FHE for RAM computation from RingLWE

Wei-Kai Lin
Northeastern

Ethan Mook
Northeastern

Daniel Wichs
Northeastern &
NTT Research

Motivating Example: Google Search

Google

eye twitch covid symptom?

https://www.healthline.com › health › eye-health › is-ey...

Is Eye Twitching a Sign of COVID-19? - Healthline

Jun 21, 2022 — Eye twitching can be a symptom of COVID-19, but **it can also be caused by stress and eyestrain**. Here are some other probable causes.

COVID-19 symptom · Causes · Takeaway · Clinical trials

https://www.ncbi.nlm.nih.gov › articles › PMC8743236

Eyelid Myokymia—a Presumed Manifestation of Coronavirus ...

by HA Khan · 2022 — The course and pattern of eyelid twitching were studied over 3 ... Common ocular complications/manifestations of COVID-19 include **dry eye**, ...

Introduction · Methods · Results · Discussion

https://timesofindia.indiatimes.com › photostory

Eye twitching and other eye symptoms associated with COVID ...

Jul 2, 2022 — Apart from spasms, you may notice other eye symptoms with a COVID-19 infection. These include, **dry eye, itchiness**, redness, conjunctivitis (pink ...

https://www.allaboutvision.com › coronavirus › eye-pr...

Eye Problems that Could be Related to COVID - All About Vision

Feb 17, 2021 — **Eye twitching was not identified as an ocular symptom of COVID-19** in a meta-analysis of 12 studies on COVID-19 eye symptoms.

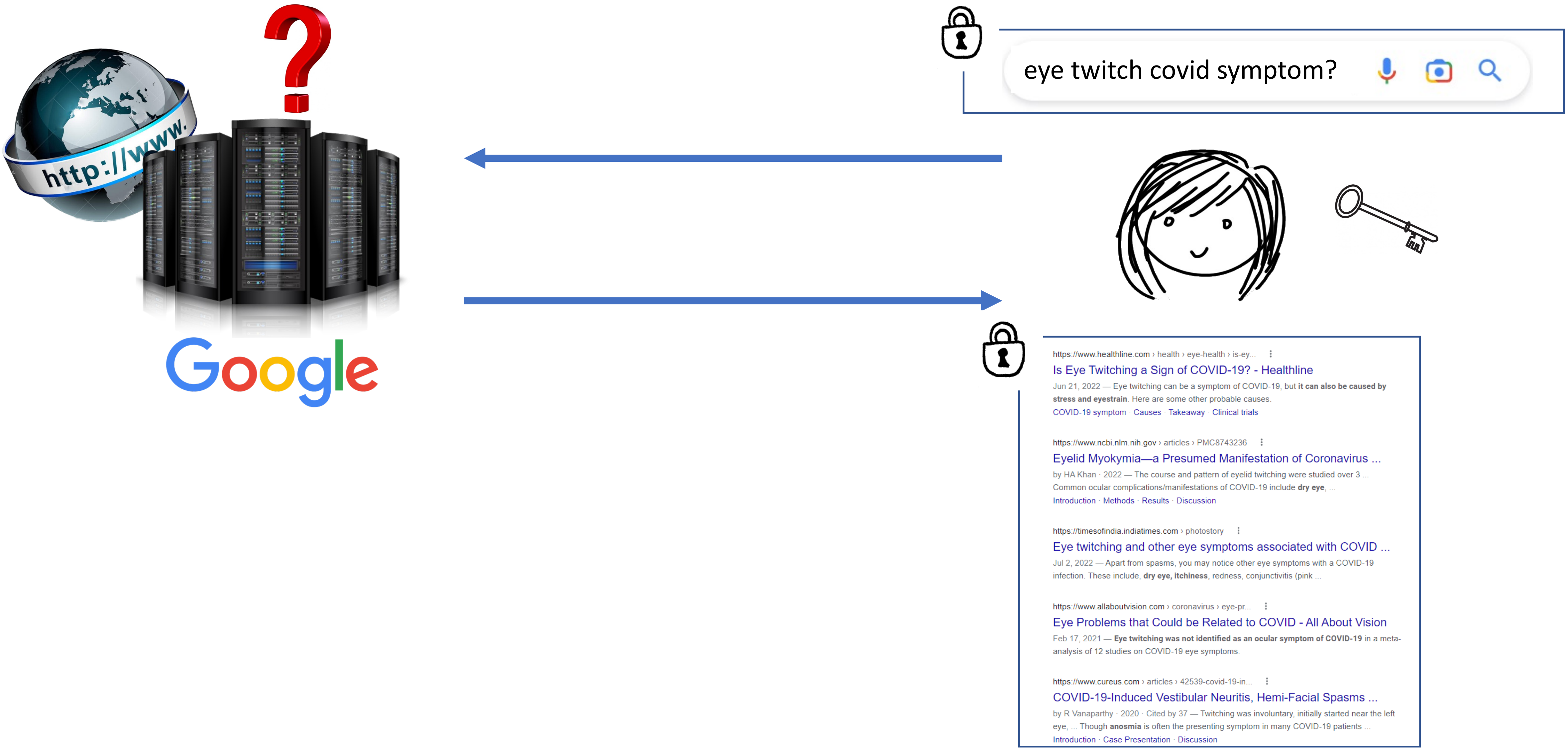
https://www.cureus.com › articles › 42539-covid-19-in...

COVID-19-Induced Vestibular Neuritis, Hemi-Facial Spasms ...

by R Vanaparthy · 2020 · Cited by 37 — Twitching was involuntary, initially started near the left eye, ... Though **anosmia** is often the presenting symptom in many COVID-19 patients ...

Introduction · Case Presentation · Discussion

Private Google Search

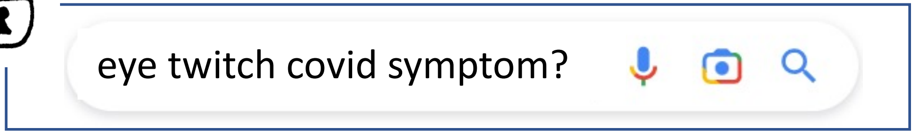


Private Google Search

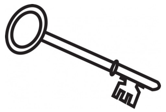
Google

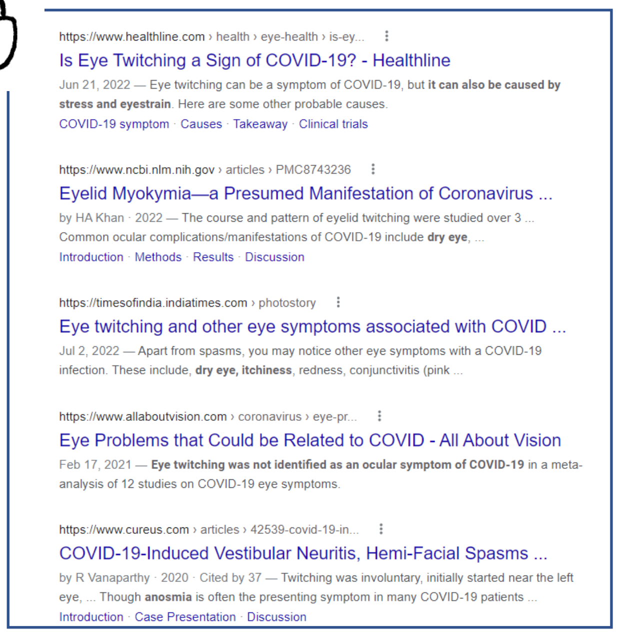
Eval(google_search, ct)

$ct = Enc_{sk}(query)$



$Enc_{sk}(response)$





https://www.healthline.com › health › eye-health › is-ey...

