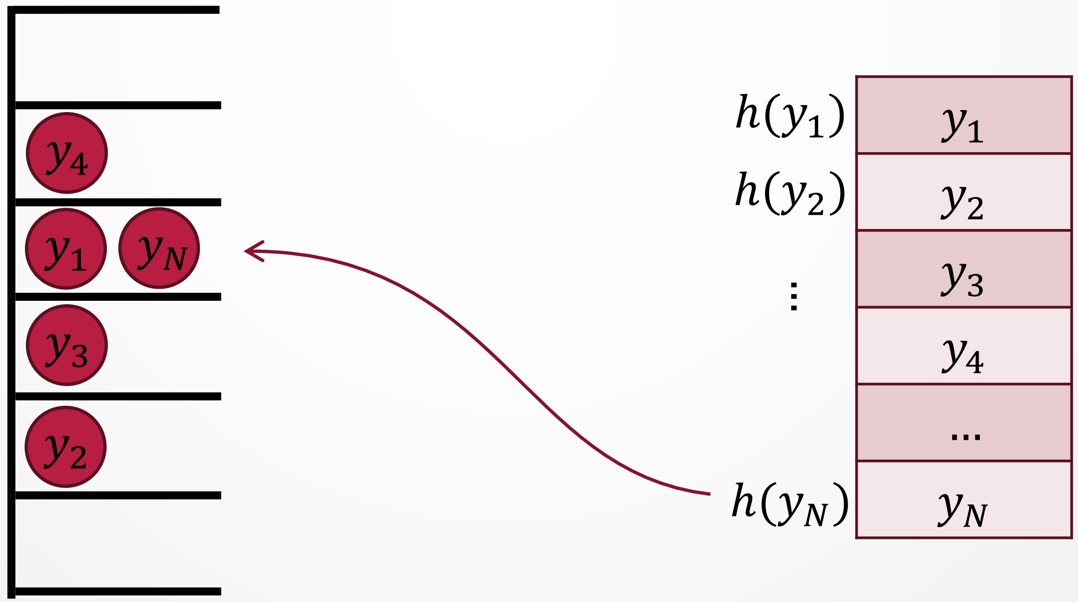
Cuckoo Hashing

- Hash table technique with $O(1)$ lookup time



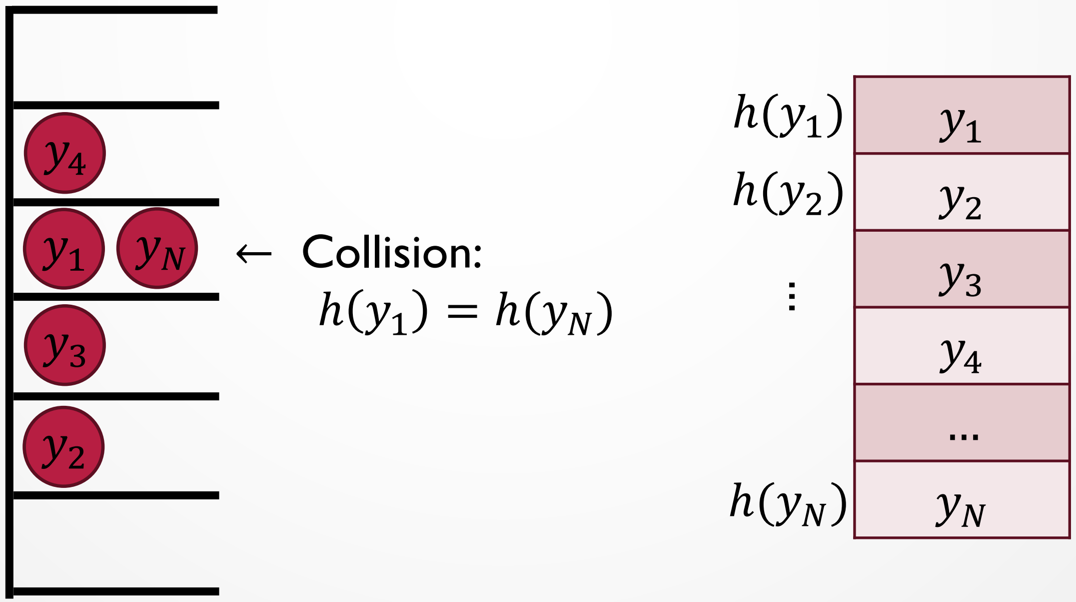
Cuckoo Hashing

- Hash table technique with $O(1)$ lookup time



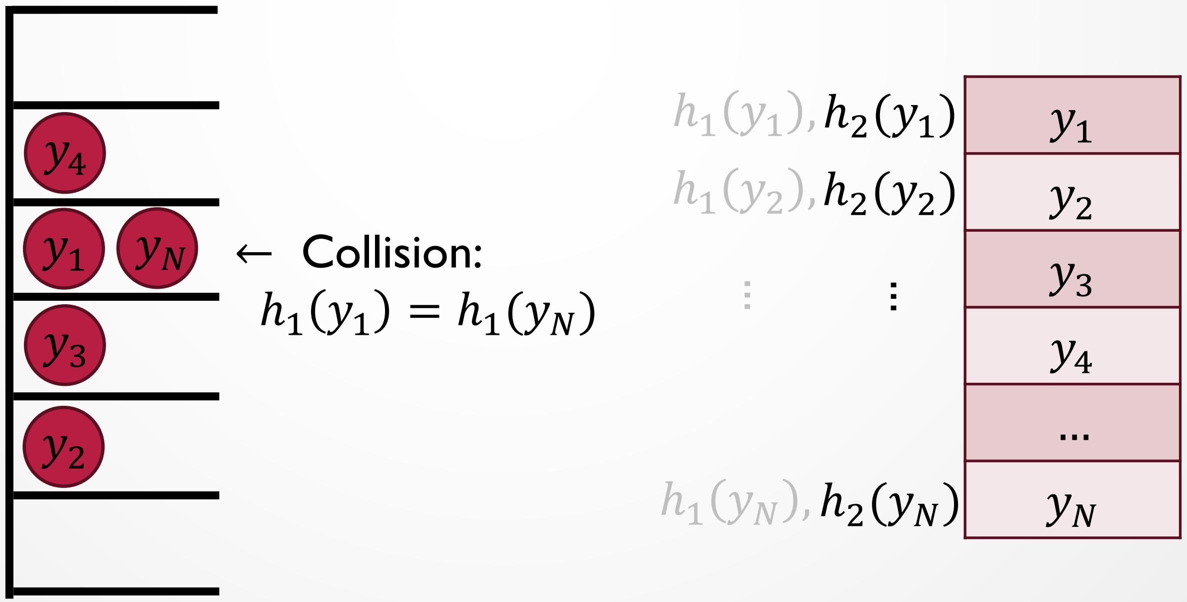
Cuckoo Hashing

- Hash table technique with $O(1)$ lookup time



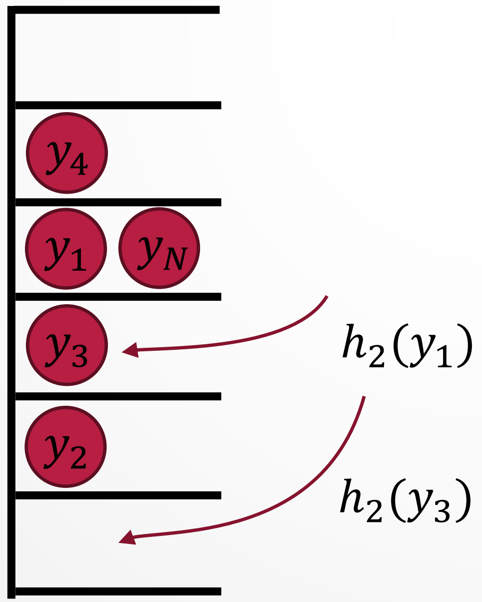
Cuckoo Hashing

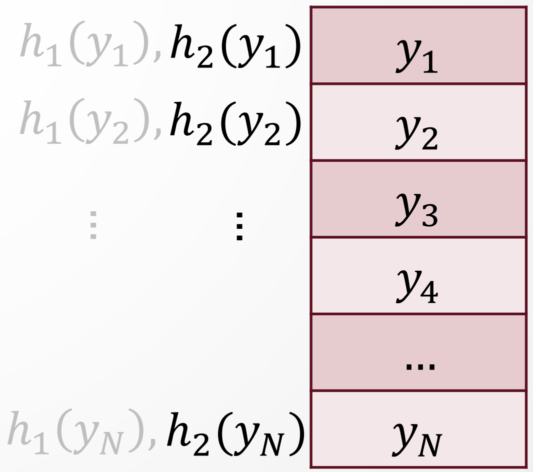
- Hash table technique with $O(1)$ lookup time



