SophOMR: Improved Oblivious Message Retrieval from SIMD-Aware Homomorphic Compression

Keewoo Lee (Ethereum Foundation), **Yongdong Yeo (Seoul National University)**

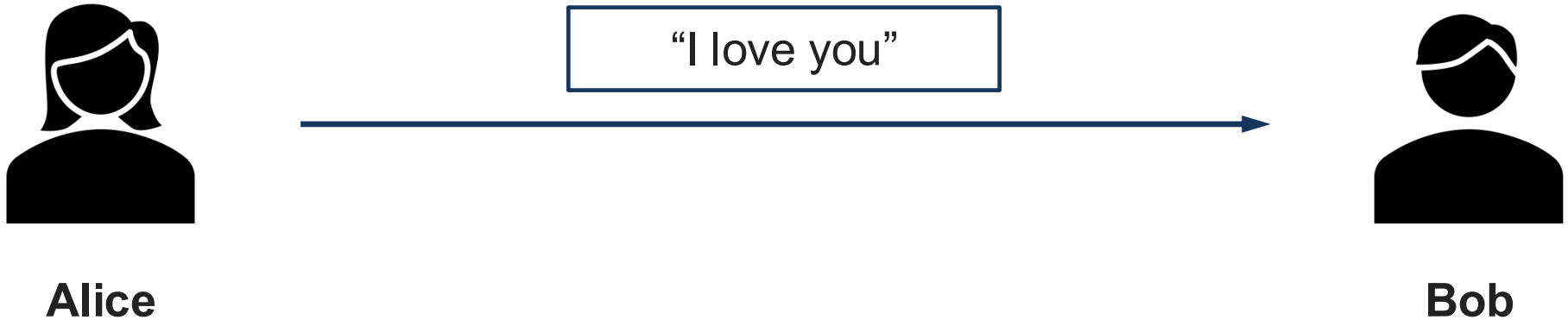




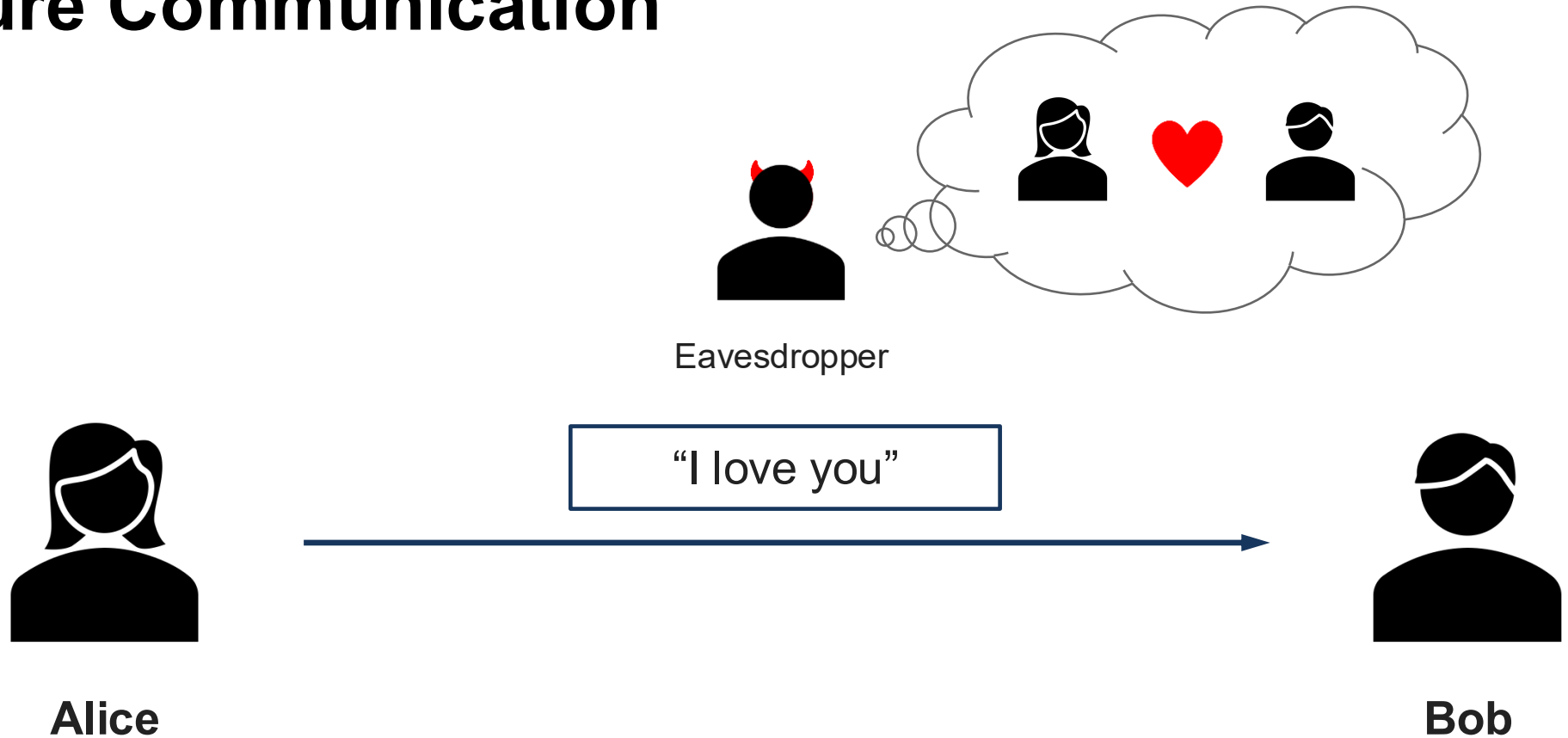
ethereum foundation

