A joint work by *City University of Hong Kong* | *Wuhan University* | *ETH Zurich*

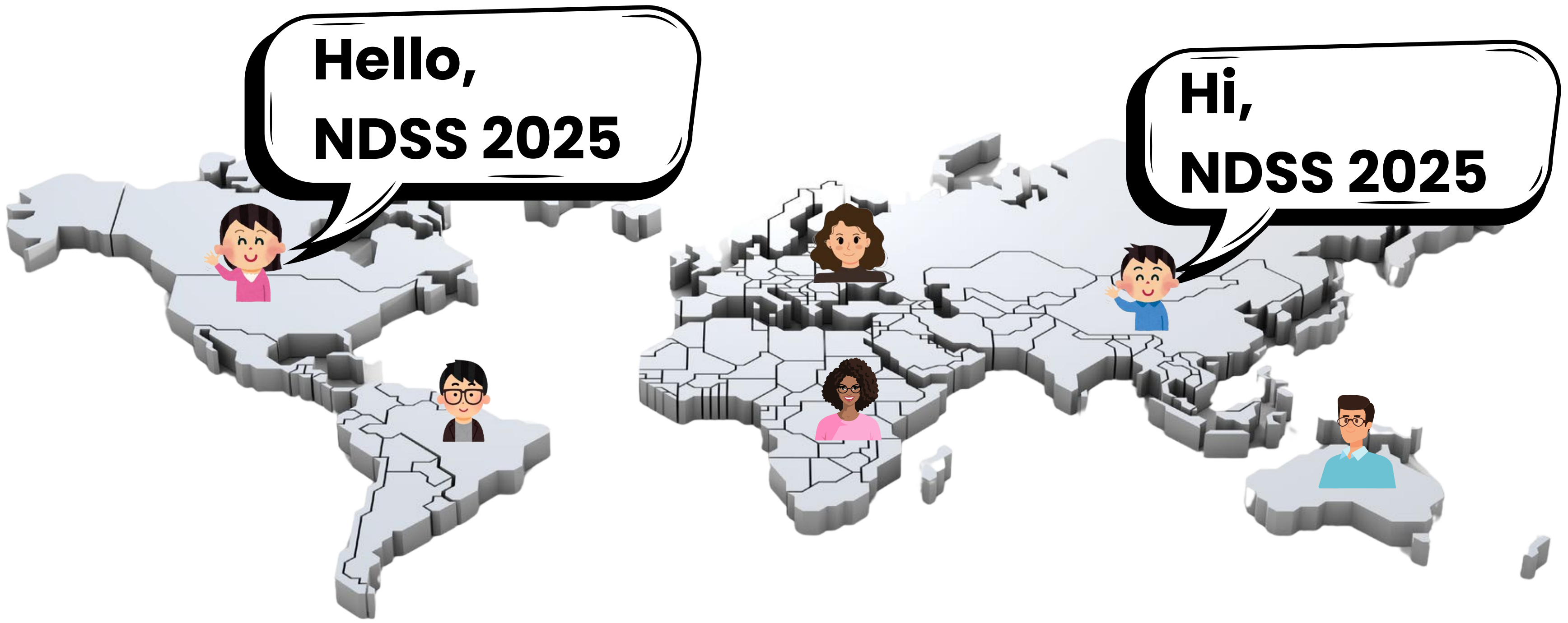
Ring of Gyges: Accountable Anonymous Broadcast via Secret-Shared Shuffle

Dong Wentao[1], Jiang Peipei[1,2], Duan Huayi[3],
Wang Cong[1], Zhao Lingchen[2], Wang Qian[2].