Cuckoo Hashing

- Hash table technique with $O(1)$ lookup time





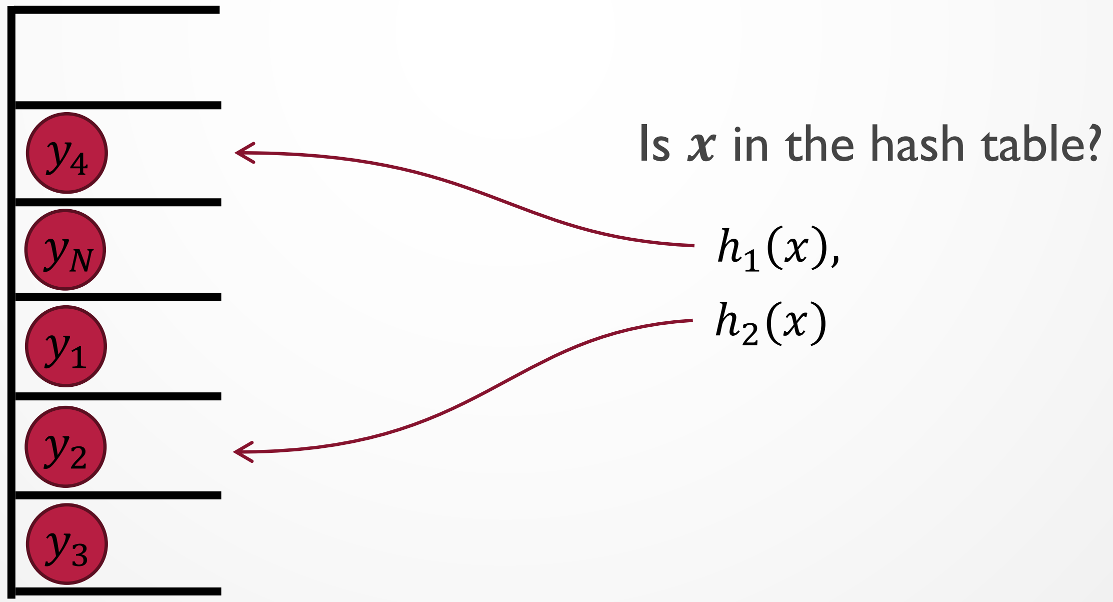
Cuckoo Hashing

- Hash table technique with $O(1)$ lookup time

*With three hash functions

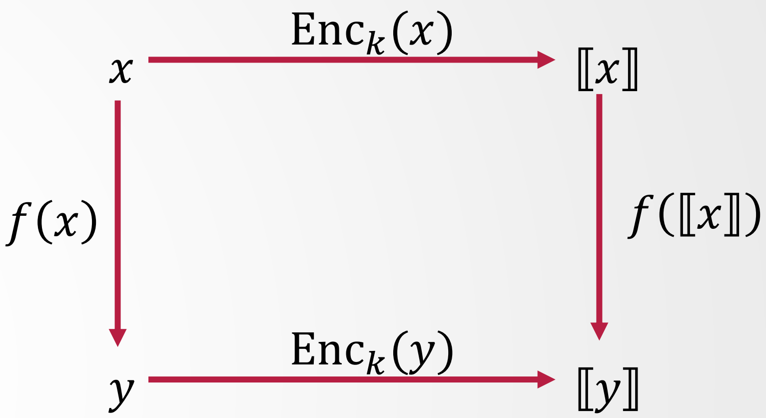
Cuckoo Hashing

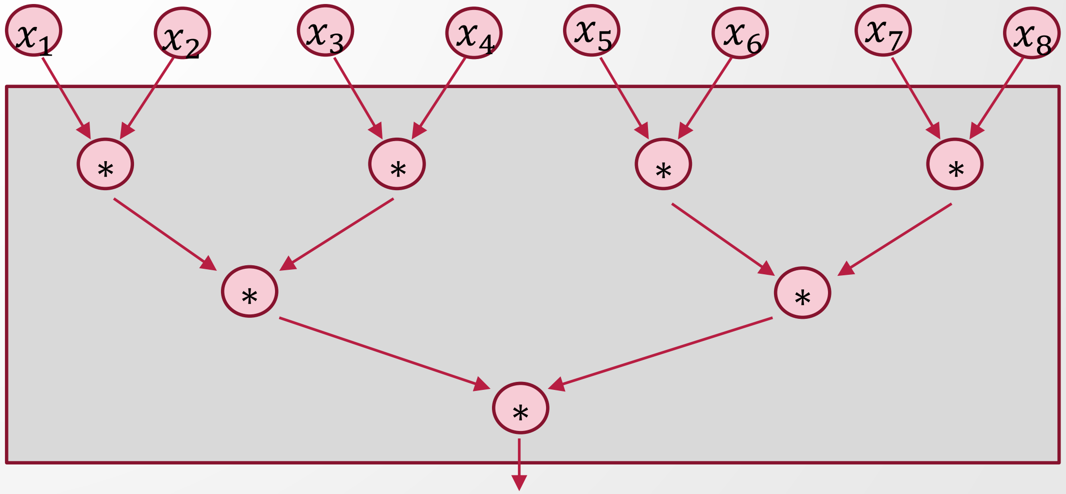
- Hash table technique with $O(1)$ lookup time



Fully Homomorphic Encryption (FHE)

- Encryption technique that allows computation
  - $\text{Enc}_k(f(x)) \equiv f(\text{Enc}_k(x))$
  - f can perform $+, -, *$
  - Addition and subtraction are very cheap.
  - Multiplication is very expensive.
    - Limited multiplication depth
    - E.g. $f(x) = \prod_{i=1}^{8} x_i$
    - Inefficient beyond depth ~6





Equality Test from FHE

[ChenLaineRindal17]

- Want to test if $y = x$

$$[\![y]\!] \coloneqq \mathrm{Enc}_k(y)$$

$$[\![z]\!] \coloneqq [\![y]\!] - x \quad \xrightarrow{[\![z]\!]}$$

$$z = 0 \Leftrightarrow y = x$$

- Issue: Receiver can recover $x = y - z$!
  - Need to randomize z

Equality Test from FHE

[ChenLaineRindal17]

- Want to test if $y = x$

$[\![y]\!] \coloneqq \mathrm{Enc}_k(y)$

Sample $r \leftarrow \mathbb{Z}_p^*$

$[\![z]\!] \coloneqq ([\![y]\!] - x)r$

$[\![z]\!]$

$z = 0 \Leftrightarrow y = x$

- Issue: Receiver can recover $x = y - z$!
  - Need to randomize z
  - Elements are in the prime field $\mathbb{Z}_p = \{0,1, \ldots, p-1\}$
  - For a random $r \in \mathbb{Z}_p^* = \{1, \ldots, p-1\}$
    - xr is a random elements in $\mathbb{Z}_p^*$, given non-zero x