USENIX Security 2026

Secure Communication

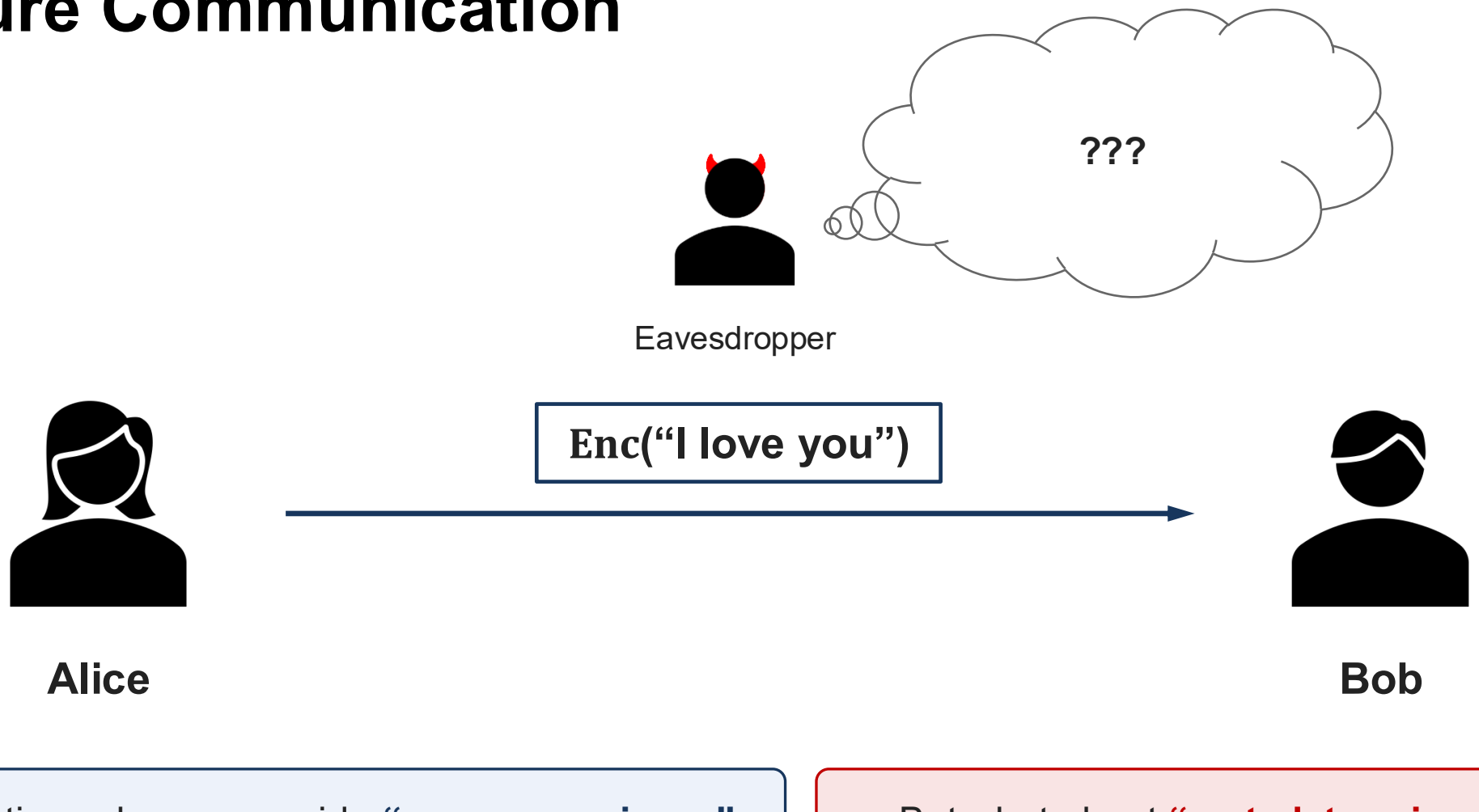


Secure Communication



No protection. The eavesdropper can read the message

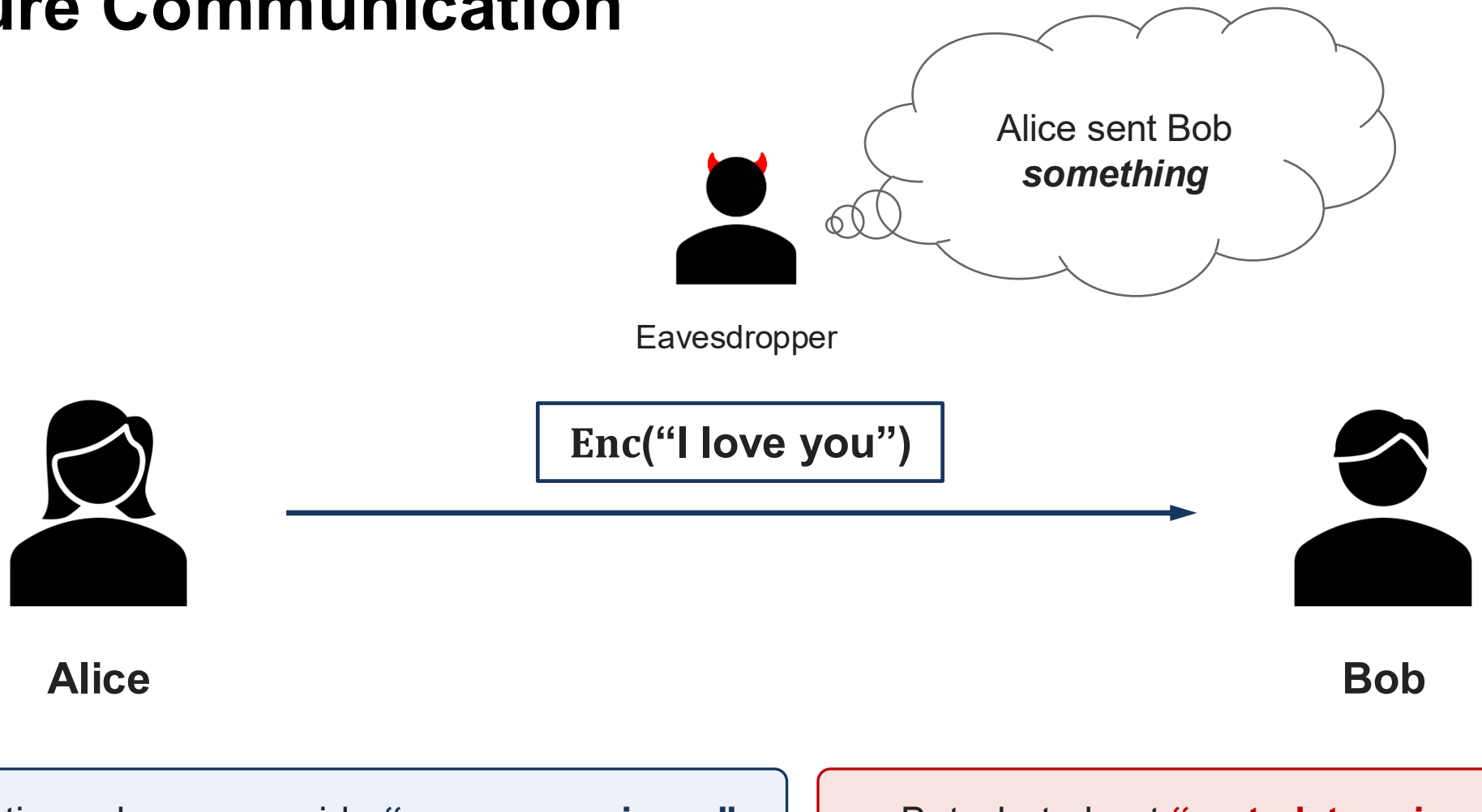
Secure Communication



Encryption schemes provide **"message-privacy"**.