Is Eye Twitching a Sign of COVID-19? - Healthline

Jun 21, 2022 — Eye twitching can be a symptom of COVID-19, but **it can also be caused by stress and eyestrain**. Here are some other probable causes.

COVID-19 symptom · Causes · Takeaway · Clinical trials

https://www.ncbi.nlm.nih.gov › articles › PMC8743236

Eyelid Myokymia—a Presumed Manifestation of Coronavirus ...

by HA Khan · 2022 — The course and pattern of eyelid twitching were studied over 3 ... Common ocular complications/manifestations of COVID-19 include **dry eye**, ...

Introduction · Methods · Results · Discussion

https://timesofindia.indiatimes.com › photostory

Eye twitching and other eye symptoms associated with COVID ...

Jul 2, 2022 — Apart from spasms, you may notice other eye symptoms with a COVID-19 infection. These include, **dry eye, itchiness**, redness, conjunctivitis (pink ...

https://www.allaboutvision.com › coronavirus › eye-pr...

Eye Problems that Could be Related to COVID - All About Vision

Feb 17, 2021 — **Eye twitching was not identified as an ocular symptom of COVID-19** in a meta-analysis of 12 studies on COVID-19 eye symptoms.

https://www.cureus.com › articles › 42539-covid-19-in...

COVID-19-Induced Vestibular Neuritis, Hemi-Facial Spasms ...

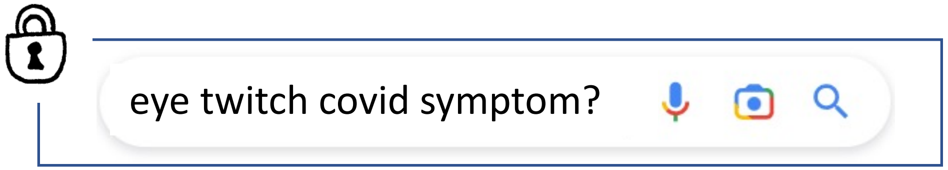
by R Vanaparthy · 2020 · Cited by 37 — Twitching was involuntary, initially started near the left eye, ... Though **anosmia** is often the presenting symptom in many COVID-19 patients ...

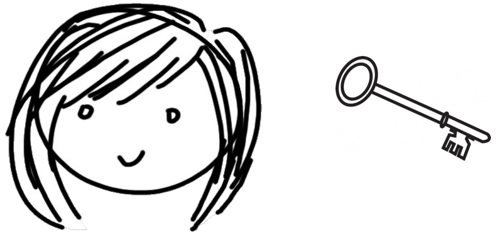
Introduction · Case Presentation · Discussion

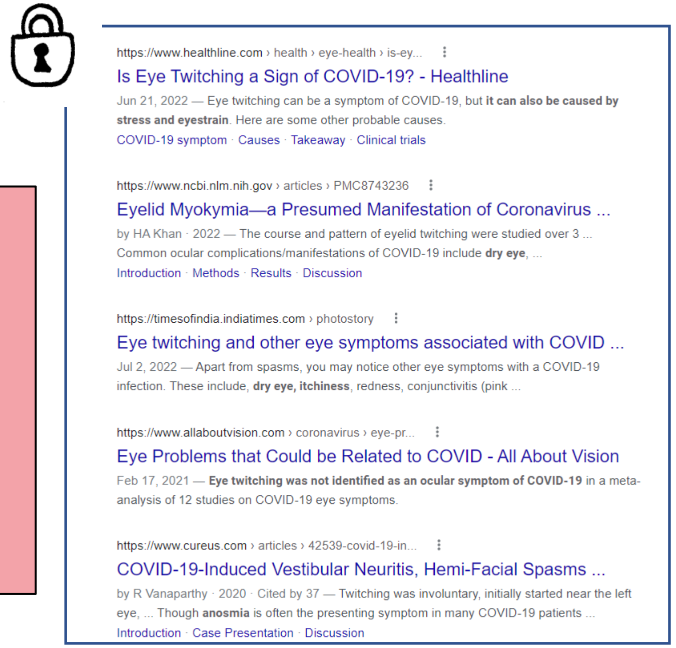
Fully Homomorphic Encryption (FHE)

- Privately evaluate arbitrary functions

Private Google Search



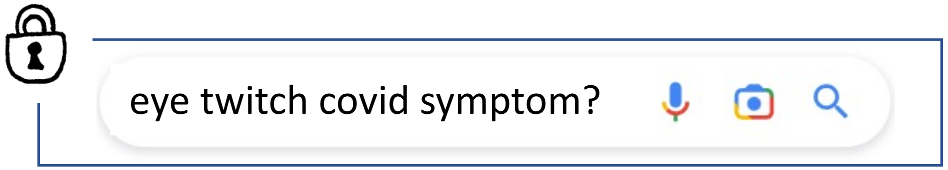


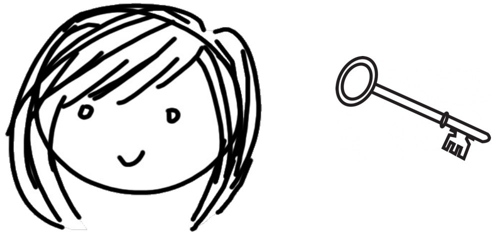


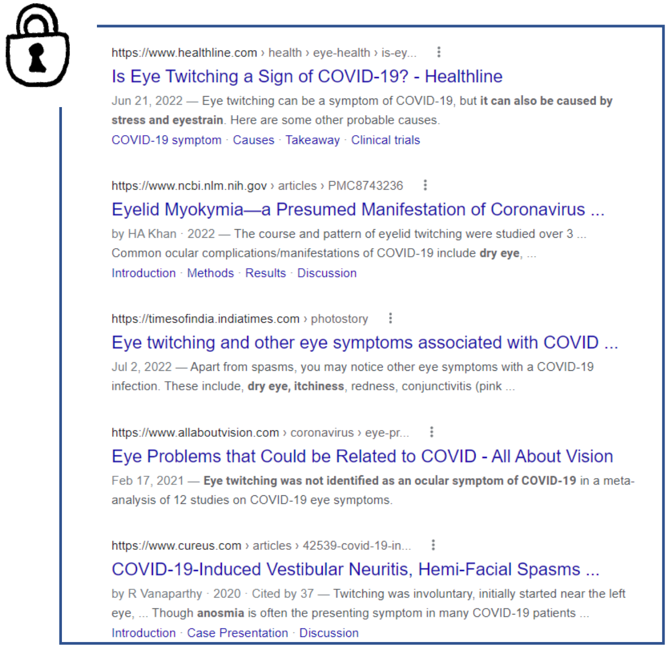
Major Caveat: FHE operates in the circuit model - can't make efficient memory access while preserving security

⇒ Google needs to read the entire content of the internet to answer each encrypted query!

Private Google Search







Result: We build FHE in the RAM model

- Google preprocesses the Internet content into a specialized *data structure*
- Can answer any future encrypted query efficiently by only accessing a few locations!

Fully Homomorphic Encryption (FHE)

[Rivest-Adleman-Dertouzos’78,Gentry ‘09,Brakerski-Vaikuntanathan11,…]

y

Circuit C for $f(\cdot, y)$

$ct_x \leftarrow Enc_{sk}(x)$

$ct^* = Eval(C, ct_x)$

x

$sk \leftarrow \$$

$Dec_{sk}(ct^*) = C(x) = f(x, y)$

☺ Server learns nothing about x

☺ Client time/communication $O(|x| + |f(x, y)|)$

☹ Server eval time is at least $|C| > |y|$

RAM-FHE

Preprocess

$y \rightarrow \tilde{y}$

$P(\cdot,\cdot)$: RAM program for f with worst-case run-time T.