Membership from FHE

[ChenLaineRindal17]

- Want to test if $y \in X$

Sample $r \leftarrow \mathbb{Z}_p^*$

$$[\![y]\!] := \mathrm{Enc}_k(y)$$

$$[\![z]\!] := r \prod_{x \in X} ([\![y]\!] - x)$$

$$[\![z]\!]$$

$$z = 0 \Leftrightarrow y \in X$$

- Issue: Depth of the computation is $\log N = \log |X|$
  - E.g. $N = 2^{28} \;\Rightarrow\; \text{depth} = 28 > 6$
- Observe the polynomial
  - Symmetric poly. ⇒ efficiently computable
- Need to compute y^N in low degree…

$$[\![z]\!] := f(y) = r \prod_{x \in X} (y - x)$$
$$= a_N\, y^N + \cdots + a_2 y^2 + a_1 y + a_0$$

Windowing: computing y^N in low depth

- Need to compute $[\![z]\!] \coloneqq a_N\, y^N + \cdots + a_2 y^2 + a_1 y + a_0$
- Depth $\log N$ solution, send $[\![y]\!]$ and compute:
  - $[\![y^2]\!] = [\![y]\!][\![y]\!]$
  - $[\![y^4]\!] = [\![y^2]\!][\![y^2]\!]$
  - …
- Depth 0 solution, send all $[\![y]\!], [\![y^2]\!], \dots, [\![y^N]\!]$
  - $O(N)$ communication…
- Depth $\mathbf{\log\log N}$ soltuion, send $[\![y]\!], [\![y^2]\!], [\![y^4]\!], \dots, \left[\!\!\left[y^{2^i}\right]\!\!\right], \dots, \left[\!\!\left[y^{2^{\log N}}\right]\!\!\right]$
  - Compute all other powers in depth $\log\log N$
    - E.g. $[\![y^7]\!] = [\![y^4]\!][\![y^2]\!]\,[\![y]\!]$
    - E.g. $N = 2^{28} \Rightarrow \text{ depth} = 5$
  - $O(\log N)$ communication.

Membership from FHE

[ChenLaineRindal17]

- Want to test if $y \in X$:

$\{[\![y^i]\!] \mid i = 1,2,4,8,\ldots,\log N\}$ ←

Compute all $[\![y^i]\!]$

$[\![z]\!] := r\prod_{x\in X}([\![y]\!] - x)$

$:= a_N\, y^N + \cdots + a_1 y + a_0$

$[\![z]\!]$ →

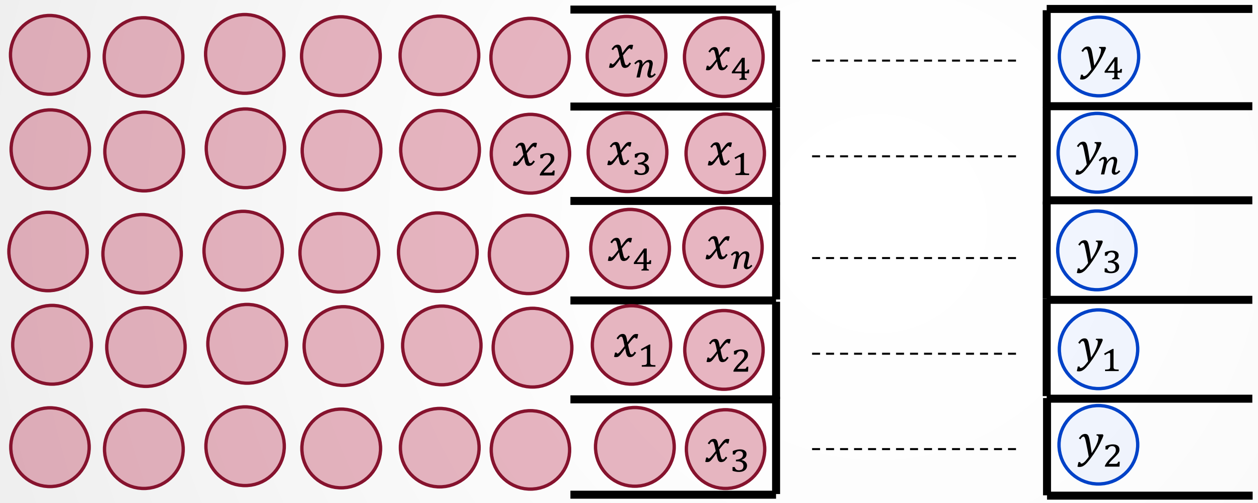
$z = 0 \Leftrightarrow y \in X$

- Performance,
  - Computation = $O(N)$
  - Depth = $O(\log\log N)$
  - Communication = $O(\log N)$
- Set intersection: For $y \in Y$, run set membership protocol
  - Require $O(nN)$ computation!!
  - Where $n = |Y|$,
  - e.g. $n = 1000$

Cuckoo Hashing

[PinkasScheiderZohner14, ChenLaineRindal17]

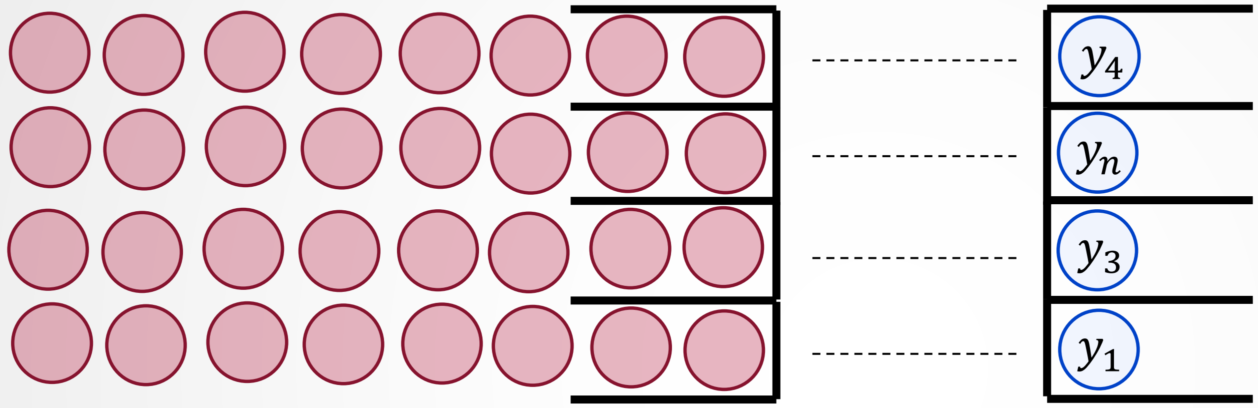
- Receiver performs Cuckoo hashing



- Use two hash functions h, h'
- For each bin, perform 1 membership test
  - When $N \gg n$, bin size $O(N/n)$
  - Overall complexity $O(N)$