But what about **"metadata-privacy"?**

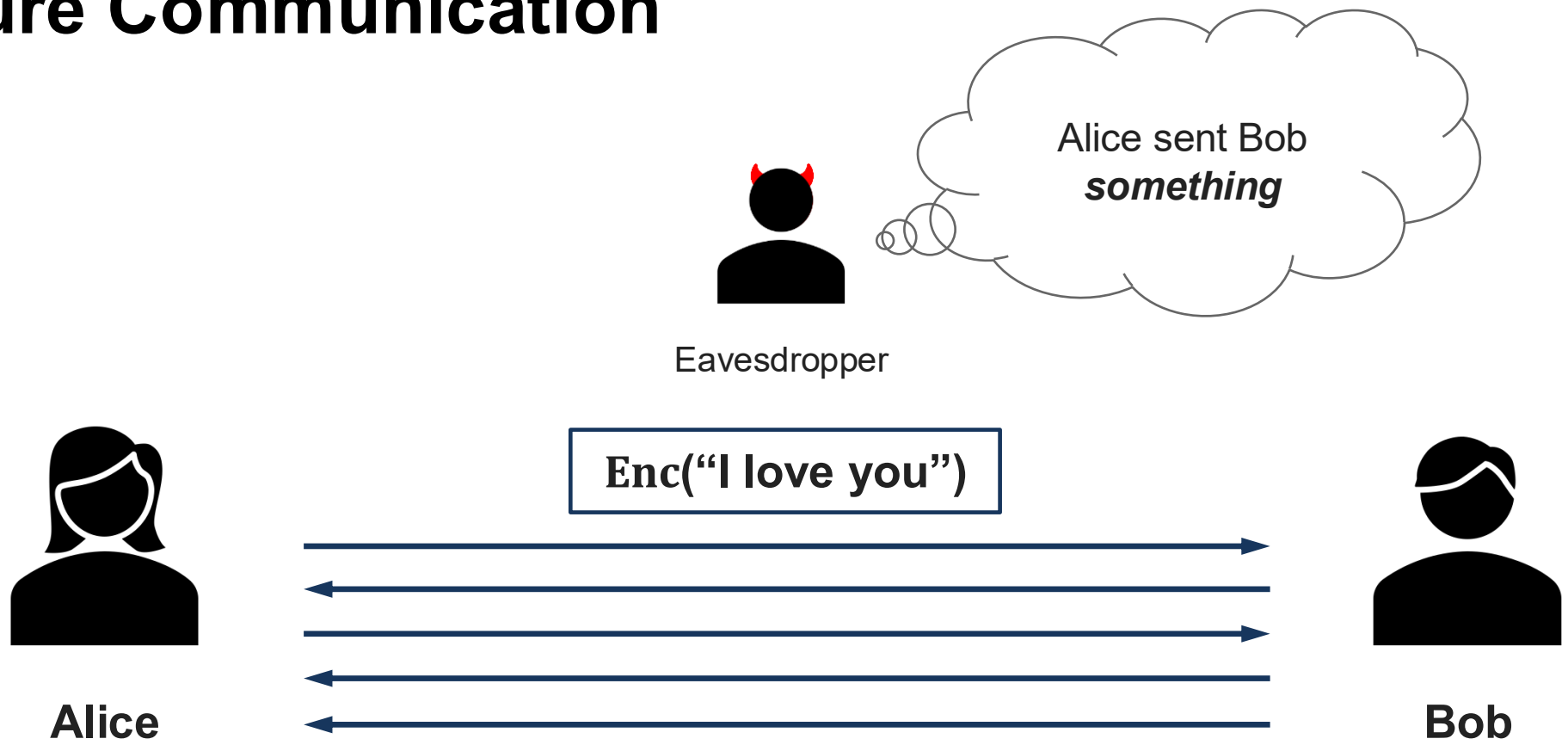
Secure Communication



Encryption schemes provide **"message-privacy"**.

But what about **"metadata-privacy"?**

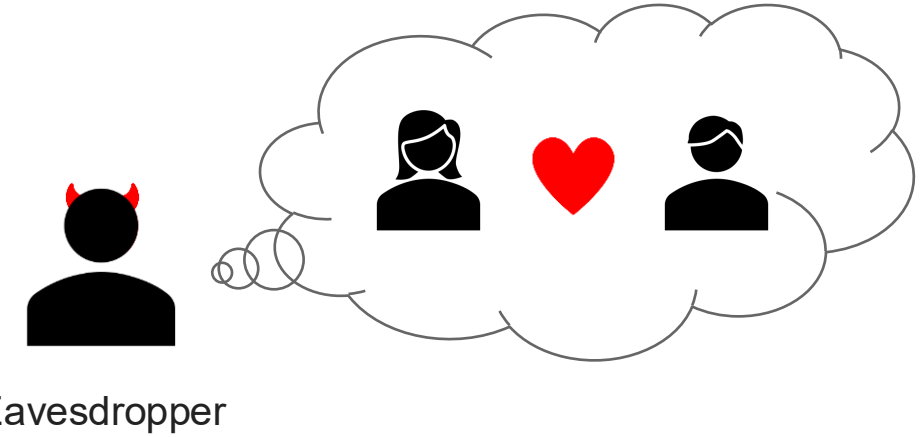
Secure Communication



Encryption schemes provide **"message-privacy"**.

But what about **"metadata-privacy"?**

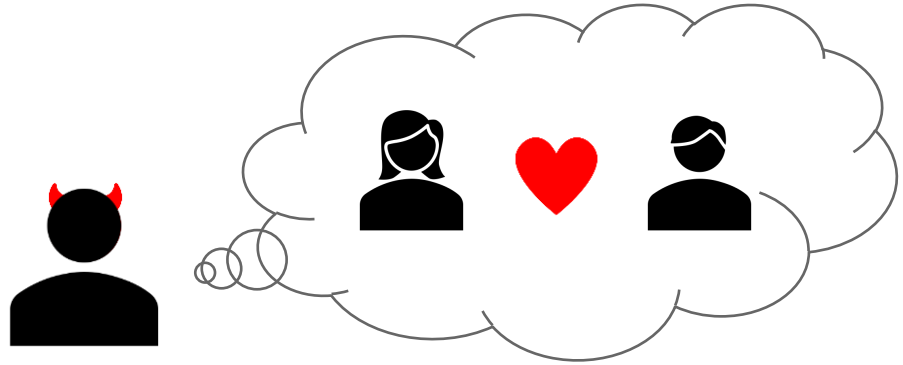
Secure Communication



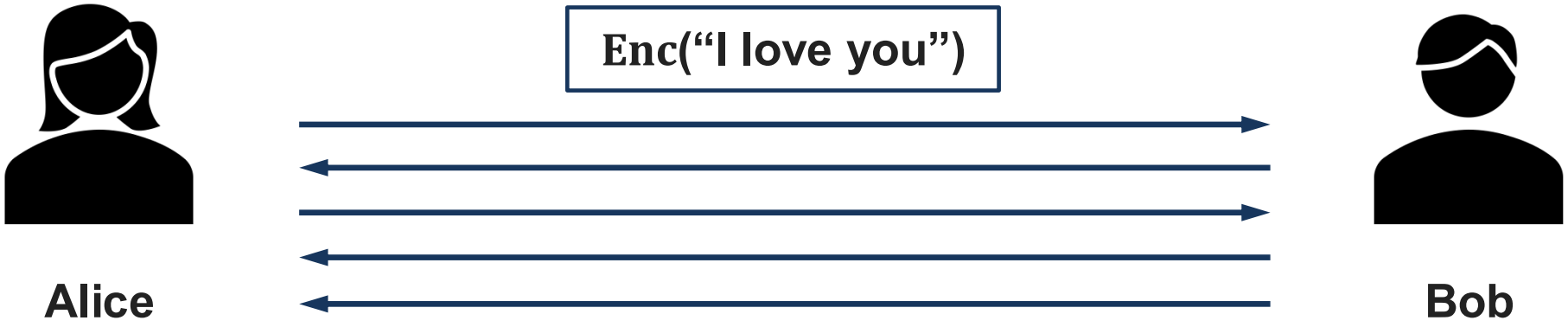
Encryption schemes provide **"message-privacy"**.