$ct_x \leftarrow Enc_{sk}(x)$

$ct^* = Eval(P, ct_x, \tilde{y})$

x

$sk \leftarrow \$$

$Dec_{sk}(ct^*) = P(x, y)$

☺ Server learns nothing about x

☺ Client time/communication $O(|x| + |f(x, y)|)$ (nearly)

☺ Server eval time is nearly $O(T)$

Server preprocessing time is nearly $O(|y|)$

RAM-FHE

Preprocess

$y \rightarrow \tilde{y}$

$P(\cdot,\cdot)$: RAM program for f with worst-case run-time T.

$ct_x \leftarrow Enc_{sk}(x)$

$ct^* = Eval(P, ct_x, \tilde{y})$

x

$sk \leftarrow \$$

$Dec_{sk}(ct^*) = P(x, y)$

P gets efficient access to **both** x and y

Use cases over Circuit-FHE:

- Private query to large public database
- Outsource computation on large private database
- Avoid blowup converting RAM program to circuit

Google search

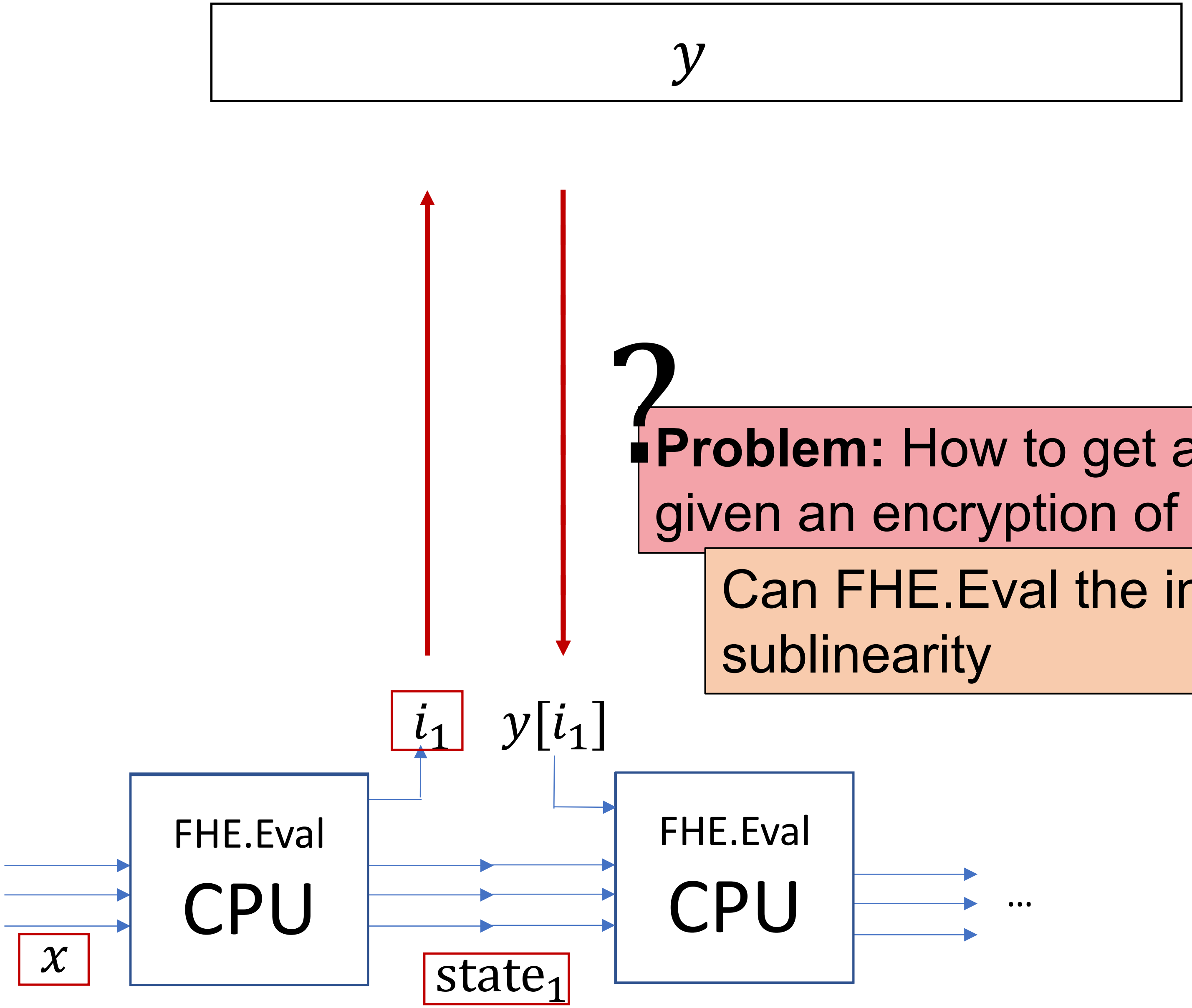
RAM-FHE: Prior work and Our Result

Prior Work: [Holmgren-Hamlin-Weiss-Wichs ’19] build a weaker variant of RAM-FHE based on heuristic use of obfuscation

Result: We build RAM-FHE based on the Ring Learning with Errors (RingLWE) assumption (+ circular security)

- RingLWE is a well studied assumption
  - As hard as finding approximate shortest vector in ideal lattices in worst case.
  - Basis of new NIST standard for next generation public-key encryption.
- Alternate constructions: *approximate GCD, NTRU, O(1)-Rank Module LWE*

Main Challenge: Allow efficient database access under FHE without revealing the access pattern



Problem: How to get an encryption of $y[i_1]$ given an encryption of i_1?

Can FHE.Eval the indexing circuit, but lose sublinearity

Private Information Retrieval (PIR)

[CGKS95,KO00]

$\text{DB} \in \{0,1\}^N$

$i \in [N]$

Goal: Retrieve $\text{DB}[i]$ without revealing i.

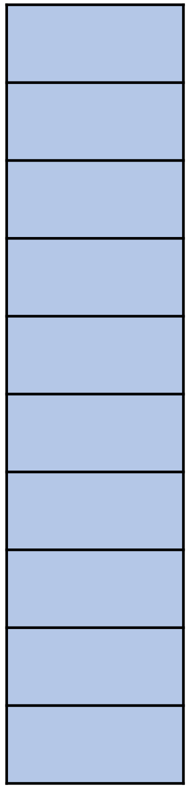
Trivial solution: server sends entire DB.

Can think of as special case of FHE
⇒ FHE yields polylog ***communication***

Private Information Retrieval (PIR)

[CGKS95,KO00]