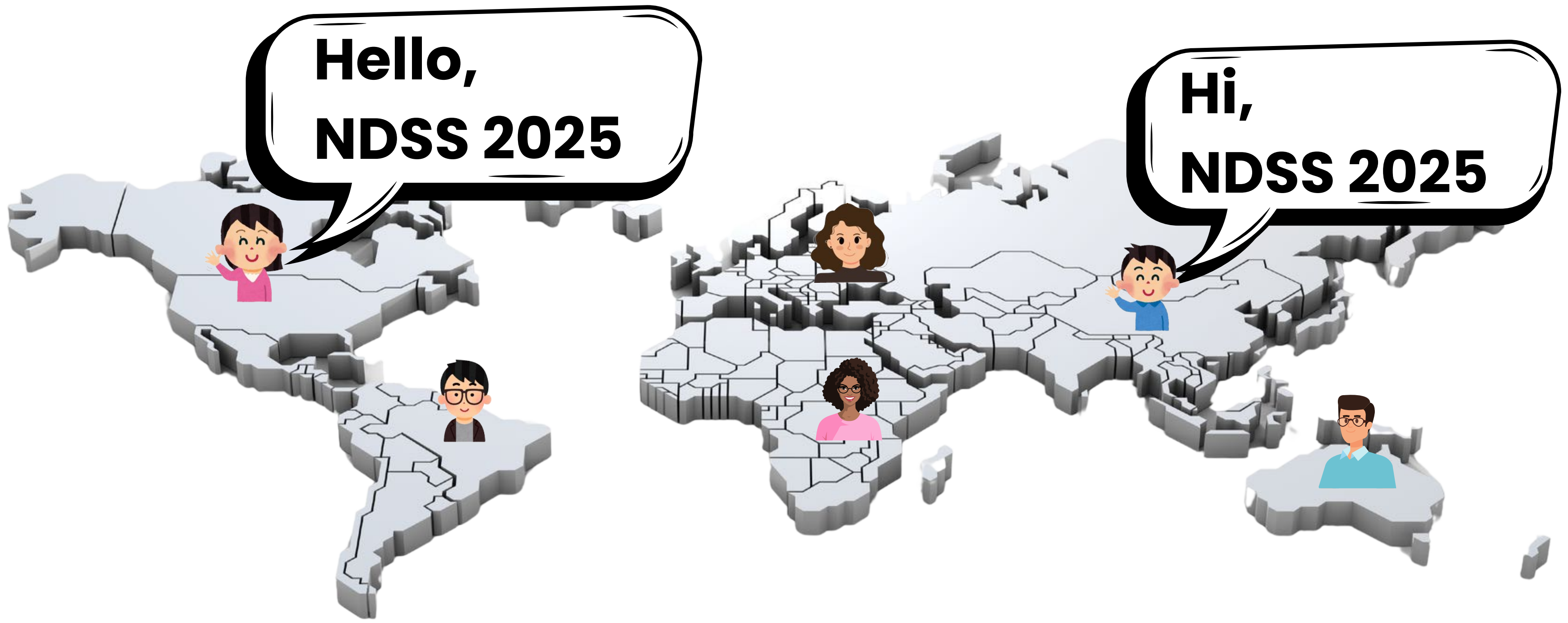
[1]**City University of Hong Kong**, [2]Wuhan University, [3]ETH Zurich

Network and Distributed System Security (NDSS) Symposium
Feb. 24–28, 2025 | San Diego, California, USA