But what about **"metadata-privacy"?**

Secure Communication

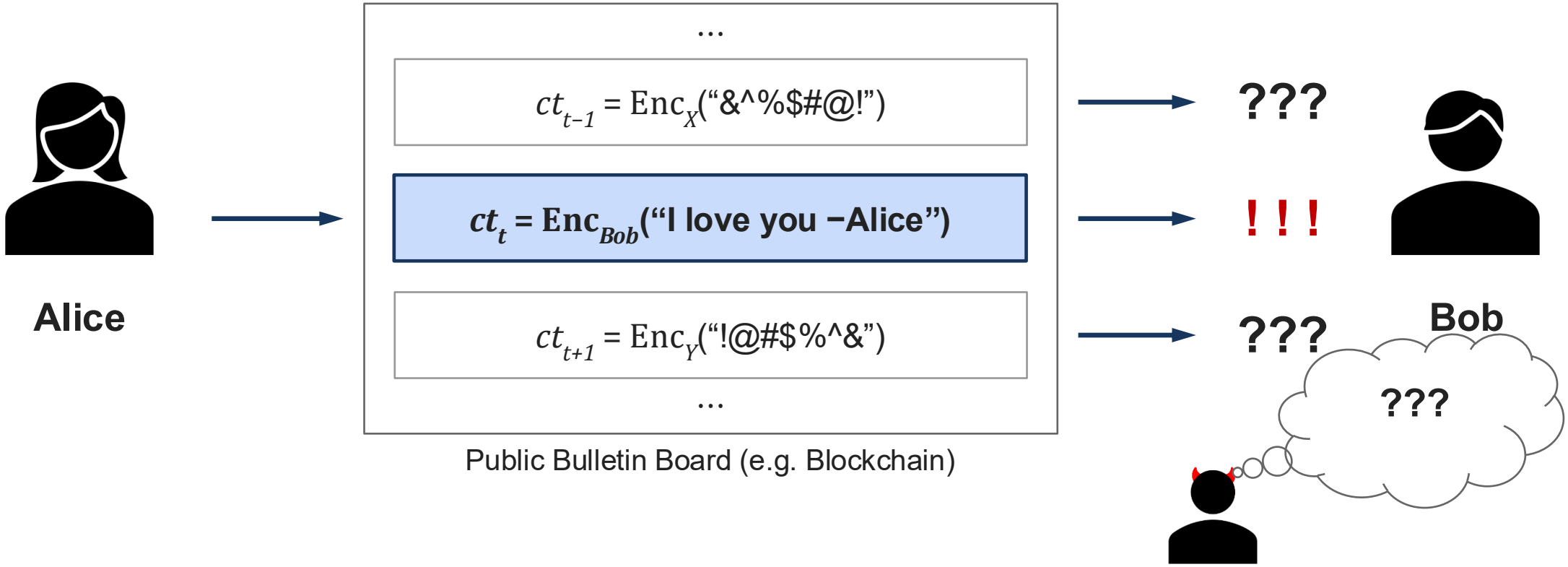


Eavesdropper



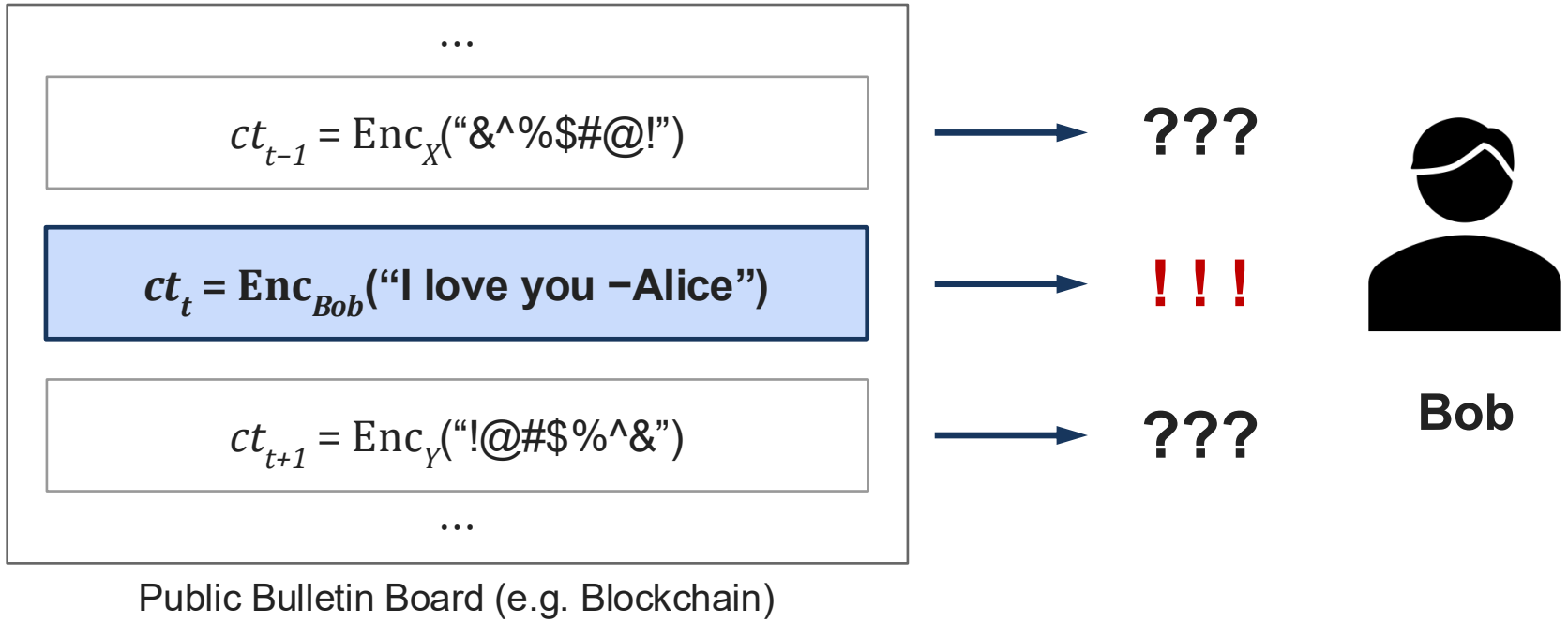
Reciever-privacy: Hiding which message the receiver retrieves

Metadata-Privacy via Public Bulletin Board



Many privacy-preserving blockchains follow this approach.
e.g. Zcash, Aztec, Monero, ERC-5564 (Stealth Address)