$\mathrm{DB} \in \{0,1\}^N$



$i \in [N]$

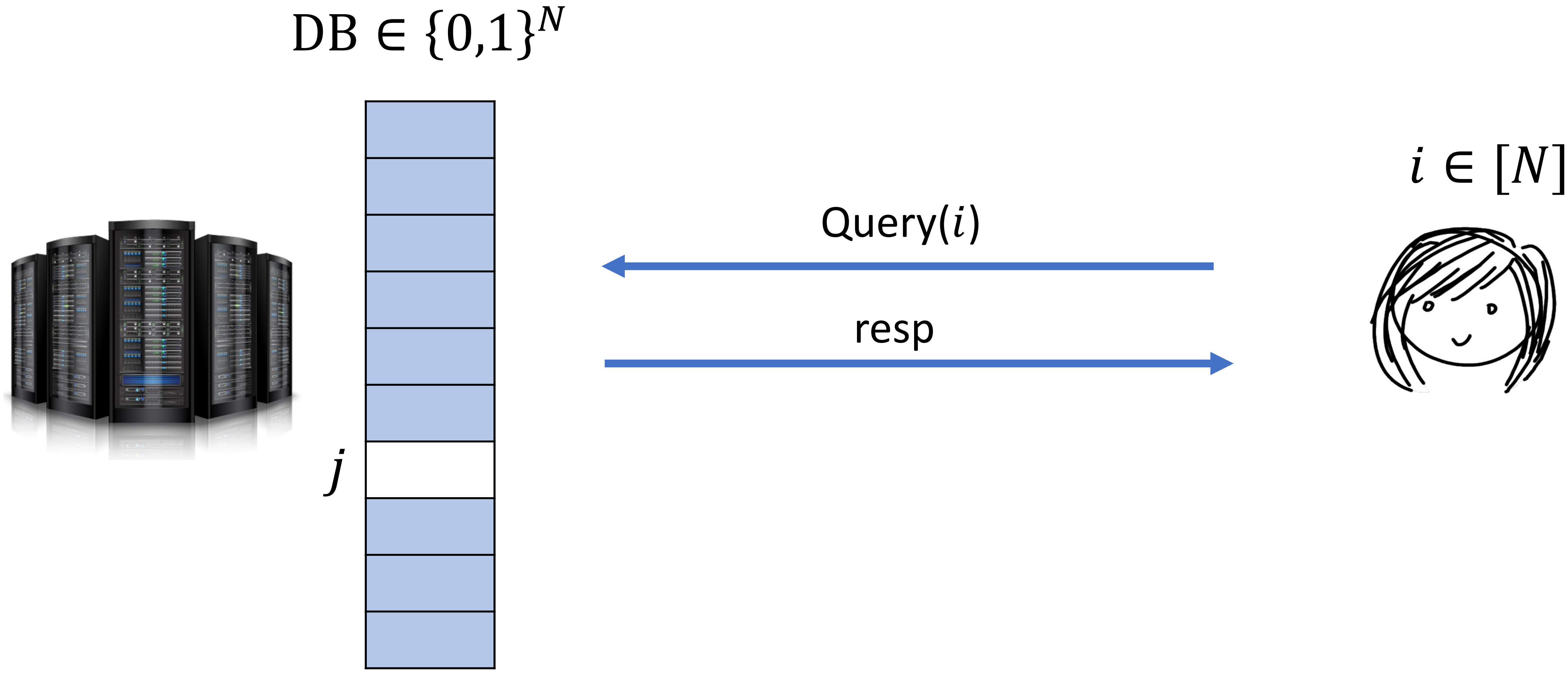
Goal: Retrieve $\mathrm{DB}[i]$ without revealing i.

Caveat: Server reads entire DB during protocol

⇒ <u>Server computation</u> is $\geq N$.

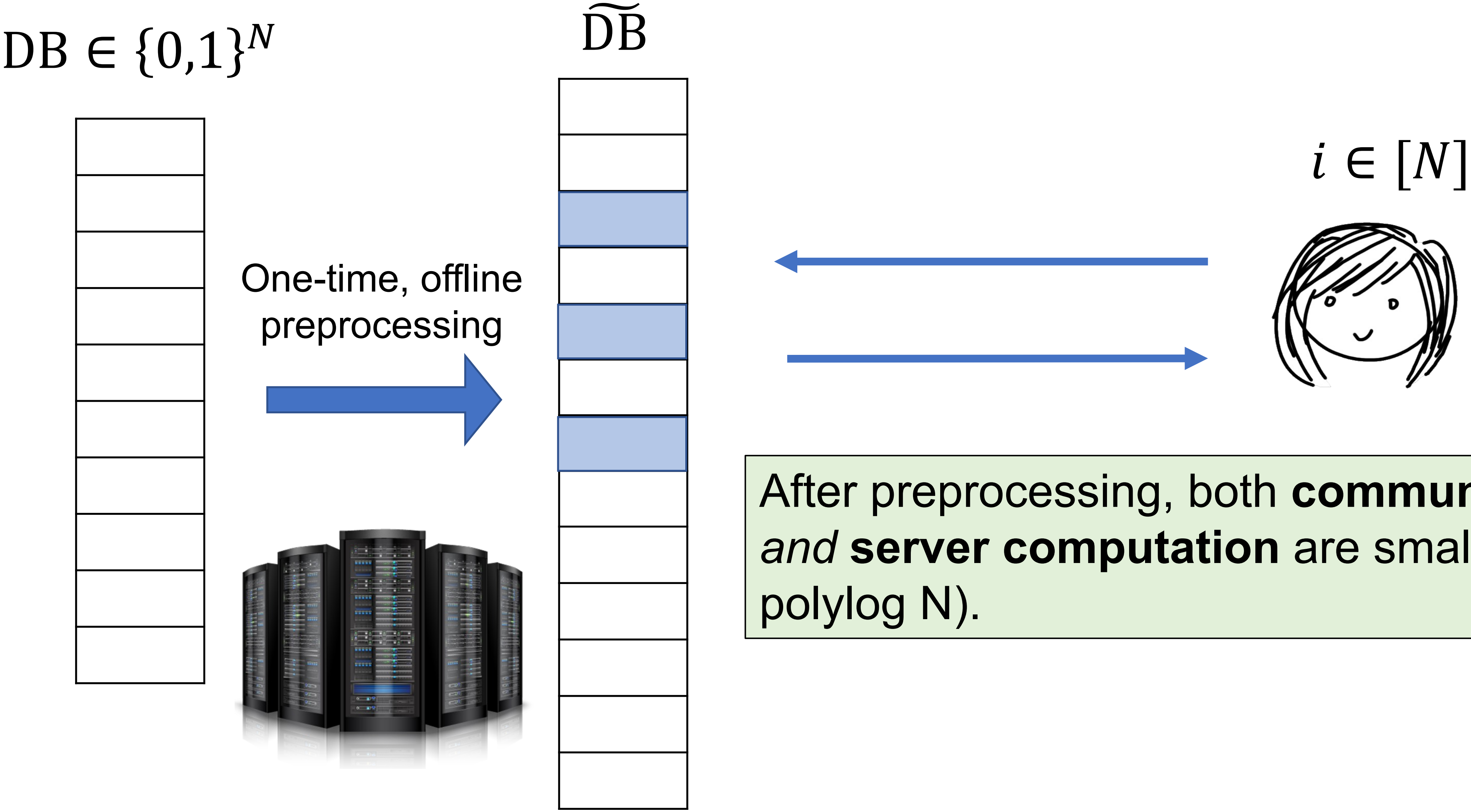
This is **inherent** if the server only stores DB.

PIR Lower Bound



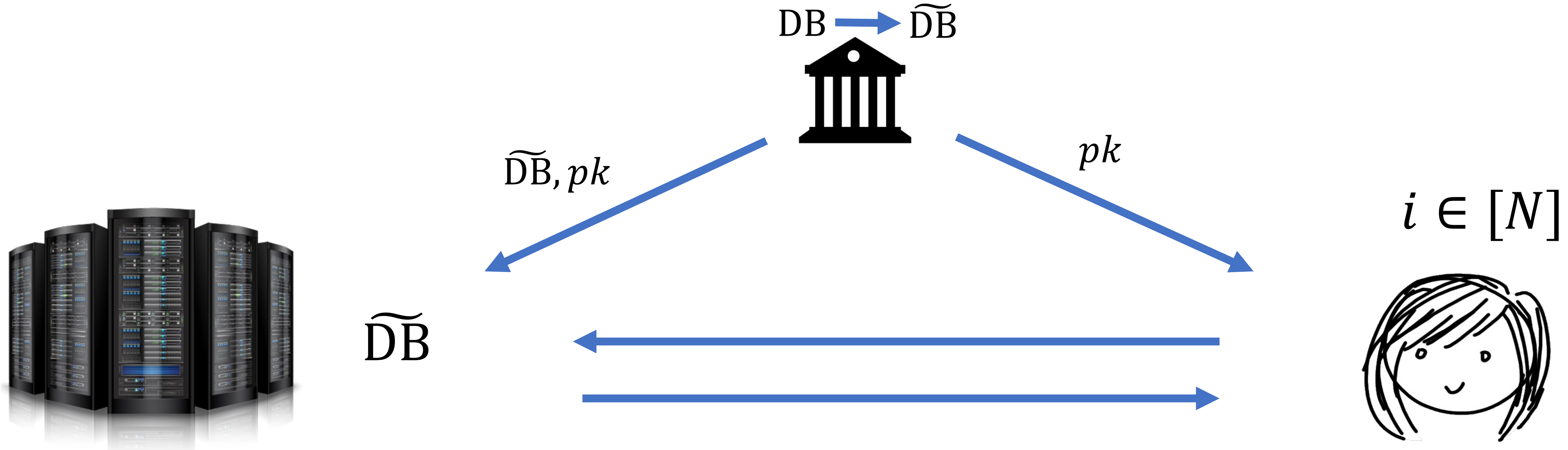
Server learns $i \neq j$!

