Laconic Function Evaluation, Functional Encryption and Obfuscation for RAMs with Sublinear Computation

Fangqi Dong

IIIS, Tsinghua University

Zihan Hao

IIIS, Tsinghua University

Ethan Mook

Northeastern University

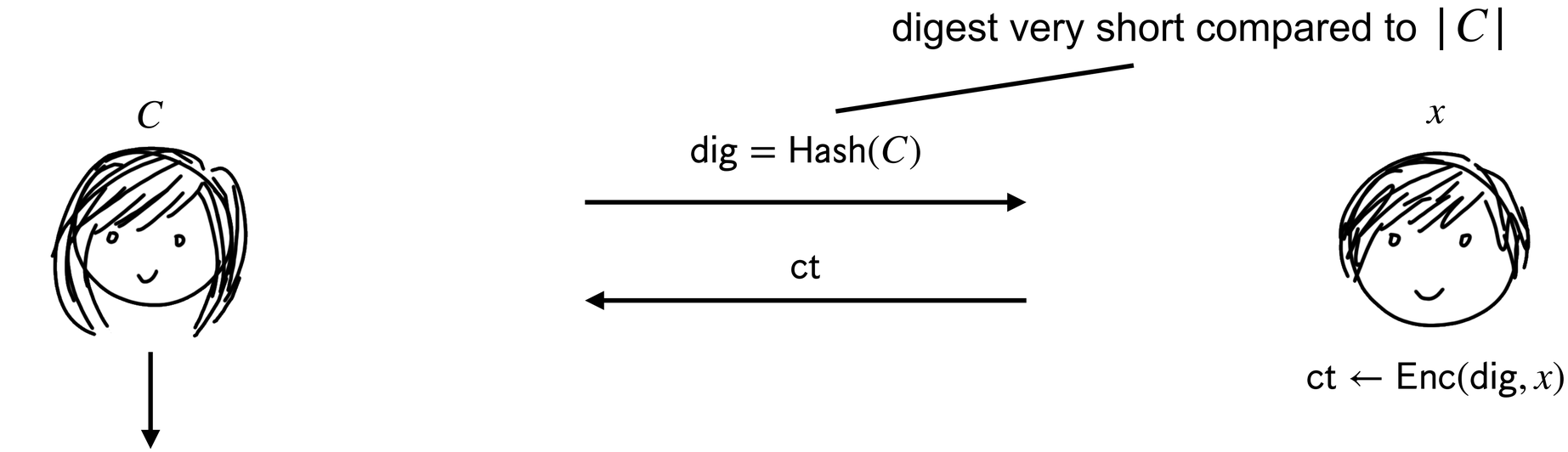
Daniel Wichs

Northeastern University
&
NTT Research

Eurocrypt 2024

Laconic Function Evaluation (LFE)

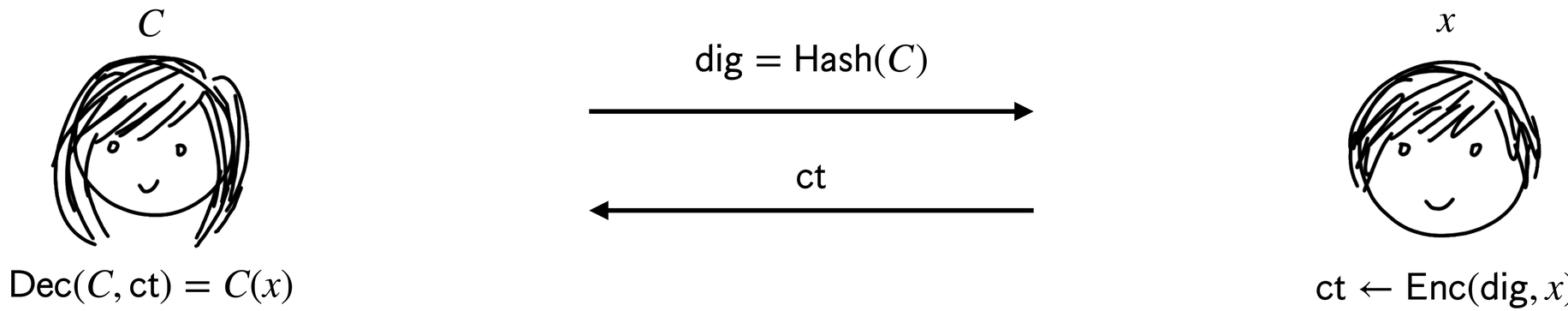
* in CRS model, CRS hidden



Security: Server learns nothing more than $C(x)$

Like FHE: 2-round 2PC where Server does the computational work
But "flipped": Server learns the output (instead of Client)