Metadata-Privacy via Public Bulletin Board

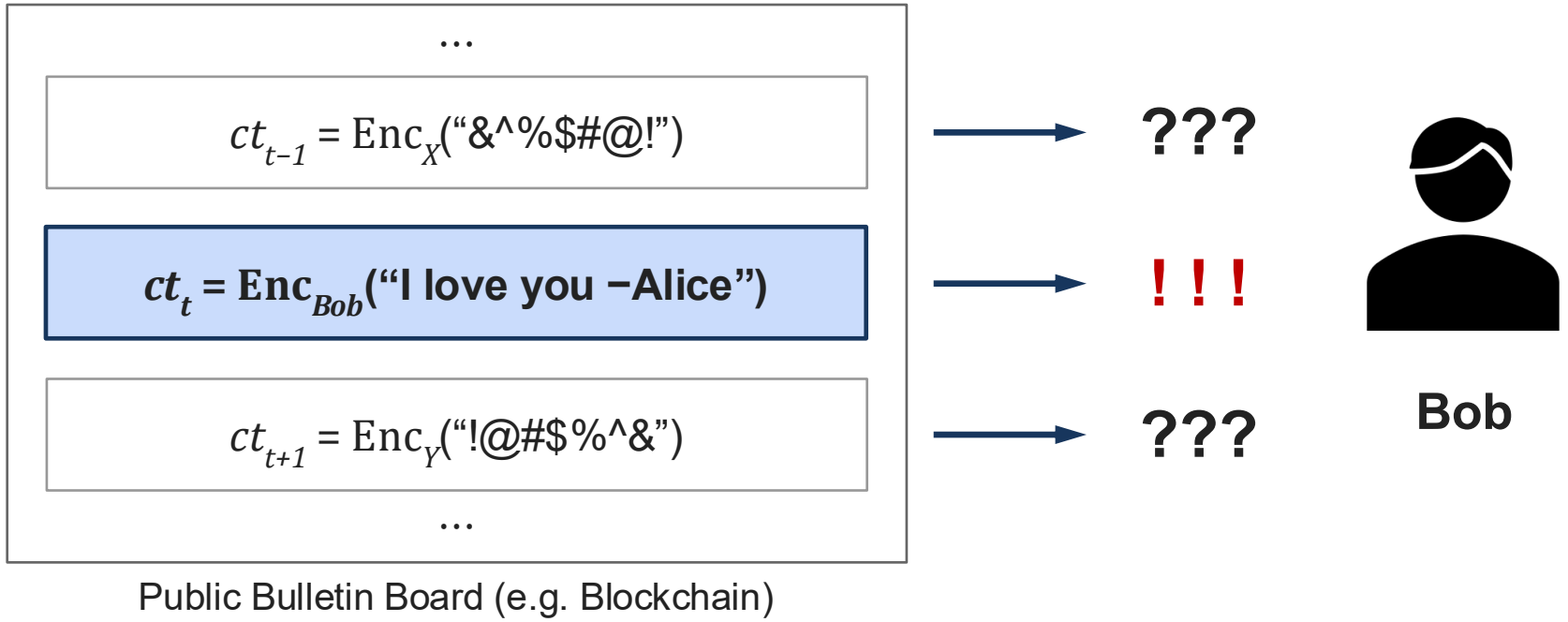


Public Bulletin Board (e.g. Blockchain)

UX Issue. Bob must scan the entire bulletin board

Blockchain context: even light clients must scan every transaction

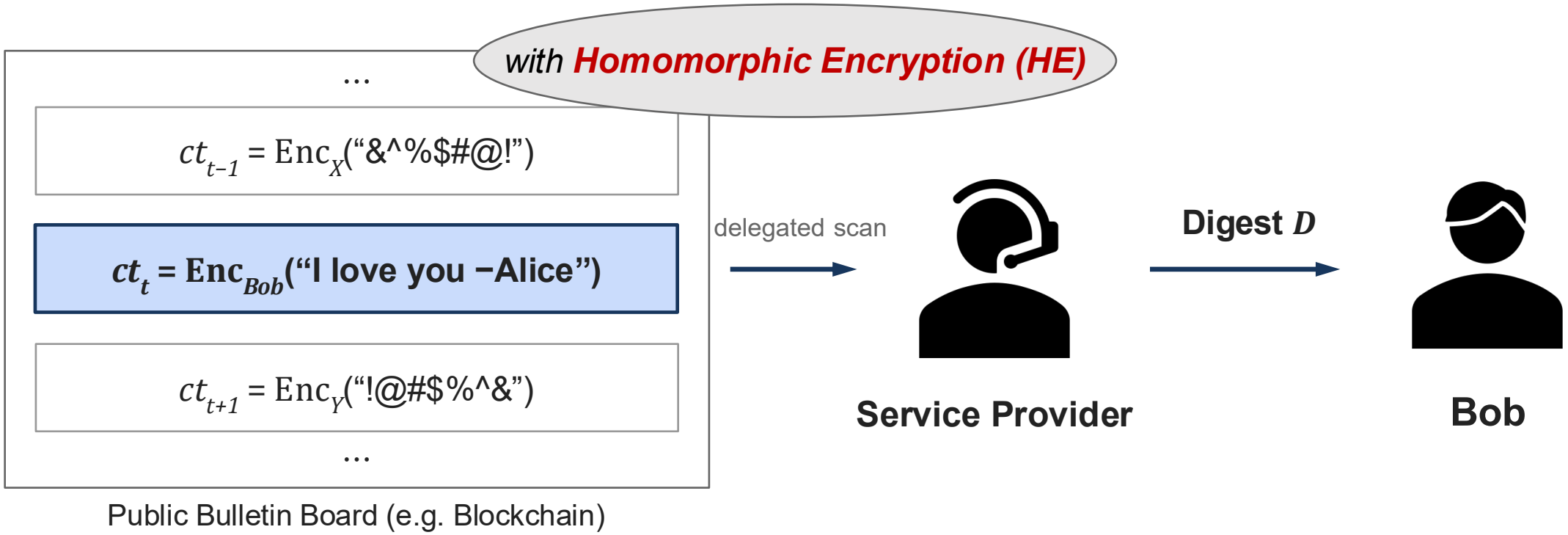
Metadata-Privacy via Public Bulletin Board



UX Issue. Bob must scan the entire bulletin board
Blockchain context: even light clients must scan every transaction

Idea. Let's delegate the expensive scan!

Scan Delegation Framework



① Bob can recover **all** the messages from the Digest
② The Service Provider learns **nothing**
③ The Digest is **small** — ideally, proportional only to #(messages for Bob)

Oblivious Message Retrieval (oversimplified)

Step 1. Homomorphic Decryption (HomDec)

Check decryptability using HE.Enc(sk_{Bob})