Optimization: FHE Batching

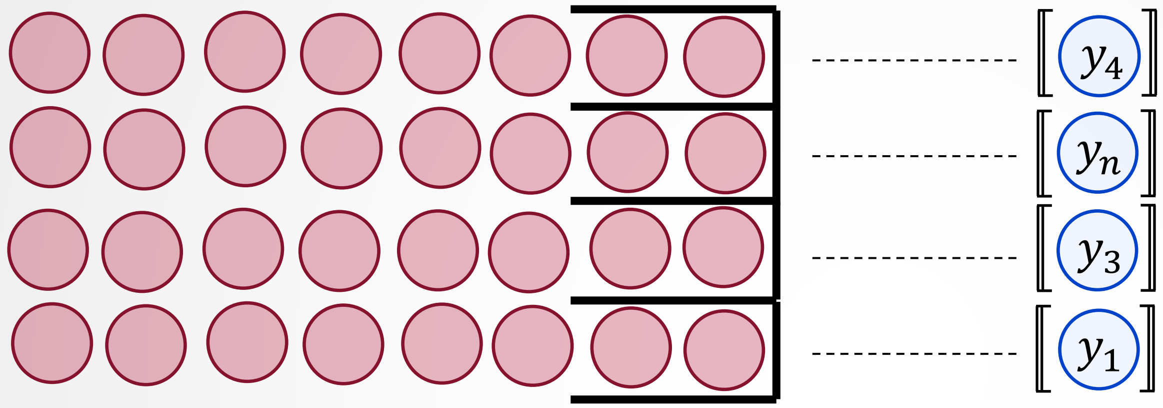
[ChenLaineRindal17]



- Fully homomorphic encryption naturally support "SIMD" type operations
  - A single FHE ciphertext/plaintext can be large…
  - Use Chinese Remainder Theorem (CRT) to pack several items into 1 cipher-text
    - E.g. 4096

Optimization: FHE Batching

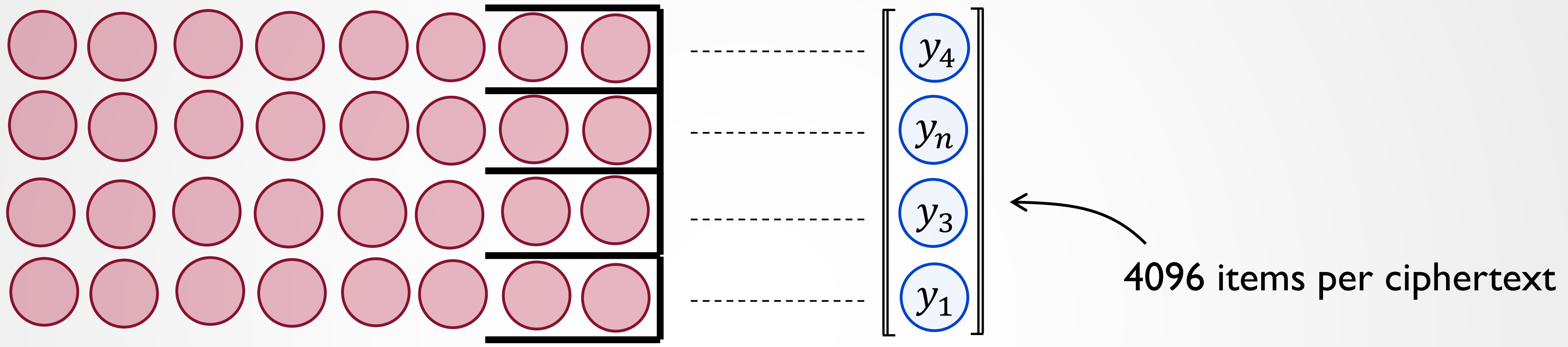
[ChenLaineRindal17]



- Fully homomorphic encryption naturally support "SIMD" type operations
  - A single FHE ciphertext/plaintext can be large…
  - Use Chinese Remainder Theorem (CRT) to pack several items into 1 cipher-text
    - E.g. 4096

Optimization: FHE Batching

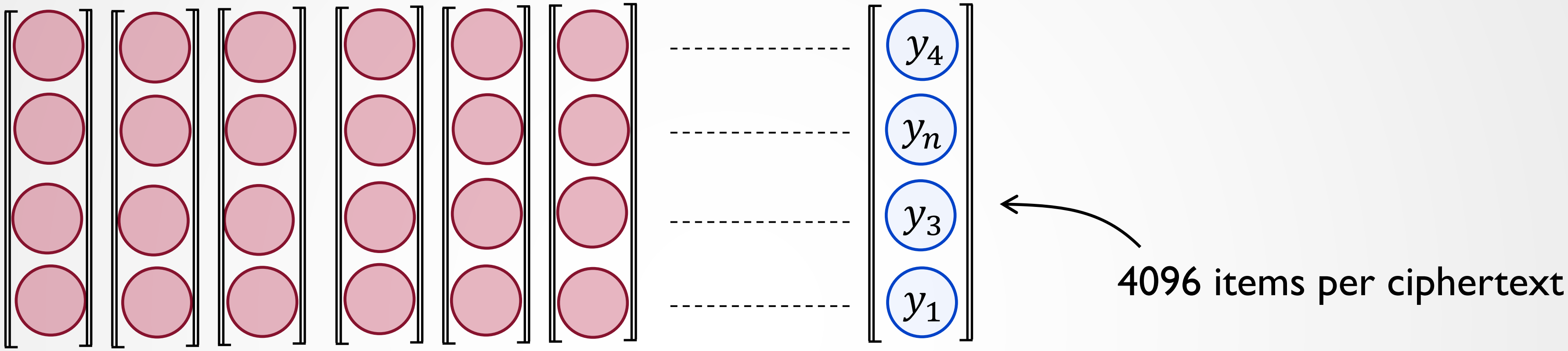
[ChenLaineRindal17]



- Fully homomorphic encryption naturally support "SIMD" type operations
  - A single FHE ciphertext/plaintext can be large…
  - Use Chinese Remainder Theorem (CRT) to pack several items into 1 cipher-text
    - E.g. 4096

Optimization: FHE Batching

[ChenLaineRindal17]



$\{[\![\vec{y}^i]\!] \mid i = 1,2,4,\ldots\}$

$[\![\vec{z}]\!] \coloneqq \vec{r}\prod_{\vec{x}\in\vec{X}}([\![\vec{y}]\!] - \vec{x})$