Doubly Efficient PIR (DEPIR)



After preprocessing, both **communication** *and* **server computation** are small (ideally polylog N).

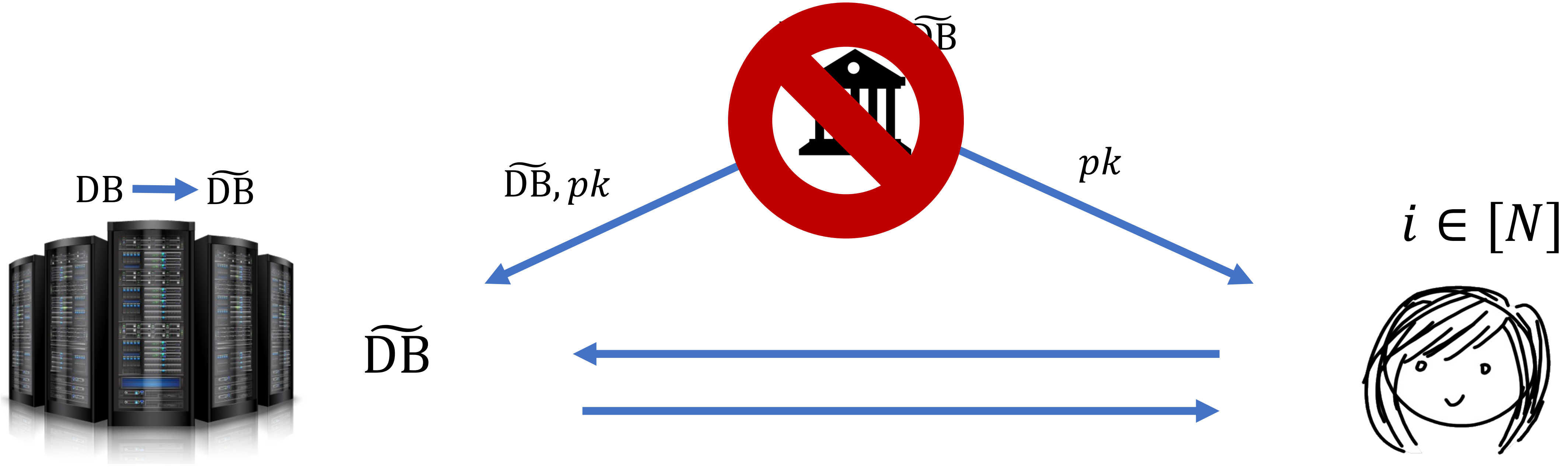
Prior Work on DEPIR



Prior Work:

- Originally proposed by [Beimel-Ishai-Malkin ‘00]
- First evidence from [Canetti-Holmgren-Richelson ’17] and [Boyle-Ishai-Pass-Wootters ’17]: give constructions of ***keyed*** DEPIR that rely on a new non-standard assumption and heuristic use of obfuscation

Our Results on DEPIR



Result: We construct ***unkeyed*** DEPIR from the RingLWE assumption

- Server deterministically computes preprocessing on its own
- Later any client can query DB in a 2-Round Protocol

Our Results on DEPIR

Result: We construct ***unkeyed*** DEPIR from the RingLWE assumption

- Server deterministically computes preprocessing on its own
- Later any client can query DB in a 2-Round Protocol

Efficiency: For any $\epsilon > 0$, database size N:

- Preprocessing run-time/size: $O(N^{1+\epsilon})$
- PIR protocol run-time/communication: $\mathrm{polylog}\, N$
- Also: ***Updatable*** DEPIR – update $\widetilde{DB}$ in time: $O(N^{\epsilon})$

Alternatively:

$\rightarrow\ N \cdot 2^{O(\sqrt{\log N})} = N^{1+o(1)}$

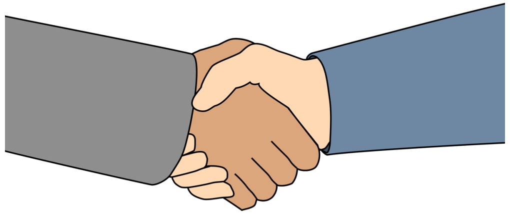
$\rightarrow\ 2^{O(\sqrt{\log N})} = N^{o(1)}$

$\rightarrow\ 2^{O(\sqrt{\log N})} = N^{o(1)}$

DEPIR Template

Simple PIR from Homomorphic Encryption

Cryptography



Algorithms

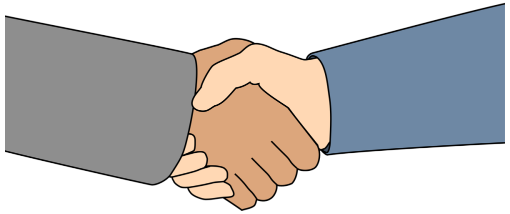
Preprocessing polynomial evaluation
[Kedlaya-Umans'08]

DEPIR Template

Can only evaluate **low degree** functions

Simple PIR from Somewhat Homomorphic Encryption (SHE)

Cryptography



Algorithms

Preprocessing polynomial evaluation [Kedlaya-Umans'08]

Basic PIR from SHE

Write i in base d (prime),
note $m = \log_d N$

$\mathrm{DB} \in \{0,1\}^N$

$i \in [N]$

$i = (i_1, i_2, \ldots, i_m) \in \mathbb{Z}_d^m$

$\forall j\text{: } \alpha_j \leftarrow Enc_{sk}(i_j)$

$\beta = Eval(f_{DB}, \alpha_1, \ldots, \alpha_m)$

$sk \leftarrow \$$

$f_{DB} \in \mathbb{Z}_d[X_1, \ldots, X_m]$

$DB[i] = Dec_{sk}(\beta)$

$f_{DB}(i_1, \ldots, i_m) = \mathrm{DB}[i]$

f_{DB} has individual degree $< d$ and total degree at most $D = dm = d \cdot \log_d N$

Preprocessing Polynomials

[Kedlaya-Umans '08]

$N = d^m = $ # coeff's in f

Lemma: Given polynomial $f(X_1, \ldots, X_m)$ over the ring $R = \mathbb{Z}_q$ with individual degree $< d$ into can preprocess f into a data structure such that:

- Can evaluate $f(\boldsymbol{\alpha})$ for any $\boldsymbol{\alpha} \in \mathbb{Z}_q^m$ in time $\mathrm{poly}(d, m, \log|R|)$
- Preprocessing time/space: $N \cdot O(m \log N)^m \cdot \mathrm{poly}(d, m, \log|R|)$

$\rightarrow \mathrm{polylog}(N)$

$\rightarrow N^{1+\epsilon}$

Recall: We want d small for the SHE scheme

Choose parameters:

- $d = \log^c N$
- $m = \log_d N = \frac{\log N}{c \cdot \log \log N}$
- $|R| = 2^{\mathrm{polylog}(N)}$

Aside: [KU'08] extends to polys over a larger class of rings including
$R = \mathbb{Z}_q[Y, Z]/(E_1(Y), E_2(Z))$

Apply [KU08] to Basic PIR?