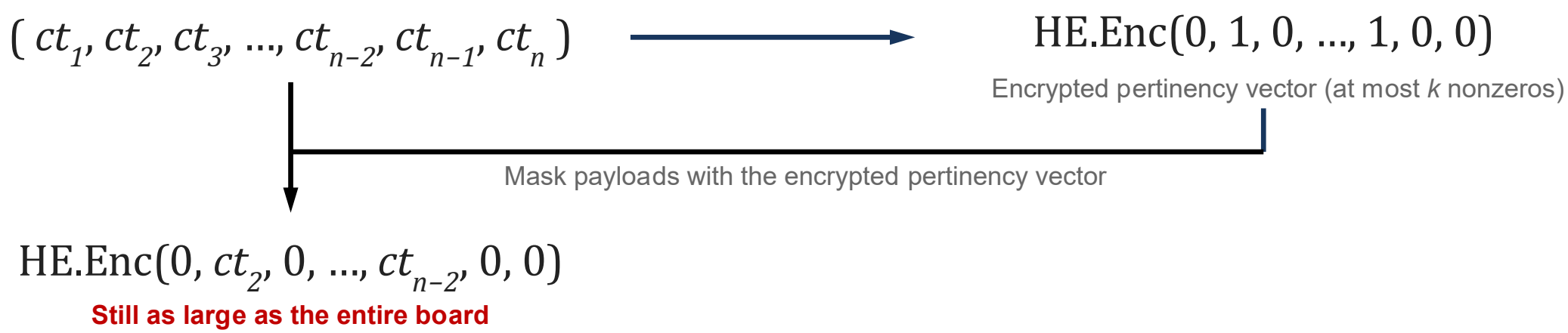
$$(ct_1, ct_2, ct_3, \ldots, ct_{n-2}, ct_{n-1}, ct_n) \longrightarrow \mathrm{HE.Enc}(0, 1, 0, \ldots, 1, 0, 0)$$

Encrypted pertinency vector (at most *k* nonzeros)

[LT22] Zeyu Liu and Eran Tromer. Oblivious Message Retrieval. In Advances in Cryptology – CRYPTO 2022, pages 753–783, 2022.

Oblivious Message Retrieval (oversimplified)

Step 1. Homomorphic Decryption (HomDec)
Check decryptability using HE.Enc(sk_{Bob})



[LT22] Zeyu Liu and Eran Tromer. Oblivious Message Retrieval. In Advances in Cryptology – CRYPTO 2022, pages 753–783, 2022.

Oblivious Message Retrieval (oversimplified)

Step 1. Homomorphic Decryption (HomDec)
Check decryptability using HE.Enc(sk_{Bob})

$(ct_1, ct_2, ct_3, \ldots, ct_{n-2}, ct_{n-1}, ct_n)$ → $\mathrm{HE.Enc}(0, 1, 0, \ldots, 1, 0, 0)$

Encrypted pertinency vector (at most k nonzeros)

Mask payloads with the encrypted pertinency vector

$\mathrm{HE.Enc}(0, ct_2, 0, \ldots, ct_{n-2}, 0, 0)$ → $\mathrm{HE.Enc}(ct_2, \ldots, ct_{n-2})$

Still as large as the entire board

Short digest

Step 2. Homomorphic Compression (HomComp)
Compress "a length-n vector with at most k non-zero entries" into "a length-k vector"

[LT22] Zeyu Liu and Eran Tromer. Oblivious Message Retrieval. In Advances in Cryptology – CRYPTO 2022, pages 753–783, 2022.

Oblivious Message Retrieval (oversimplified)

Step 1. Homomorphic Decryption (HomDec)
Check decryptability using HE.Enc(sk_{Bob})

$(\ ct_1, ct_2, ct_3, \ldots, ct_{n-2}, ct_{n-1}, ct_n\ )$ → $\mathrm{HE.Enc}(0, 1, 0, \ldots, 1, 0, 0)$

Encrypted pertinency vector (at most k nonzeros)

Mask payloads with the encrypted pertinency vector

$\mathrm{HE.Enc}(0, ct_2, 0, \ldots, ct_{n-2}, 0, 0)$ → $\mathrm{HE.Enc}(ct_2, \ldots, ct_{n-2})$

Still as large as the entire board

Short digest

Step 2. Homomorphic Compression (HomComp)
Compress "a length-n vector with at most k non-zero entries" i

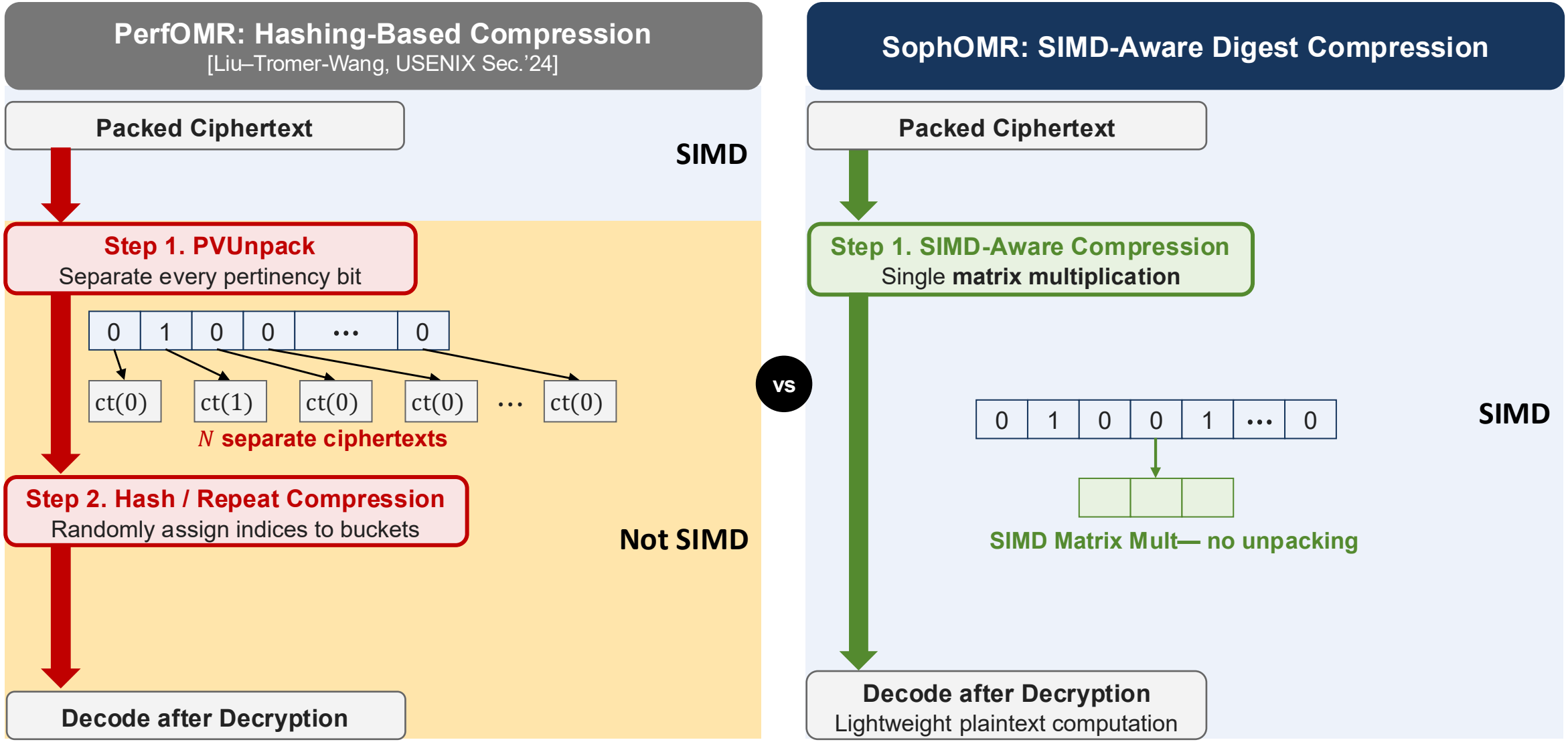
Bottleneck. This was not SIMD-friendly

[LT22] Zeyu Liu and Eran Tromer. Oblivious Message Retrieval. In Advances in Cryptology – CRYPTO 2022, pages 753–783, 2022.