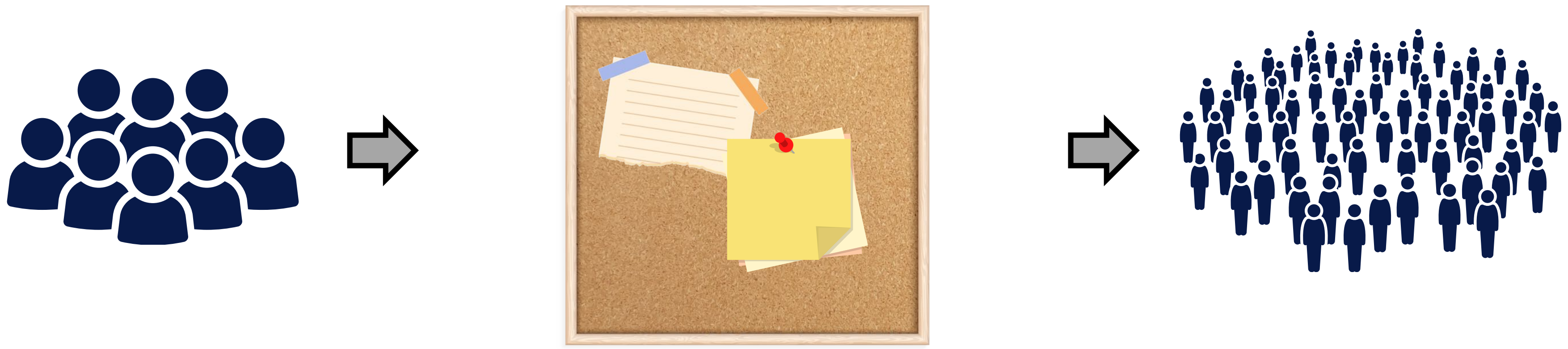
Broadcast



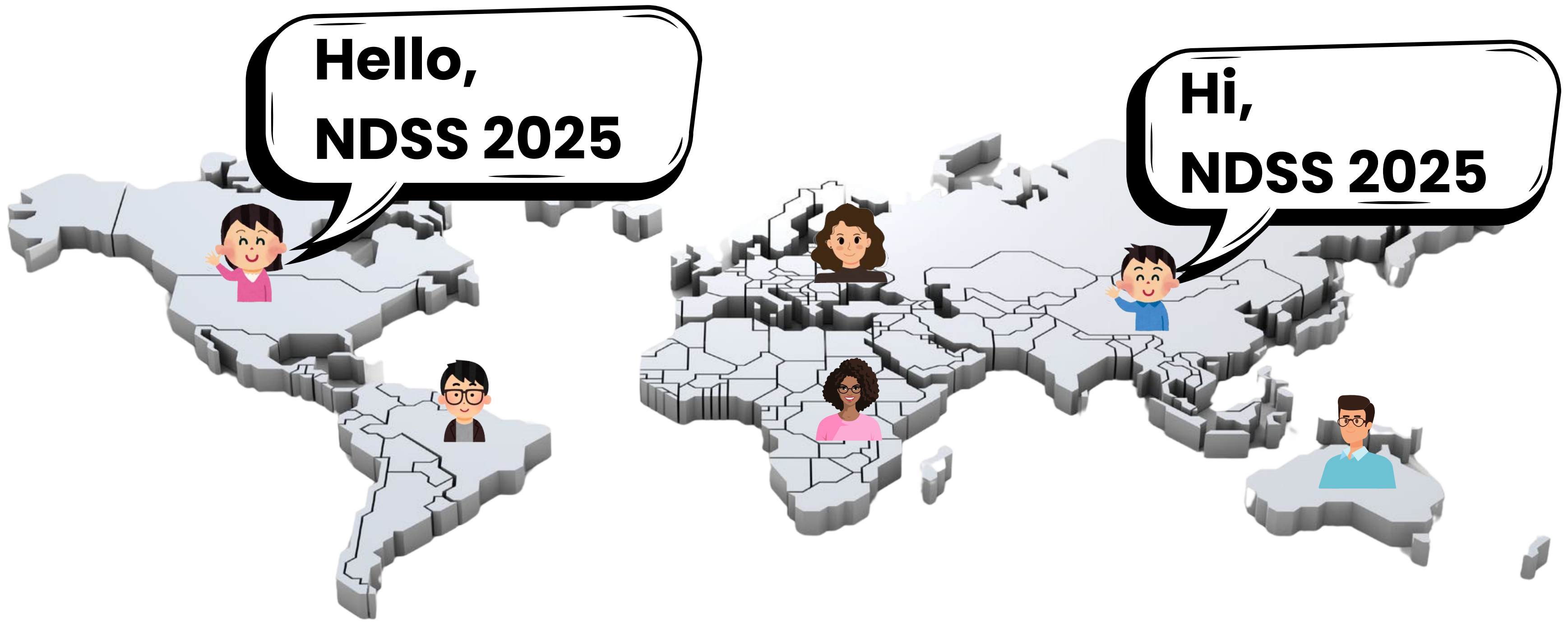
Broadcast



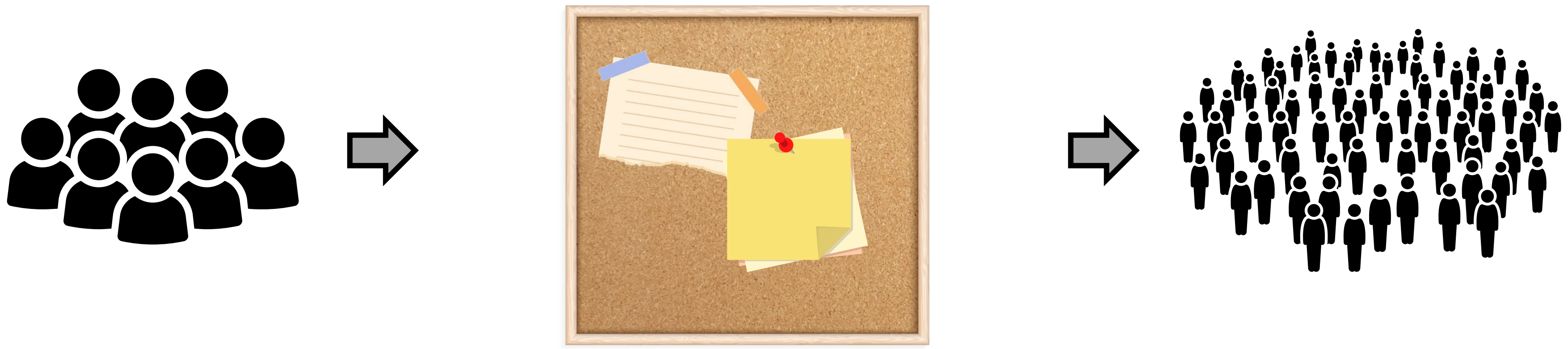
Senders **Public bulletin board** **Receivers**



Broadcast