Laconic Function Evaluation (LFE)



Prior work:
- [Quach-Wee-Wichs'17]: LFE for circuits from LWE
- [Döttling-Gajland-Malavolta'23]: LFE for TMs from iO + SSB

Problem: Server computation is at least linear in inputs!

*suppressing poly(λ) and polylog factors

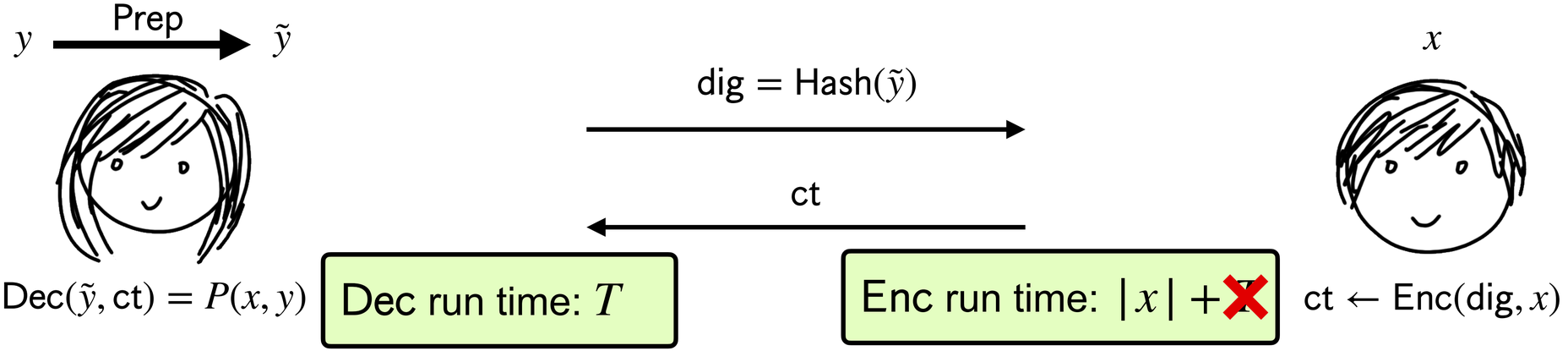
LFE for RAMs

Some fixed RAM program (e.g. universal)