SophOMR

Our scheme: Focusing on SIMD-Aware Digest Compression

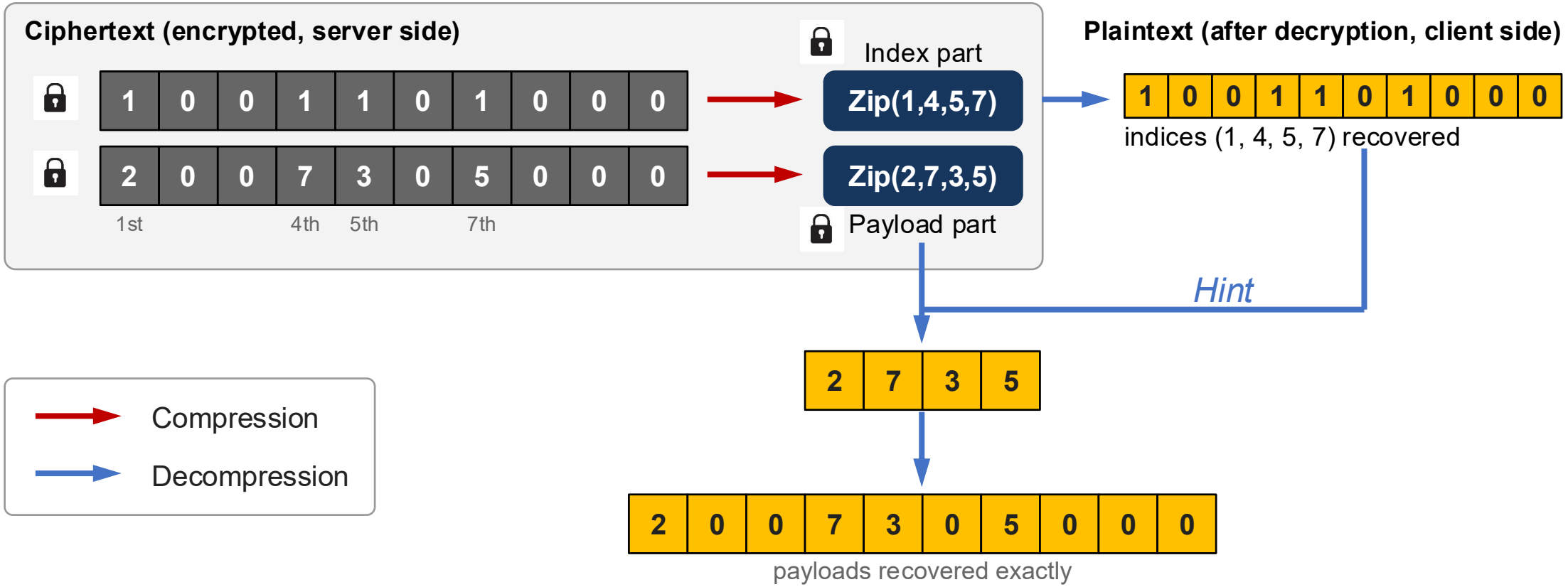
Core Idea: Designing SIMD-Aware Digest Compression



[LTW24] Zeyu Liu, Eran Tromer, and Yunhao Wang. PerfOMR: Oblivious Message Retrieval with Reduced Communication and Computation. In 33rd USENIX Security Symposium (USENIX Security 24), pages 3169–3186, 2024.

SIMD-aware Compression [Cheon-Lee-Park-**Yeo**24]

Homomorphic compression compresses an encrypted *sparse vector* into a small size.



[CLPY24] Jung Hee Cheon, Keewoo Lee, Jai Hyun Park, and Yongdong Yeo. “SIMD-Aware Homomorphic Compression and Application to Private Database Query.” ar Xiv preprint arXiv:2408.17063, 2024.

SIMD-aware Compression [Cheon-Lee-Park-**Yeo**24]

Generalization of converting a 1-hot vector to its nonzero position **(w/ Newton's identity)**

$$(0, 0, 0, 1, 0, 0, 0, 0) \rightarrow (4)$$

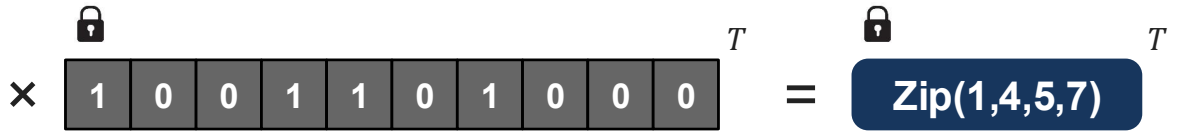
$$(1, 0, 0, 1, 1, 0, 1, 0, 0, 0) \rightarrow (1{+}4{+}5{+}7,\ 1^2{+}4^2{+}5^2{+}7^2,\ 1^3{+}4^3{+}5^3{+}7^3,\ 1^4{+}4^4{+}5^4{+}7^4)$$

[CDG+, CCS'21]

This can be performed by a **SIMD-friendly PCMV multiplication** (plaintext-matrix × ciphertext-vector)

1st			4th	5th		7th			
1	2	3	**4**	**5**	6	**7**	8	9	10
$\mathbf{1^2}$	2^2	3^2	$\mathbf{4^2}$	$\mathbf{5^2}$	6^2	$\mathbf{7^2}$	8^2	9^2	10^2
$\mathbf{1^3}$	2^3	3^3	$\mathbf{4^3}$	$\mathbf{5^3}$	6^3	$\mathbf{7^3}$	8^3	9^3	10^3
$\mathbf{1^4}$	2^4	3^4	$\mathbf{4^4}$	$\mathbf{5^4}$	6^4	$\mathbf{7^4}$	8^4	9^4	10^4

Matrix C



[CLPY24] Jung Hee Cheon, Keewoo Lee, Jai Hyun Park, and Yongdong Yeo. "SIMD-Aware Homomorphic Compression and Application to Private Database Query." ar Xiv preprint arXiv:2408.17063, 2024.

[CDG+21] Seung Geol Choi, Dana Dachman-Soled, S. Dov Gordon, Linsheng Liu, and Arkady Yerukhimovich. Compressed Oblivious Encoding for Homomorphically Encrypted Search. In Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security (CCS), pages 2277–2291, 2021.