$\mathrm{DB} \in \{0,1\}^N$

$f_{DB} \in \mathbb{Z}_d[X_1, \dots, X_m]$

$f_{DB}(i_1, \dots, i_m) = \mathrm{DB}[i]$

Write i in base d (prime)

$i = (i_1, i_2, \dots, i_m) \in \mathbb{Z}_d$

$\forall j\colon\ \alpha_j \leftarrow Enc_{sk}(i_j)$

$\beta = Eval(f_{DB}, \alpha_1, \dots, \alpha_m)$

$sk \leftarrow \$$

$DB[i] = Dec_{sk}(\beta)$

Problem: Server doesn't directly compute f_{DB} but instead SHE $Eval$
⇒ can't preprocess server computation

Algebraic Somewhat Homomorphic Encryption (ASHE)

Plaintext space

$\mathbb{Z}_d$

Ciphertext space

A ring R

Messages $\mu_1, \mu_2 \in \mathbb{Z}_d$ $\xrightarrow{Enc}$ Ciphertexts $\alpha_1, \alpha_2 \in R$

$$Dec(\alpha_1 + \alpha_2) = \mu_1 + \mu_2$$
$$Dec(\alpha_1 \cdot \alpha_2) = \mu_1 \cdot \mu_2$$

Ring operations in R –
No FHE $Eval$ necessary

Algebraic Somewhat Homomorphic Encryption (ASHE)

Plaintext space

$\mathbb{Z}_d$

Ciphertext space

A ring R

Messages $\mu_1, \mu_2 \in \mathbb{Z}_d$ $\xrightarrow{Enc}$ Ciphertexts $\alpha_1, \alpha_2 \in R$

- Correspondence extends to polynomial evaluation:

  If $\alpha_1 \leftarrow Enc_{sk}(\mu_1), \ldots, \alpha_m \leftarrow Enc_{sk}(\mu_m)$ and f is a poly over $\mathbb{Z}_d$ of total degree $< D$,
  then $f(\alpha_1, \ldots, \alpha_m) = Enc_{sk}(f(\mu_1, \ldots, \mu_m))$ where f is "lifted" to R.

- Complexity (bit-size of ring elements, encryption/decryption time) can be $\mathrm{poly}(D)$.

Algebraic Somewhat Homomorphic Encryption (ASHE)

Plaintext space

$\mathbb{Z}_d$

Ciphertext space

A ring R

Messages $\mu_1, \mu_2 \in \mathbb{Z}_d$ $\xrightarrow{Enc}$ Ciphertexts $\alpha_1, \alpha_2 \in R$

- Get ASHE from minor modifications of prior SHE schemes
  - From [BV11] based on RingLWE → Main construction
  - From [LTV12] based on security of NTRU
  - From [vGHV10] based on Approximate GCD

DEPIR Construction

Write i in base d (prime)

$\text{DB} \in \{0,1\}^N$

$\forall j\colon \alpha_j \leftarrow Enc_{sk}(i_j)$

$i = (i_1, i_2, \dots, i_m) \in \mathbb{Z}_d$

$sk \leftarrow \$$

$\beta = Eval(f_{DB}, \alpha_1, \dots, \alpha_m)$

$\beta = f_{DB}(\alpha_1, \dots, \alpha_m)$

$f_{DB} \in \mathbb{Z}_d[X_1, \dots, X_m]$

Lift f_{DB} to

$f_{DB} \in R[X_1, \dots, X_m]$

$\text{DB}[i] = Dec_{sk}(\beta)$

Preprocess with [KU08]

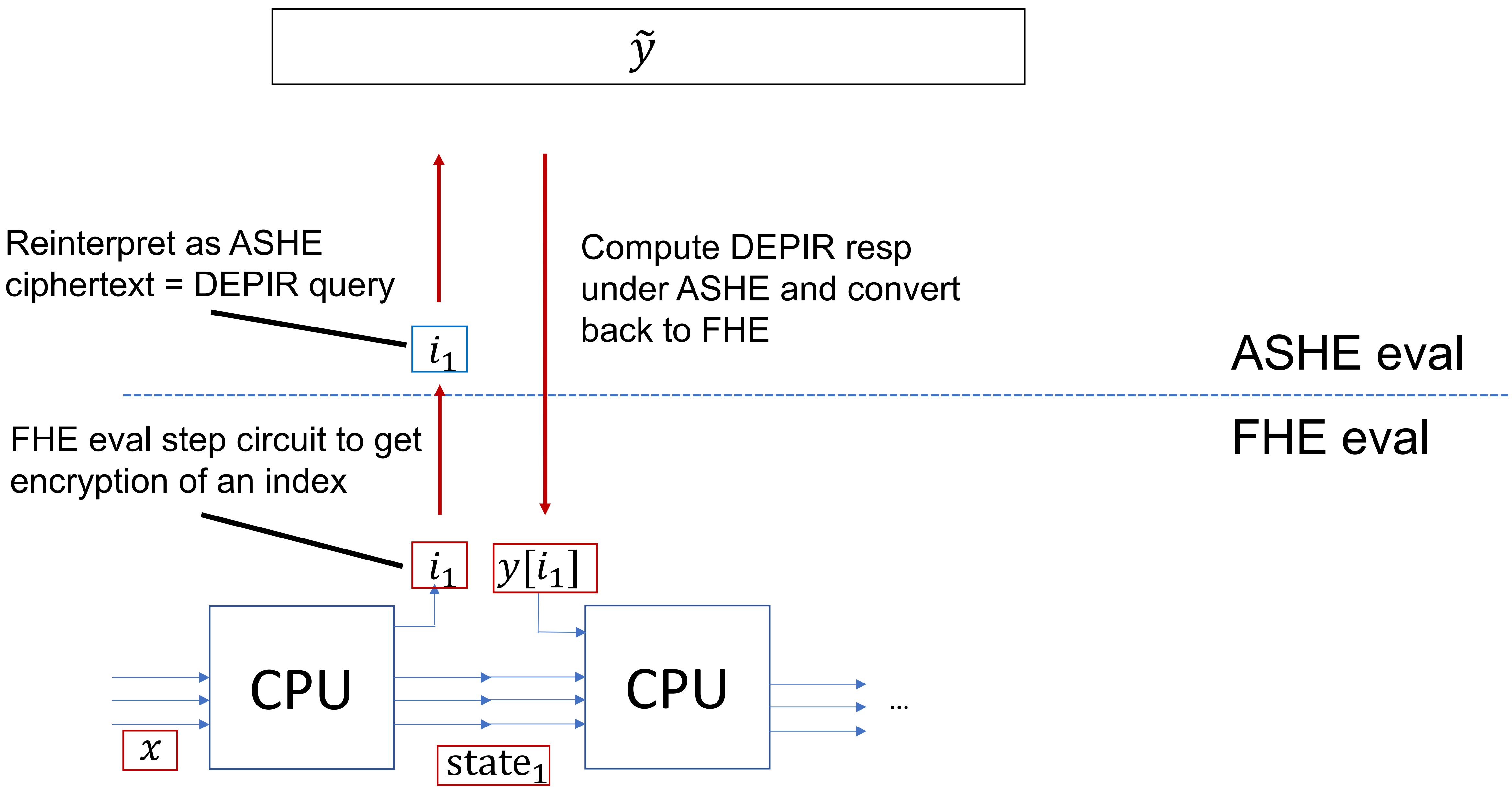
From DEPIR to RAM-FHE

Result: We use techniques from our DEPIR construction + (circuit) FHE to build RAM-FHE based on the RingLWE assumption

- We use the ASHE structure of our DEPIR to "glue" it together with a suitable circuit FHE

Efficiency: For any $\epsilon > 0$:

- Preprocessing time: $O(|y|^{1+\epsilon})$
- Client time/communication: $O(|x|^{1+\epsilon} + |f(x,y)|) \cdot \text{polylog}(|x| + |y|)$
- Server time: $O(T^{1+\epsilon}) \cdot \text{polylog}(|x| + |y|)$


$\tilde{y}$
Reinterpret as ASHE ciphertext = DEPIR query
i_1
Compute DEPIR resp under ASHE and convert back to FHE
ASHE eval
FHE eval
FHE eval step circuit to get encryption of an index
i_1
$y[i_1]$
CPU
CPU
…
x
state_1

Conclusions

We construct DEPIR and RAM-FHE from RingLWE.