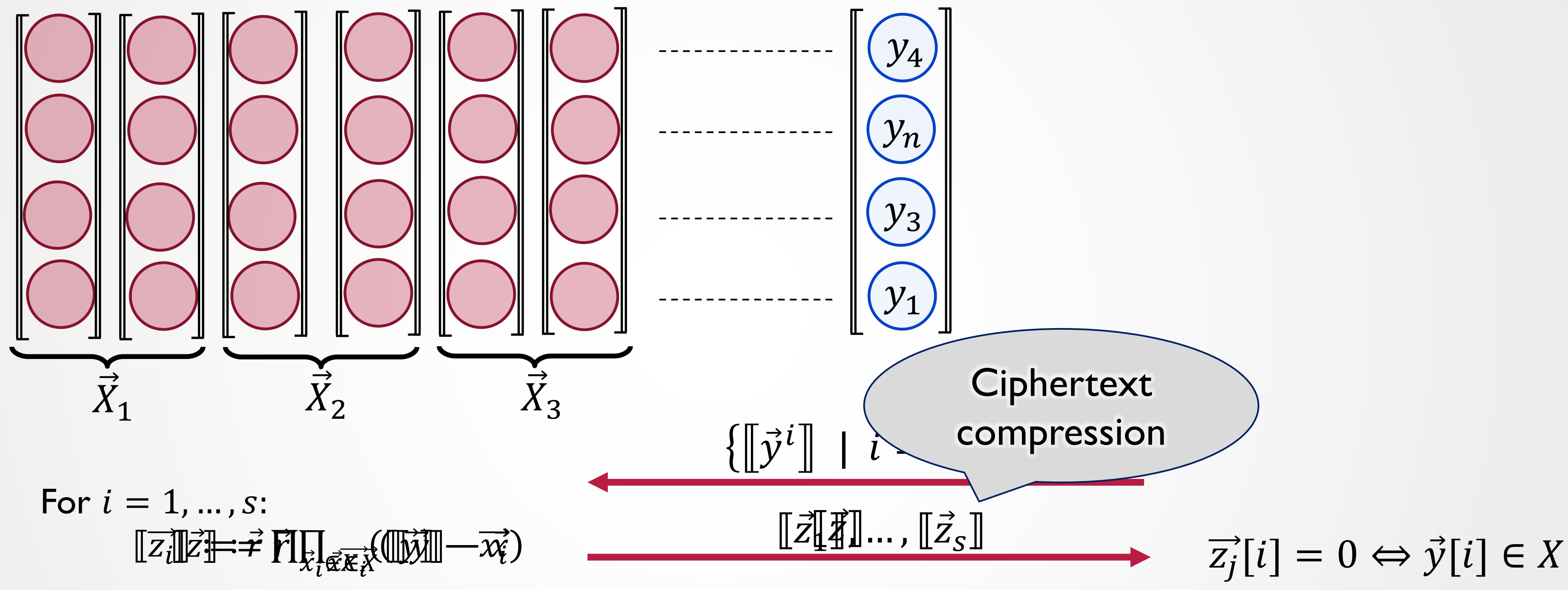
$[\![\vec{z}]\!]$

$\vec{z}[i] = 0 \Leftrightarrow y[i] \in X$

- Fully homomorphic encryption naturally support "SIMD" type operations
  - A single FHE ciphertext/plaintext can be large…
  - Use Chinese Remainder Theorem (CRT) to pack several items into 1 cipher-text
    - E.g. 4096

Optimization: Splitting

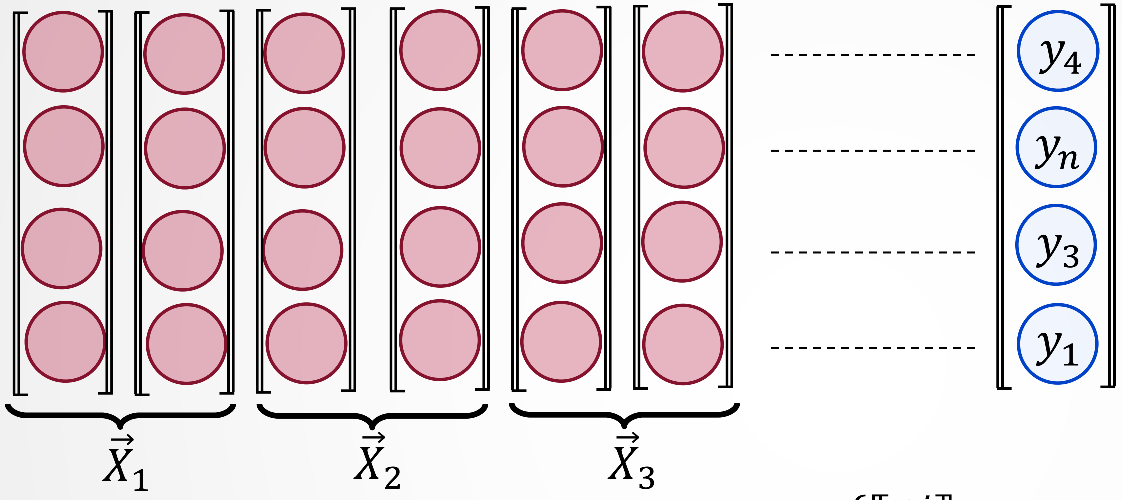
[ChenLaineRindal17]



- Observe that the communication is unbalanced.
- Partition $\vec{X}$ into s splits $\vec{X}_1, \ldots, \vec{X}_s$
  - Reduces depth to $\log\log\frac{N}{ns}$
  - Large impact in practice, e.g. depth =3 .

Final Protocol

[ChenLaineRindal17]



$\{[\![\vec{y}^i]\!] \mid i = 1,2,4,\dots\}$ ←

For $i = 1, \dots, s$:

$$[\![\vec{z_i}]\!] := \vec{r} \prod_{\vec{x}_i \in \vec{X_i}} ([\![\vec{y}]\!] - \vec{x_i})$$

$[\![\vec{z}_1]\!], \dots, [\![\vec{z}_s]\!]$ →

$$\vec{z_j}[i] = 0 \Leftrightarrow \vec{y}[i] \in X$$

- Sender:
  - $O(N)$ Computation w/ quasi-constant depth
  - $O(n \ \log N)$ communication
  - Practical on server
- Receiver:
  - $O(n \log N)$ Encryptions/Decryptions
  - $O(n \ \log N)$ communication
  - Practical on cellphone

Malicious Receiver

[ChenLaineHaungRindal18]

- Want to test if $y \in X$:

$\{[\![y^i]\!] \mid i = 1,2,4,8, \dots, \log N\}$

Compute all $[\![y^i]\!]$

$[\![z]\!]$

$$[\![z]\!] \coloneqq r \prod_{x\in X}([\![y]\!] - x)$$
$$\coloneqq a_N\, y^N + \cdots + a_1 y + a_0$$

$z = 0 \Leftrightarrow y \in X$

Malicious Receiver

[ChenLaineHaungRindal18]

- Want to test if $y \in X$:

$\{[\![y^i]\!] \mid i = 1,2,4,8,\ldots,\log N\}$ ←

Compute all $[\![y^i]\!]$

$[\![z]\!] := r \prod_{x \in X}([\![y]\!] - x)$
$:= a_N\, y^N + \cdots + a_1 y + a_0$

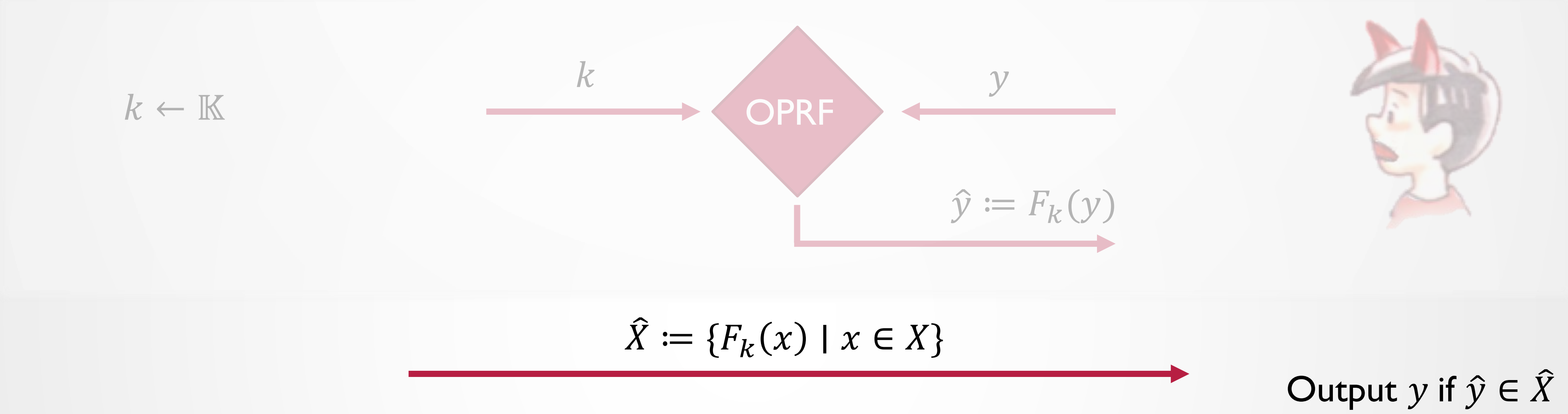
$[\![z]\!]$ →

$z = 0 \Leftrightarrow y \in X$

- Receiver can send bad ciphertexts
  - Leaks information about the sender's set

OPRF Preprocessing

[ChenLaineHaungRindal18, JareckiLiu10]