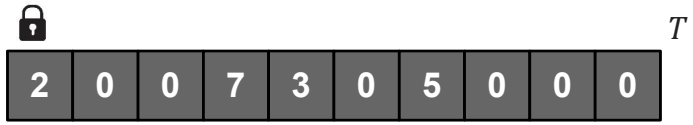
SIMD-aware Compression [Cheon-Lee-Park-**Yeo**24]

Compression:

** For an arbitrary sparse vector, the same matrix C compresses the payload vector*

1st			4th	5th		7th			
$\mathbf{1}$	2	3	$\mathbf{4}$	$\mathbf{5}$	6	$\mathbf{7}$	8	9	10
$\mathbf{1^2}$	2^2	3^2	$\mathbf{4^2}$	$\mathbf{5^2}$	6^2	$\mathbf{7^2}$	8^2	9^2	10^2
$\mathbf{1^3}$	2^3	3^3	$\mathbf{4^3}$	$\mathbf{5^3}$	6^3	$\mathbf{7^3}$	8^3	9^3	10^3
$\mathbf{1^4}$	2^4	3^4	$\mathbf{4^4}$	$\mathbf{5^4}$	6^4	$\mathbf{7^4}$	8^4	9^4	10^4

Matrix C

$\times$ 

encrypted (server)

$=$ 

Decompression (after decryption & index recovery):

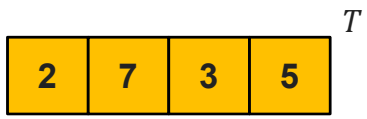
** We already have the index data (1, 4, 5, 7) from the previous index decompression*

$\mathbf{1}$	$\mathbf{4}$	$\mathbf{5}$	$\mathbf{7}$
$\mathbf{1^2}$	$\mathbf{4^2}$	$\mathbf{5^2}$	$\mathbf{7^2}$
$\mathbf{1^3}$	$\mathbf{4^3}$	$\mathbf{5^3}$	$\mathbf{7^3}$
$\mathbf{1^4}$	$\mathbf{4^4}$	$\mathbf{5^4}$	$\mathbf{7^4}$

$^{-1}$

columns 1, 4, 5, 7 of C

$\times$ 

$=$ 

payloads at 1st, 4th, 5th, 7th

Note. Any 4 column vectors of C are linearly independent.
Solve one 4×4 linear system in plaintext.
(exact, nothing probabilistic)

[CLPY24] Jung Hee Cheon, Keewoo Lee, Jai Hyun Park, and Yongdong Yeo. “SIMD-Aware Homomorphic Compression and Application to Private Database Query.” ar Xiv preprint arXiv:2408.17063, 2024.

Matrix Multiplication [LHH+21]

For $k \ll N$, a homomorphic $k \times N$ matrix multiplication can be evaluated efficiently in CKKS, with cost depending mostly on k.

Method	#Rotation
HElib [HS18]	$O(\sqrt{N})$
GAZELLE [JVC18]	$O(\lceil kN/n \rceil + \log(n/N))$
PEGASUS [LHH+21]	$\boldsymbol{O(\sqrt{k} + \log(N/k))}$

∗ N: # of payloads ∗ n: BFV ring dimension ∗ k: # of pertinent payloads ∗ t: # of BFV slots per payload ∗ p: BFV plaintext modulus

Stacking. $\boldsymbol{(t+1)}$**-times** $\boldsymbol{k \times N}$ **MatMul** $\rightarrow$ **1-time** $\boldsymbol{((t+1)k) \times N}$ **MatMul**
$\sqrt{(t+1)}$-times fewer homomorphic rotation operations (BSGS-style)

[HS18] S. Halevi and V. Shoup. Faster homomorphic linear transformations in HElib. CRYPTO 2018.
[JVC18] C. Juvekar, V. Vaikuntanathan, and A. Chandrakasan. GAZELLE: A low latency framework for secure neural network inference. USENIX Security 2018.
[LHH+21] W. Lu, Z. Huang, C. Hong, Y. Ma, and H. Qu. PEGASUS: Bridging polynomial and non-polynomial evaluations in homomorphic encryption. IEEE S&P 2021.

Cost Comparison for Digest Compression

	PerfOMR	SophOMR
#(PMul)	$\left(\frac{kt}{n} + 1 + \frac{vt}{n \log p}\right) N$	$\left(\frac{kt}{n} + \frac{k}{n}\right) N$
#(Rot)	$\left(4 + \frac{2}{\sqrt{n}} - \frac{4}{n}\right) N$	$\sqrt{\frac{4k(t+1)}{n}} \sqrt{N} + \log\left(\frac{n}{k(t+1)}\right)$
Depth	2	1

* N: # of payloads
* n: BFV ring dimension
* t: # of BFV slots per payload
* k: # of pertinent payloads
* p: BFV plaintext modulus
* $2^{-\nu}$: false negative probability

PerfOMR [LTW24]

✗ ***O(N)*** **homomorphic rotations · depth 2**

✗ **One extra level spent on PVUnpack**

SophOMR

✓ ***O(√N)*** **homomorphic rotations · depth 1**

✓ **No unpacking — SIMD throughout**

[LTW24] Zeyu Liu, Eran Tromer, and Yunhao Wang. PerfOMR: Oblivious Message Retrieval with Reduced Communication and Computation. In 33rd USENIX Security Symposium (USENIX Security 24), pages 3169–3186, 2024.

Evaluation: Faster and Smaller

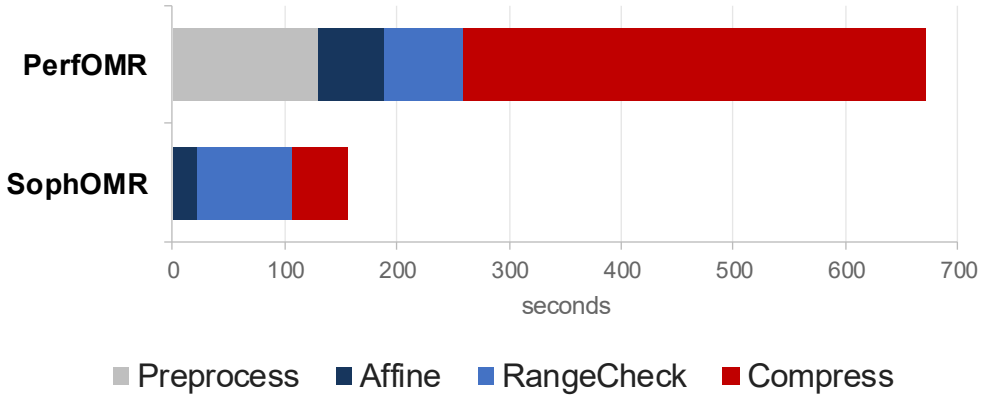
BSGS Affine Transform | Hybrid Key-Switching | Ring-Switching

HE Optimizations

End-to-End Results ($N = 2^{16}$, $k = 50$, single thread)

Scheme	Digest	Decode	Key	Digest size
PerfOMR	541 s	14.6 ms	171 MB	567 KB
SophOMR	157 s	14.0 ms	114 MB	263 KB

Digest Runtime Breakdown



3.4×
Faster OMR digest
541 s → 157 s

2.2×
Smaller OMR digest
567 KB → 263 KB

1.5×
Smaller detection key
171 MB → 114 MB

7.5×
Faster OMR compression
main source of speedup

Thank You

SophOMR: Improved Oblivious Message Retrieval from SIMD-Aware Homomorphic Compression

Full paper of SophOMR
eprint.iacr.org/2024/1814

Implementation code
github.com/keewoolee/SophOMR

Yongdong Yeo
@ Seoul National University
yongdongyeo.github.io

Keewoo Lee
@ Ethereum Foundation
keewoolee.github.io

ethereum foundation