Applications: In upcoming work, we use DEPIR/RAM-FHE to build RAM versions of:

Laconic Function Evaluation, Functional Encryption, MPC, Obfuscation

Open Questions:

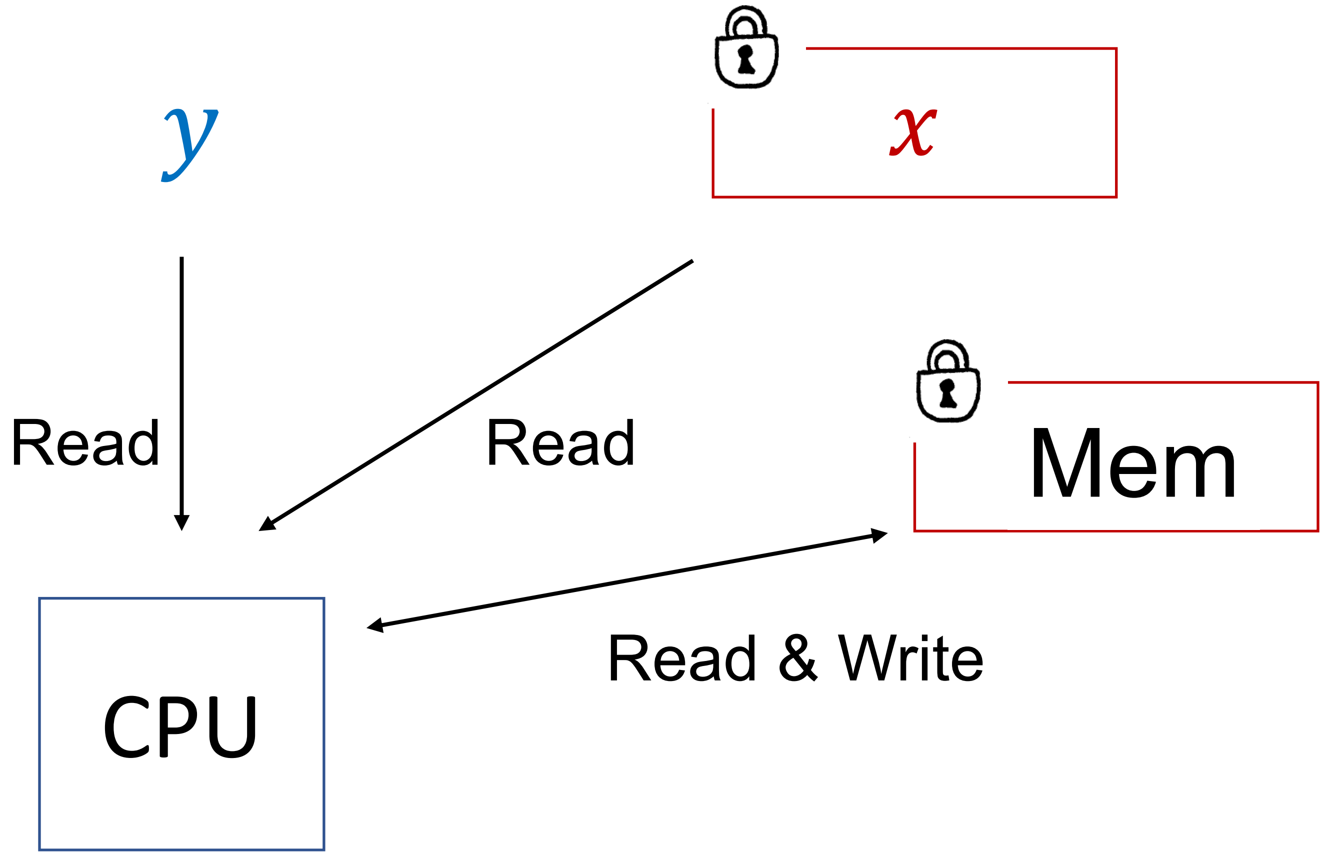
- Can we do it from plain LWE?
- “Practical” efficiency?

Thank you!

RAM Model

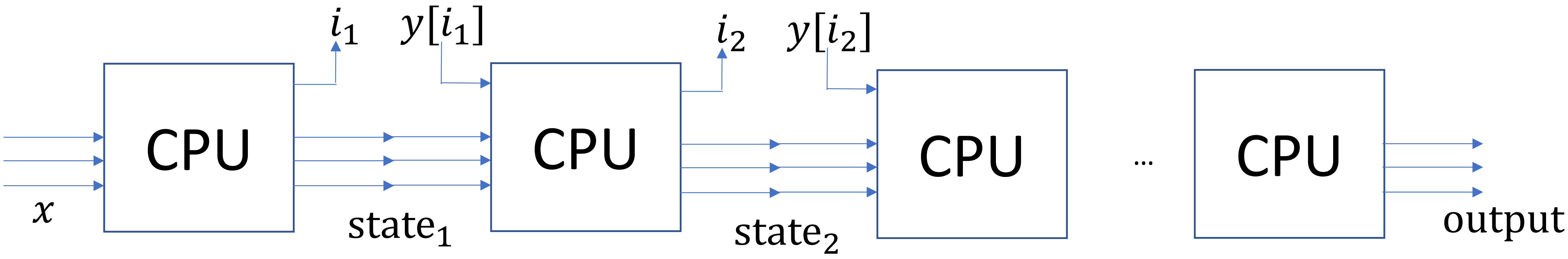
A RAM program P consists of a CPU step circuit with

- Random read access to y
- Random read access to x
- Random read/write access to Mem