- Oblivious Pseudorandom Function (OPRF)
  - Sender picks secret key k
  - Receiver learns $F_k(y)$
    - All other output are unpredictable
  - Sender learns nothing
- [JareckiLiu10] constructed PSI from an OPRF
  - Receiver learns $F_k(y)$ from OPRF
  - Send $\hat{X} = \{F_k(x) \mid x \in X\}$ to the receiver
- Issue, communication proportional to $N = |X|$

OPRF Preprocessing

[ChenLaineHaungRindal18]

$k \leftarrow \mathbb{K}$

$\hat{X} \coloneqq \{F_k(x) \mid x \in X\}$

k → OPRF ← y

$\hat{y} \coloneqq F_k(y)$

- Want to test if $\hat{y} \in \hat{X}$:

A function of $\hat{X}$ which is safe to send to the receiver

$\{[\![\hat{y}^i]\!] \mid i = 1,2,4,8, \ldots, \log N\}$

Compute all $[\![\hat{y}^i]\!]$

$[\![z]\!] \coloneqq r \prod_{\hat{x} \in \hat{X}} ([\![\hat{y}^i]\!] - \hat{x})$

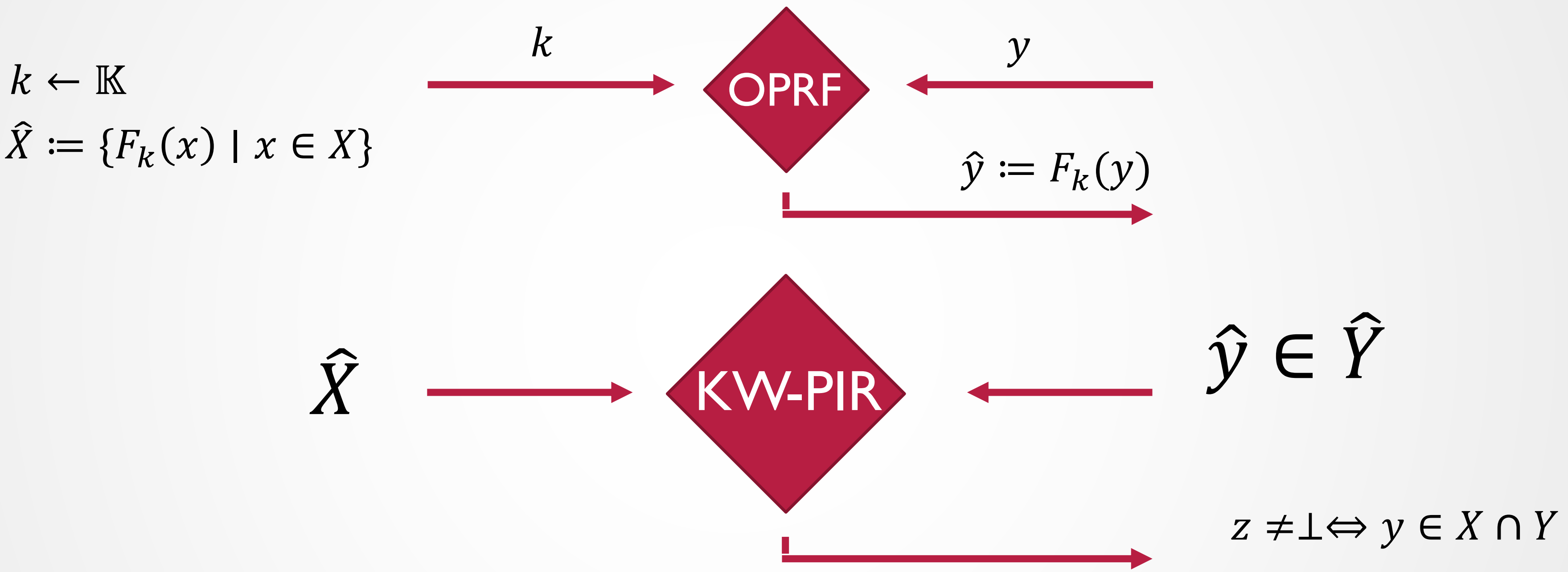
$\coloneqq a_N\, y^N + \cdots + a_1 y + a_0$

$[\![z]\!]$

$z = 0 \Leftrightarrow y \in X$

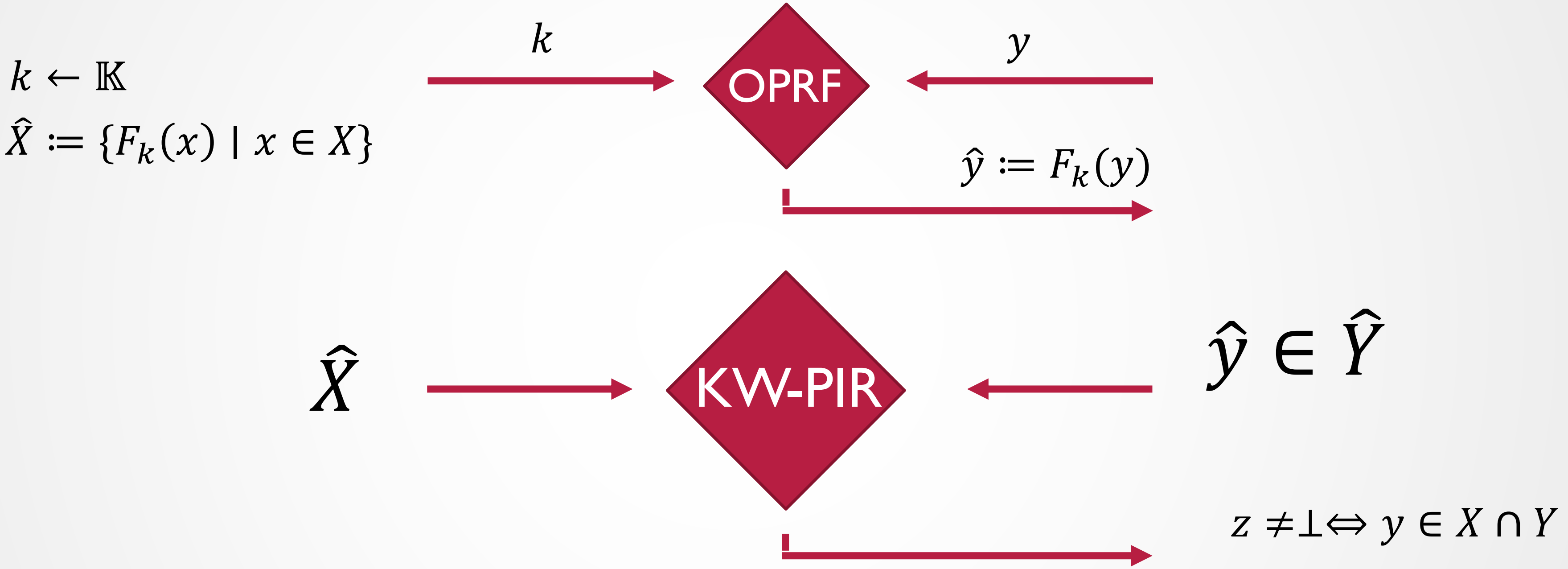
OPRF Preprocessing

[ChenLaineHaungRindal18]



OPRF Preprocessing

[ChenLaineHaungRindal18]



Not a bug, it's a feature

[ChenLaineHaungRindal18]

$k \leftarrow \mathbb{K}$

$\hat{X} := \{(\hat{x}, H(\hat{x})) \mid x \in X, \hat{x} = F_k(x)\}$

k → OPRF ← y

$\hat{y} := F_k(y)$

$\hat{X}$ → KW-PIR ← $\hat{y} \in \hat{Y}$

$z = H(\hat{y}) \Leftrightarrow y \in X \cap Y$

Not a bug, it's a feature

[ChenLaineHaungRindal18]