Senders **Public bulletin board** **Receivers**

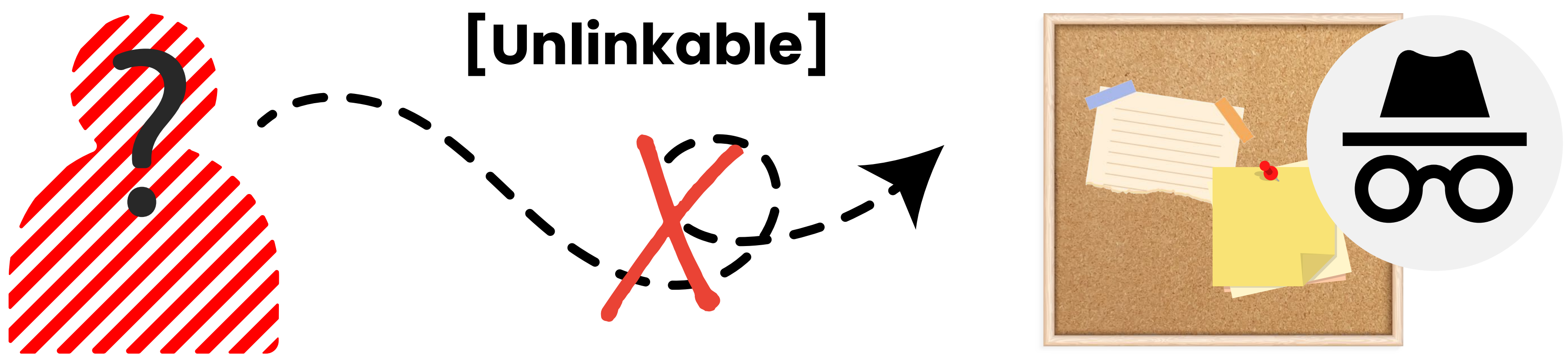


A few widely used broadcast platforms:

Privacy Needs: Anonymous Broadcast

Goal: *Sender anonymity* - no one knows who has sent which message.