Goal: output RAM computation $P(x, y)$

$P(x, y)$ has RAM runtime T

Prep run time: $|y|^{1+\varepsilon}$

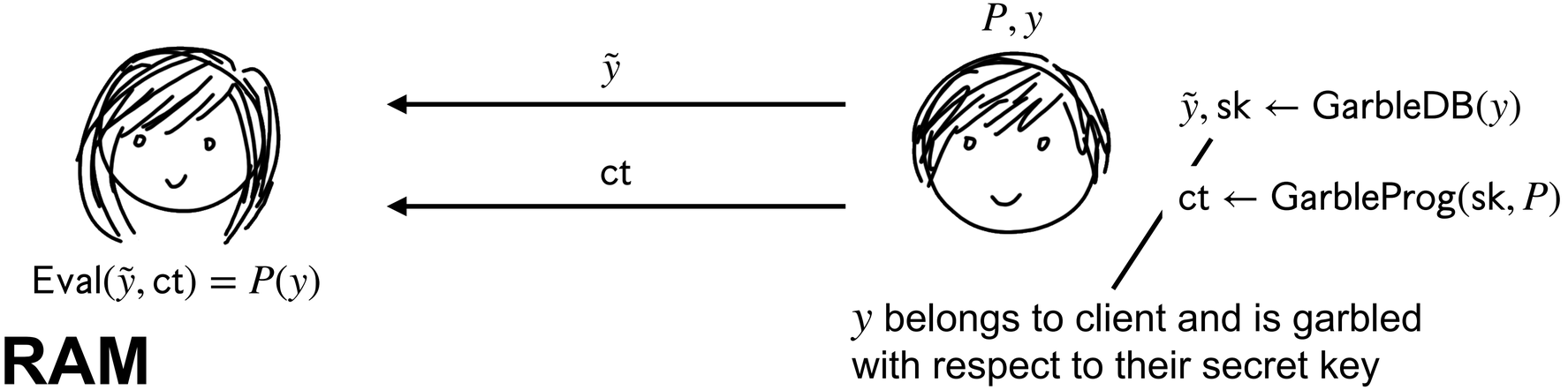


Main Result: We build LFE for RAMs assuming RingLWE

Additionally assuming iO, get Enc run time just $|x|$

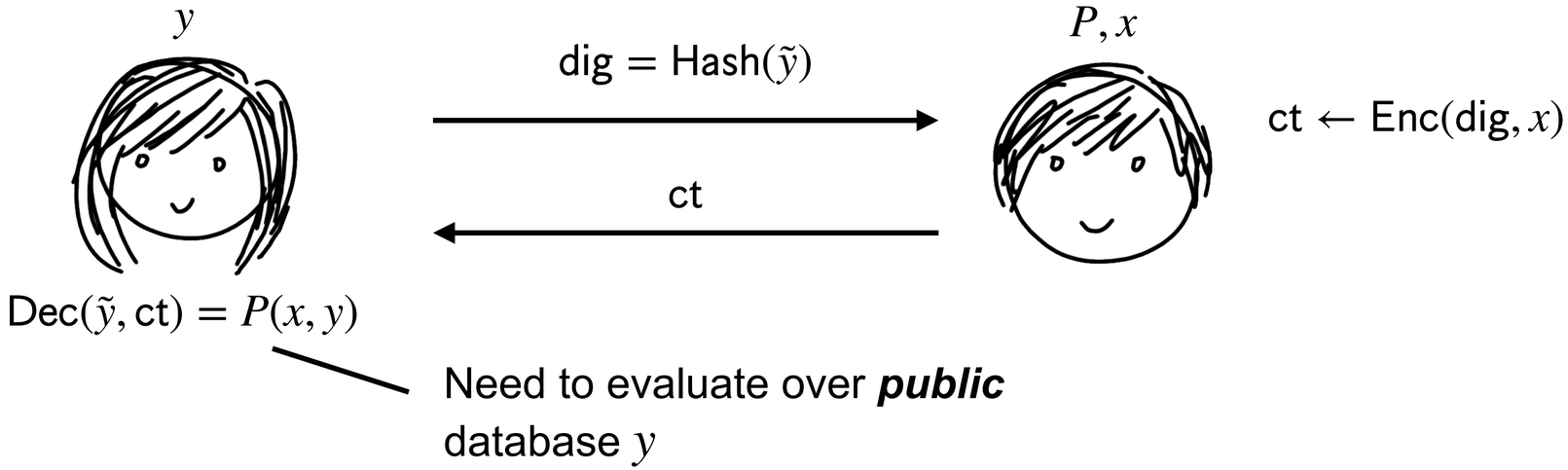
Main challenge: Privately accessing the public database y