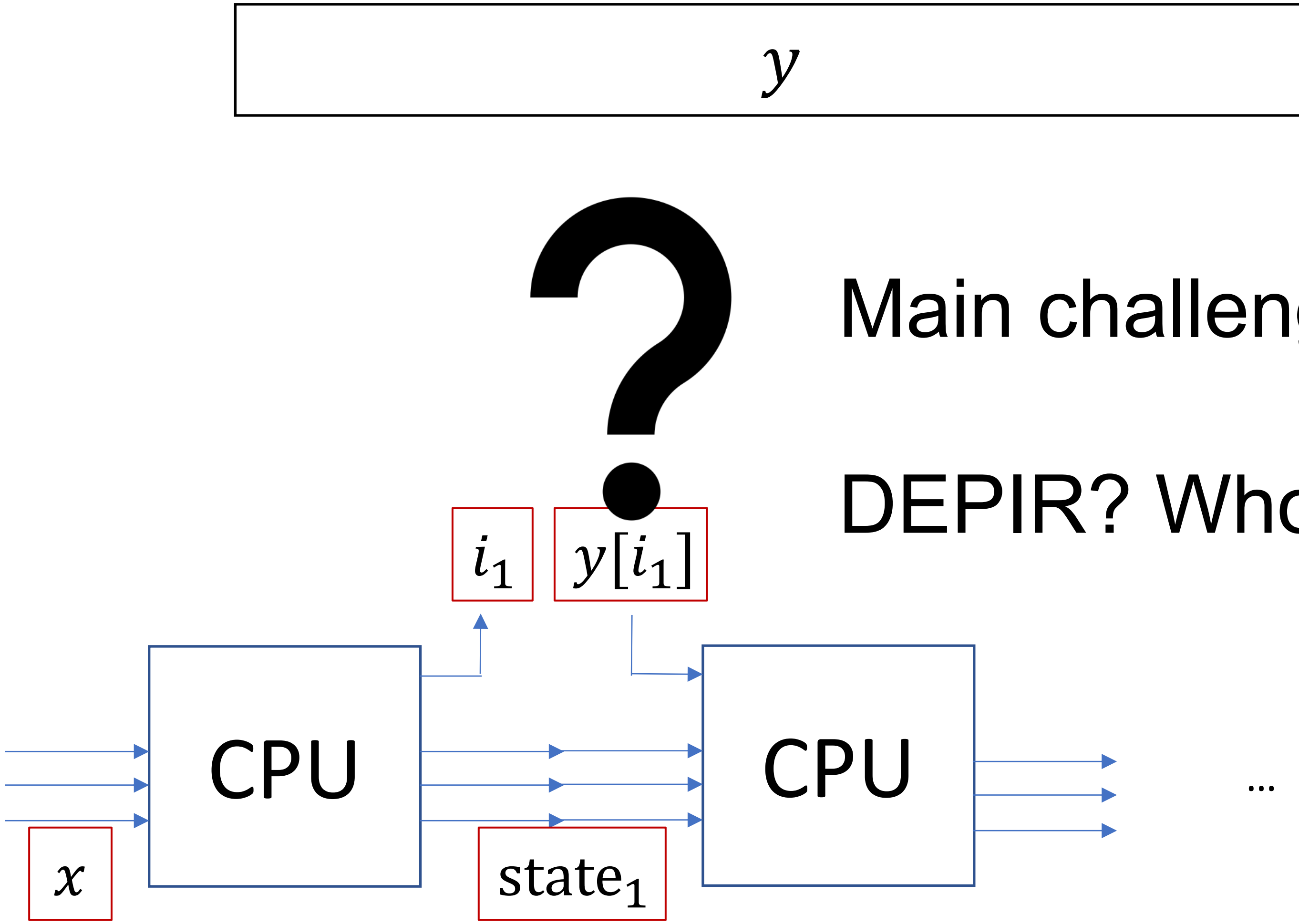


Simpler RAM-FHE

- Simpler Case: RAM program P has
  - read-only random-access to y
  - but no random-access to x or to read/write memory.

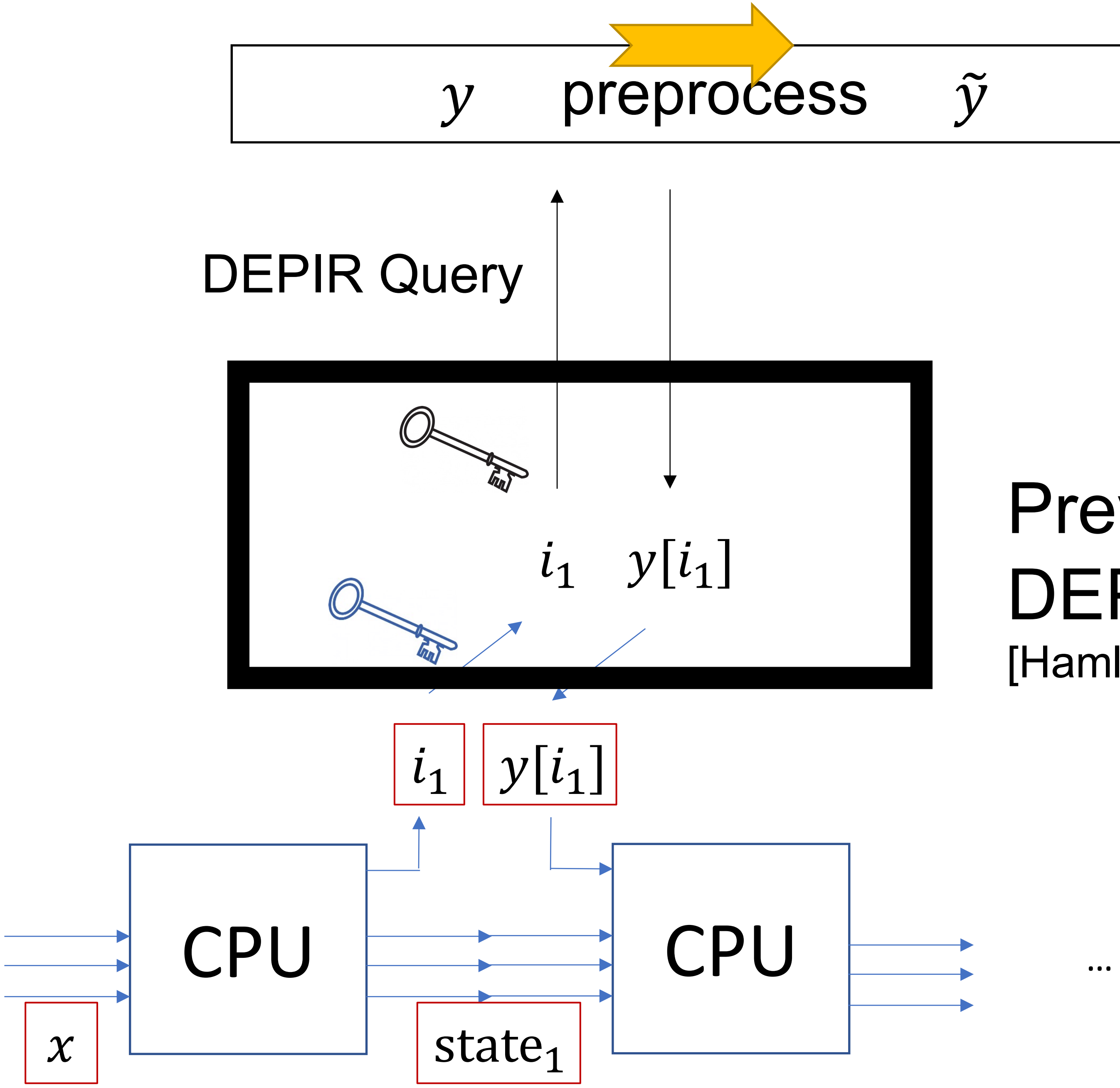


Use Circuit-FHE to compute the step circuit



Main challenge: How to access y?

DEPIR? Who makes the query?



Previous work:
DEPIR + ~~obfuscation~~
[Hamlin-Holmgren-Weiss-Wichs '19]

This work: RingLWE

Key Observation: ASHE-FHE Hybrid

ASHE:
Evaluate a low-degree polynomial on encrypted data

- Simply evaluates the (lifted) polynomial

FHE:
Evaluate any circuit over encrypted data

- Uses non-algebraic operations

switch back-and-forth

Based on RingLWE (or NTRU, ApproxGCD) + circular security

Full RAM-FHE Construction

- Random-access to x can be handled similarly to y.
  - Client first encrypts x and then applies DEPIR preprocessing on it.

- Random-access to read-write memory via updatable DEPIR.
  - Store memory contents encrypted under ASHE-FHE in an updatable DEPIR data structure.