$k \leftarrow \mathbb{K}$

$\hat{X} := \{(\hat{x}, H(\hat{x})) \mid x \in X, \hat{x} = F_k(x)\}$

k → OPRF ← y

$\hat{y} := F_k(y)$

$\hat{X}$ ← $\{[\![\hat{y}^i]\!] \mid i = 1, \dots, \log N\}$ — $\hat{y} \in \hat{Y}$

$[\![z]\!]$ →

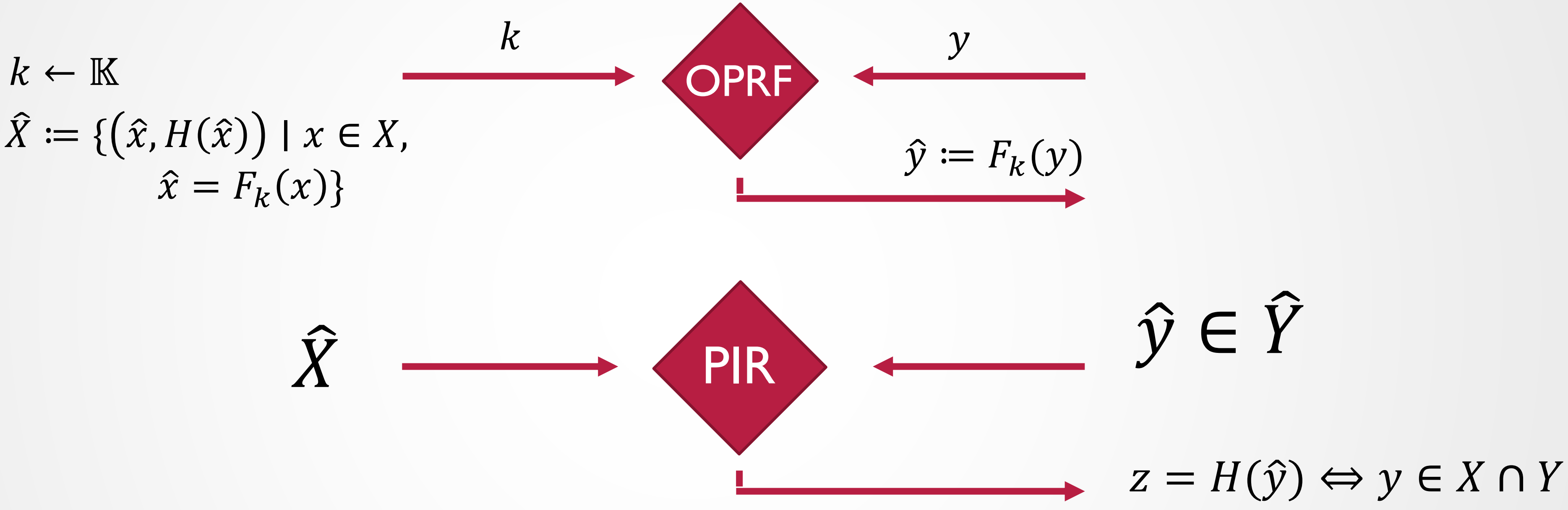
- Hard to compute

  $\{[\![H(\hat{y})]\!]\} = H'(\{[\![\hat{y}^i]\!] \mid i\})$

$z = H(\hat{y}) \Leftrightarrow y \in X \cap Y$

OPRF Preprocessing

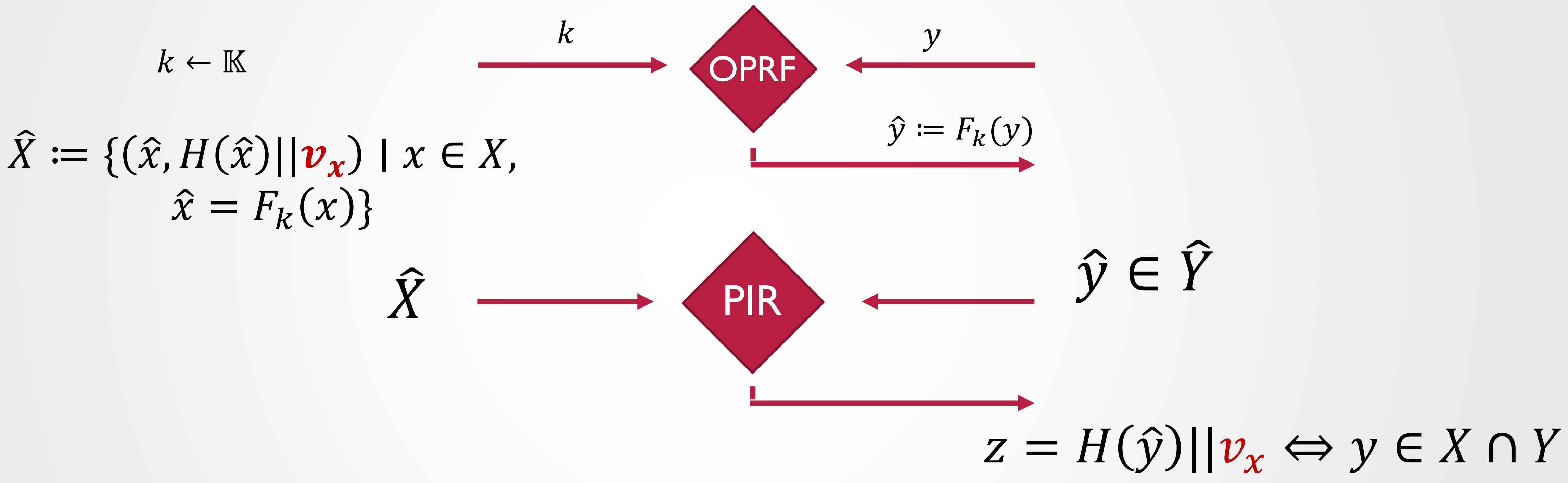
[ChenLaineHaungRindal18]



- Advantages:
  - Malicious security
  - No circuit privacy → Much smaller FHE parameters
  - “Simpler” Design
  - Reusable OPRF
- Disadvantages:
  - Requires OPRF subprotocol