RAM-LFE vs Garbled RAM

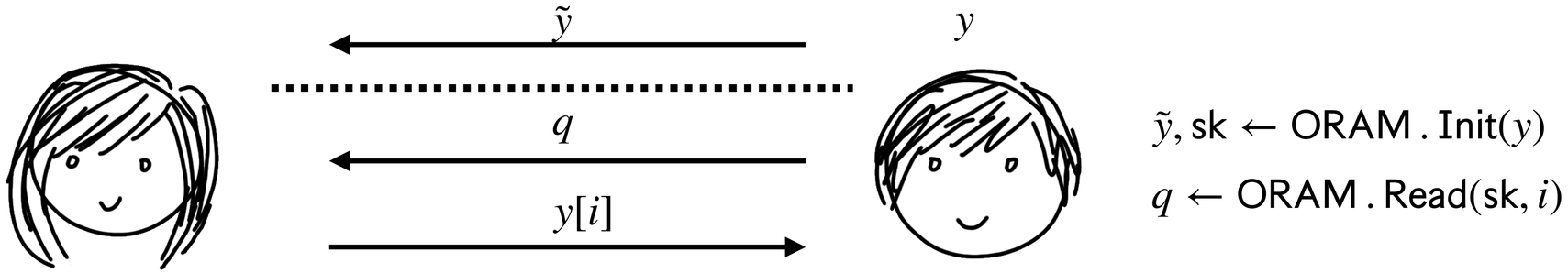


Garbled RAM

RAM-LFE

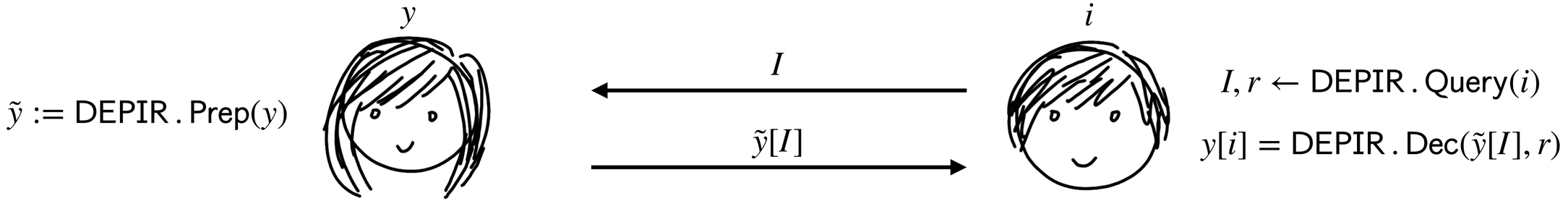


DEPIR vs ORAM



ORAM — Private database, requires client secret key

DEPIR — Public database, public deterministic preprocessing



Prior Work: [Lin-**M**-Wichs'23] build DEPIR from RingLWE

Construction template

We follow the general template for constructing Garbled RAM

1. Construct “UMA” secure version
- Security only protects ***internal state*** not the ***memory access pattern***

For LFE: Crucially need UMA version to allow public database

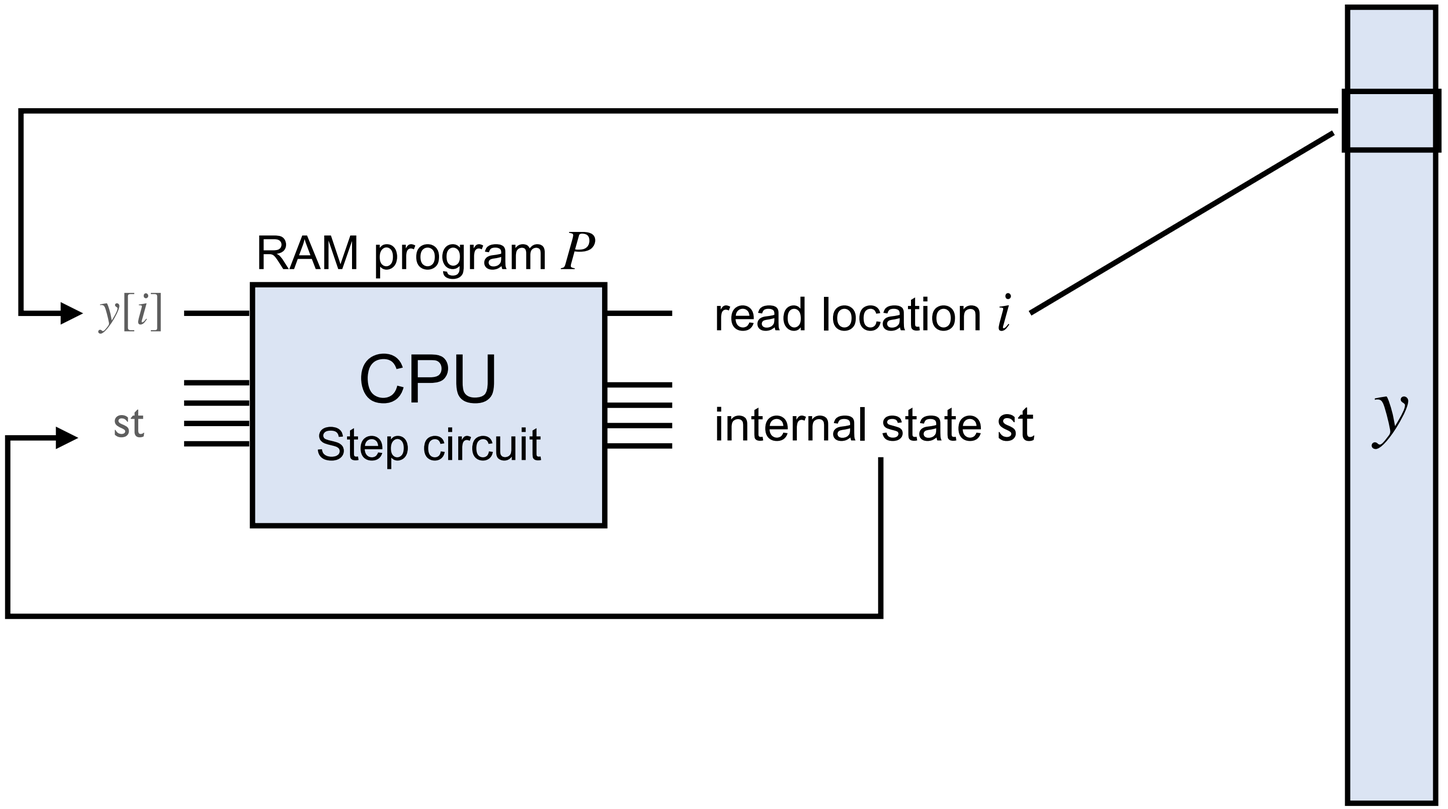
2. Upgrade to full security
- Protect access pattern with ORAM **+ DEPIR**

3. For strong efficiency: Use iO to obfuscate the client’s encryption procedure and offload to server

Requires careful argument and special ORAM construction

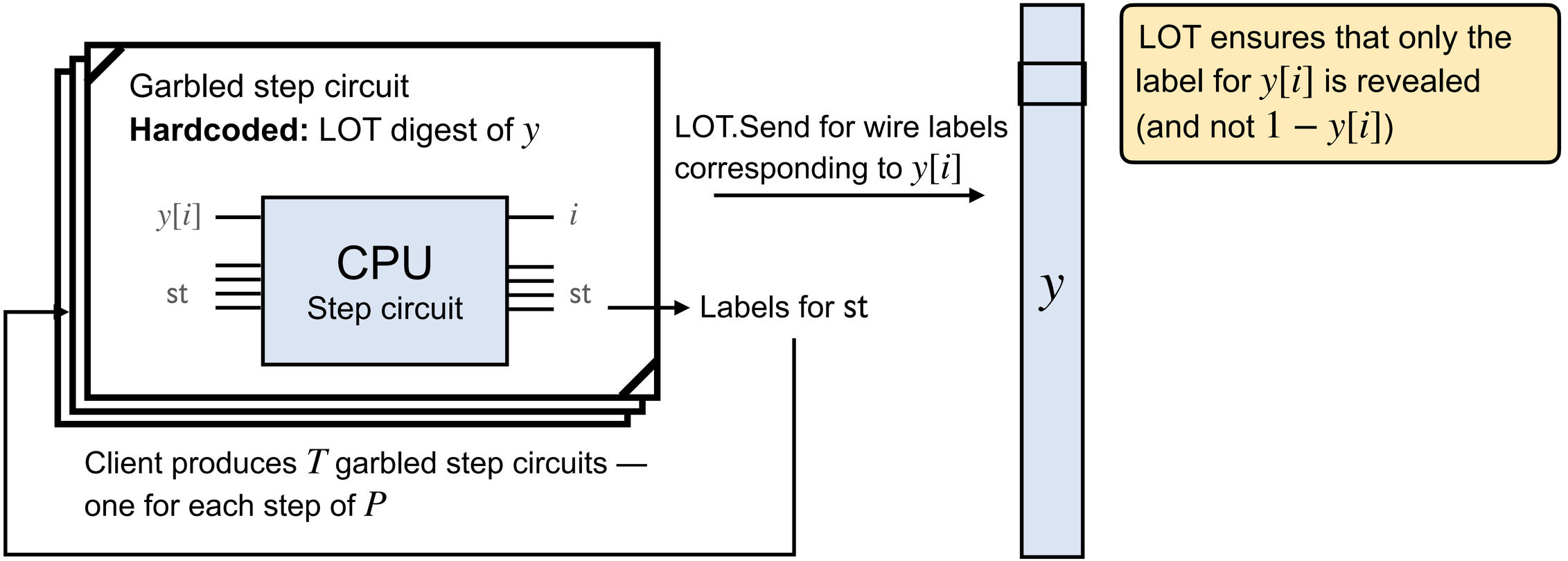
UMA secure RAM-LFE

RAM-NISC from [Cho-Döttling-Garg-Gupta-Miao-Polychroniadou'17]
Building blocks: Laconic Oblivious Transfer + Garbled circuits