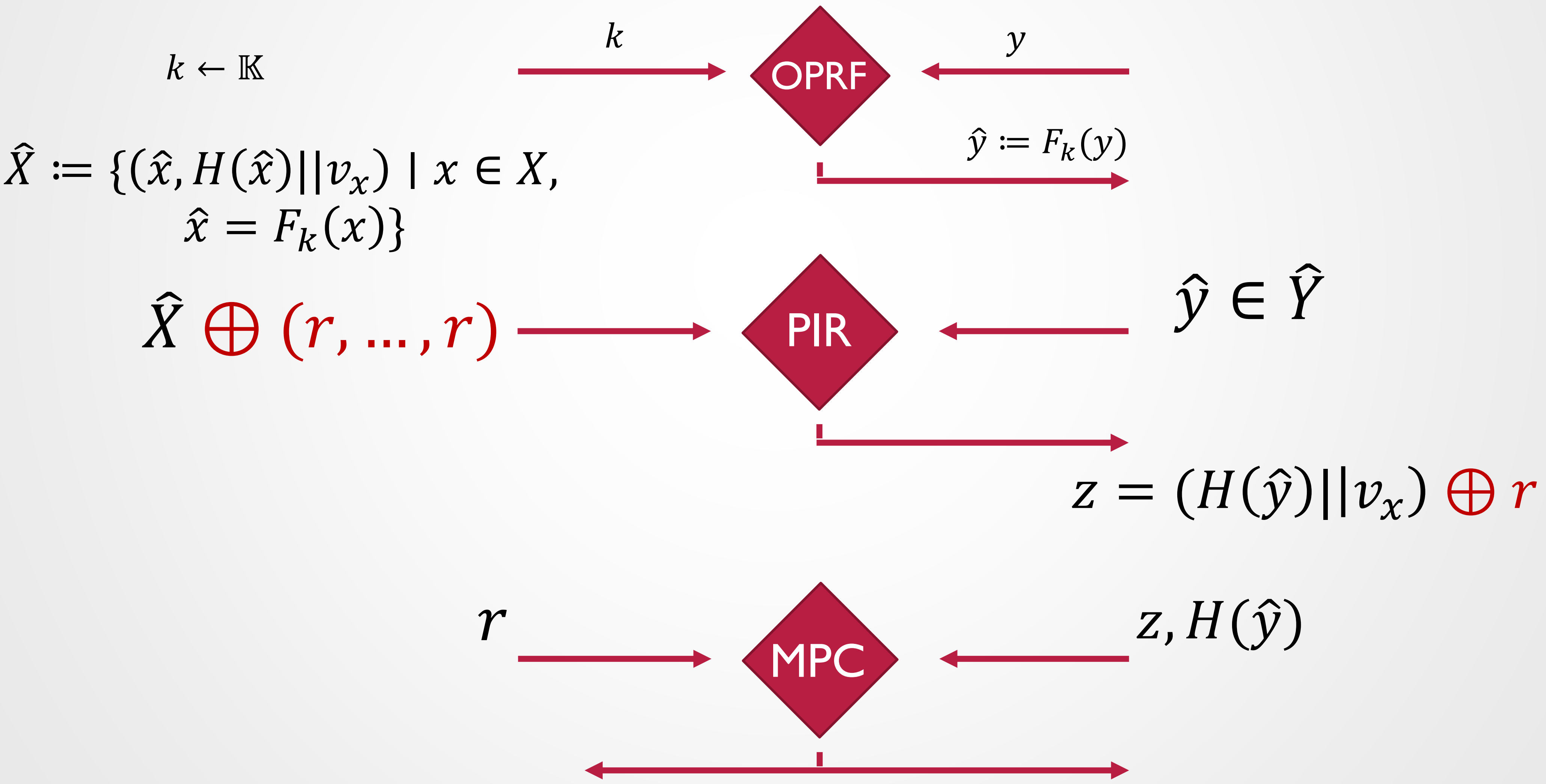
Labeled PSI

[ChenLaineHaungRindal18]



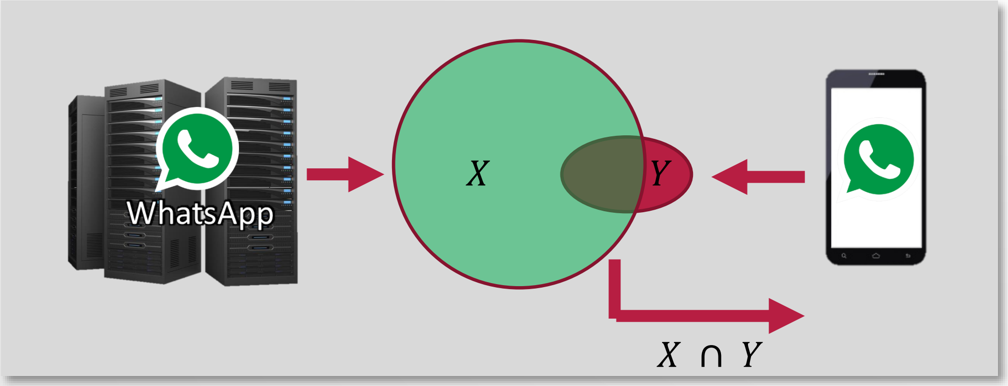
PSI with Shared Output

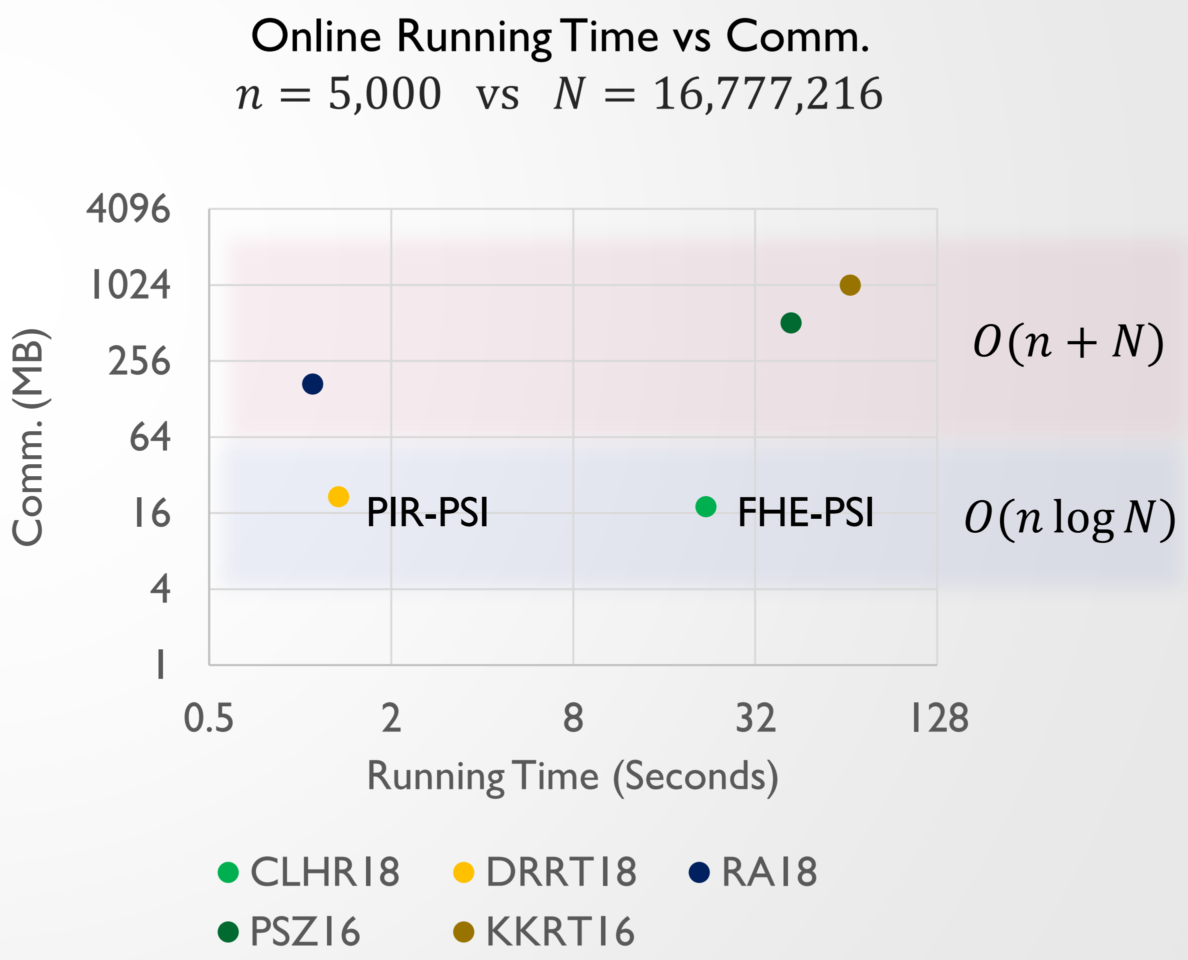
[ChenLaineHaungRindal18]



Performance

- FHE-PSI,
  - 20 seconds and **18MB**
  - Requires a single server
  - Malicious secure*
  - Has a large offline computation
- All other protocols require linear communication
  - RA18 has better running time.





The End

VISA Research | **Peter Rindal**

Microsoft Research | Hao Chen
Kim Laine

可用不可见 BLINDFOLDED COMPUTING | Zhicong Huang