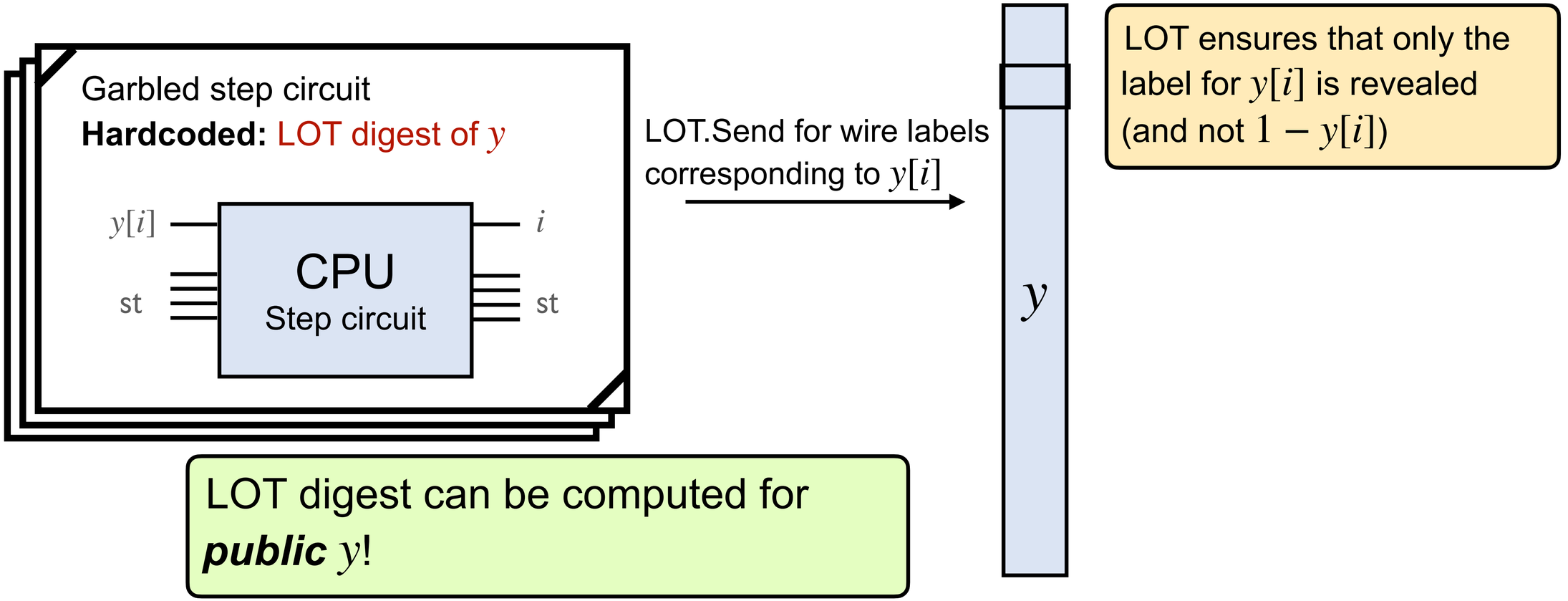
UMA secure RAM-LFE

RAM-NISC from [Cho-Döttling-Garg-Gupta-Miao-Polychroniadou'17]
Building blocks: Laconic Oblivious Transfer + Garbled circuits



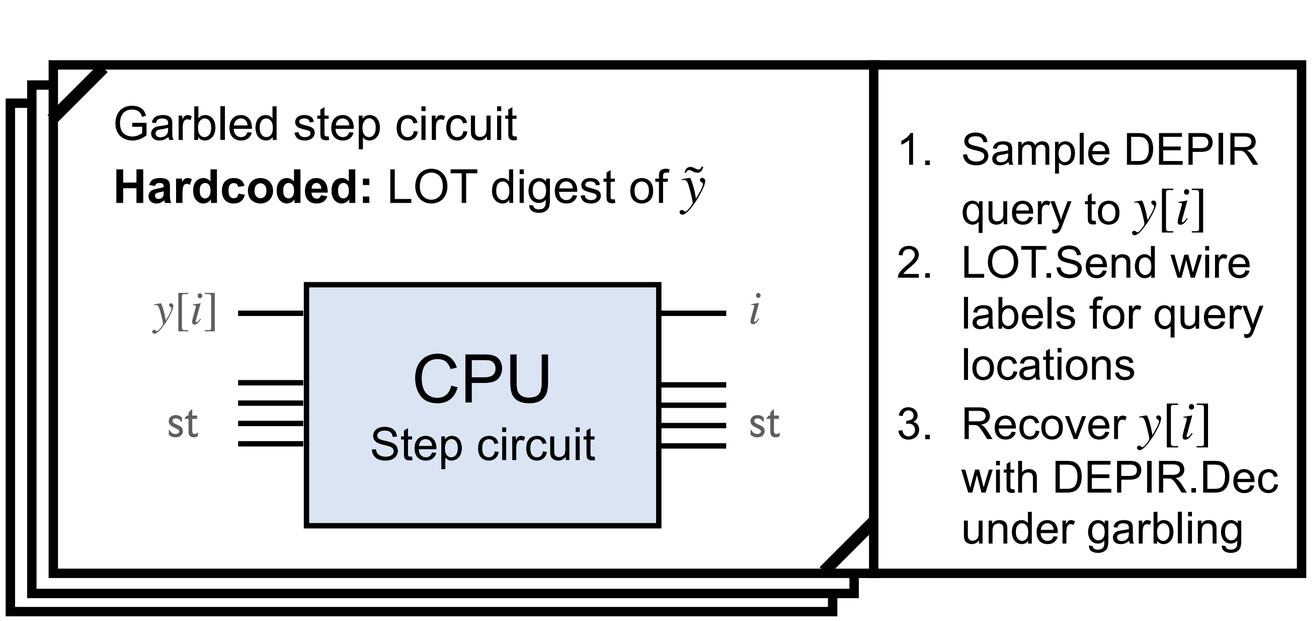
UMA secure RAM-LFE

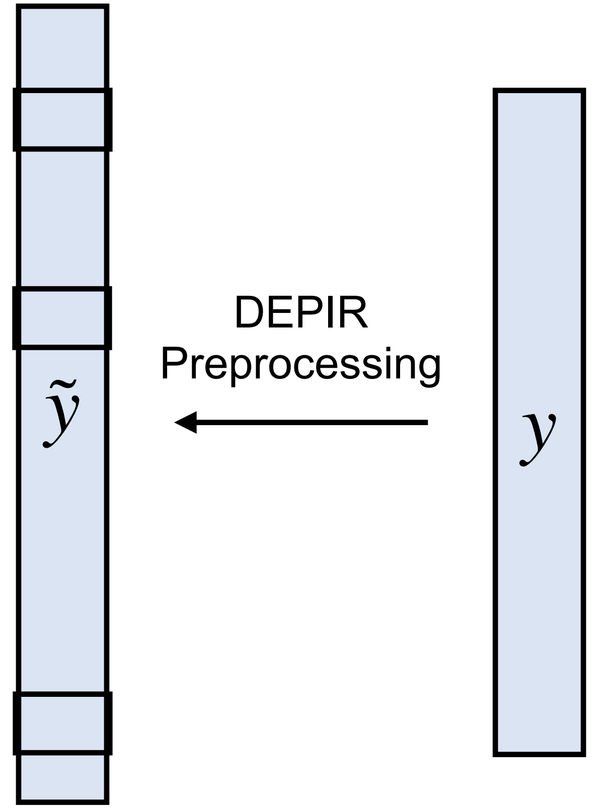
RAM-NISC from [Cho-Döttling-Garg-Gupta-Miao-Polychroniadou'17]
Building blocks: Laconic Oblivious Transfer + Garbled circuits



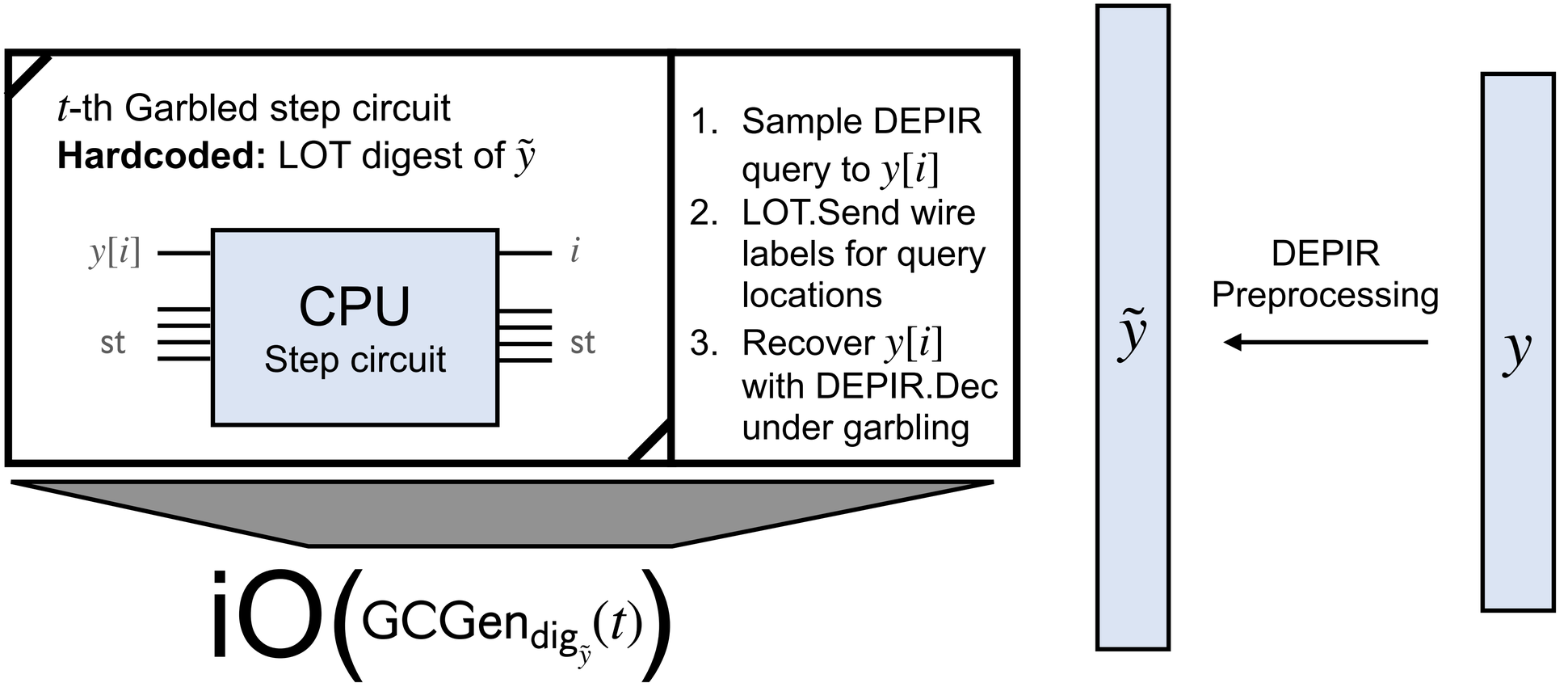
Full security with DEPIR

+ ORAM for the client's database x





Strong Efficiency with iO



Additional Results

Result: We build (multi-key) functional encryption for RAMs

- Each secret key associated to large database y
- Decryption recovers $P(x, y)$ in sublinear time in $|x|, |y|$

Assumptions: FE for circuits + RingLWE

Prior work: [ACFQ'22] only allows short secret keys

Result: We build iO for RAMs

- Given (P, y), obfuscate the program $P(\cdot, y)$
- Evaluation can be sublinear in $|y|$

Assumptions: iO for circuits + RingLWE

Prior work: [BCGHJLPTV'18] doesn't allow sublinear runtime

Thank you!

eprint: 2024/068