Senders **Public bulletin board**



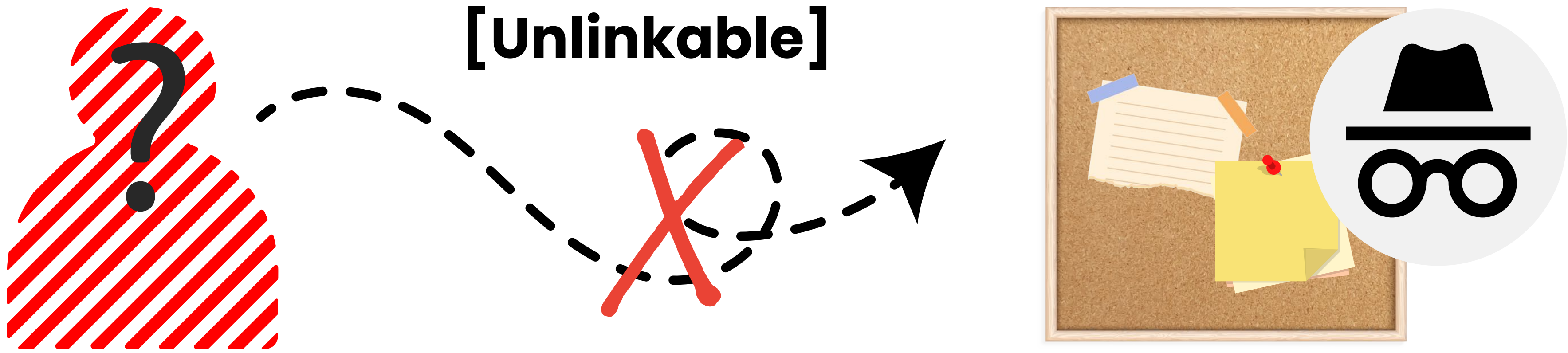
Privacy Needs: Anonymous Broadcast

Goal: *Sender anonymity* - no one knows who has sent which message.

Use Cases:

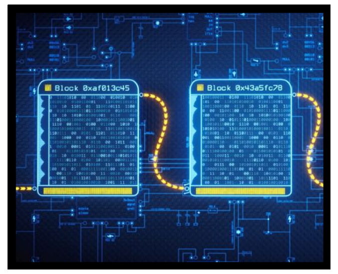
- Whistleblowing,
- Anonymous social networks,
- Anti-censorship reporting,
- ...

Senders **Public bulletin board**



Whistleblower

Blockchain community



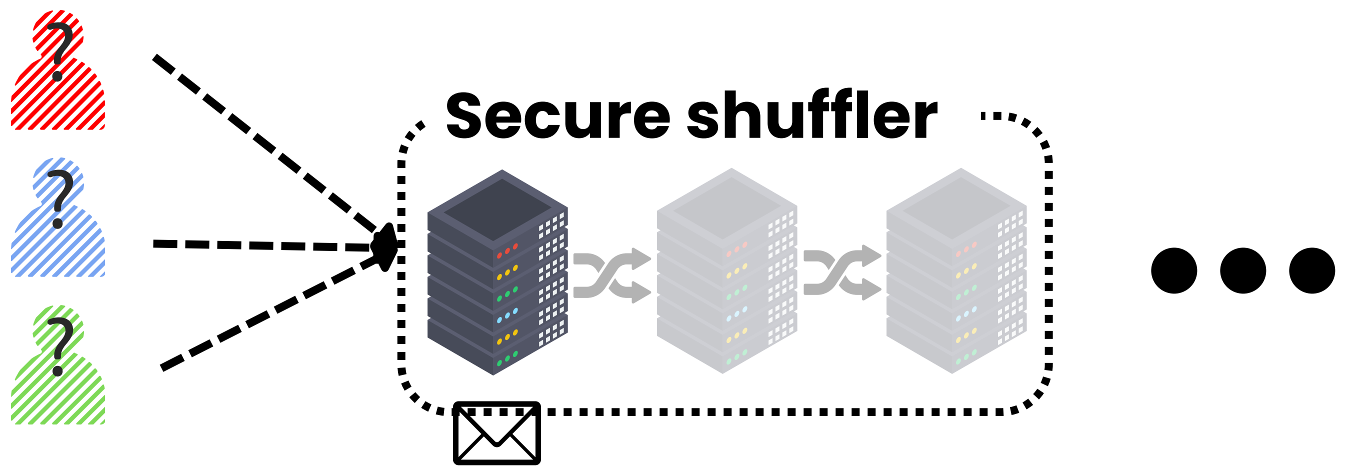
Anonymous social network

Censorship-resistant publishing

...

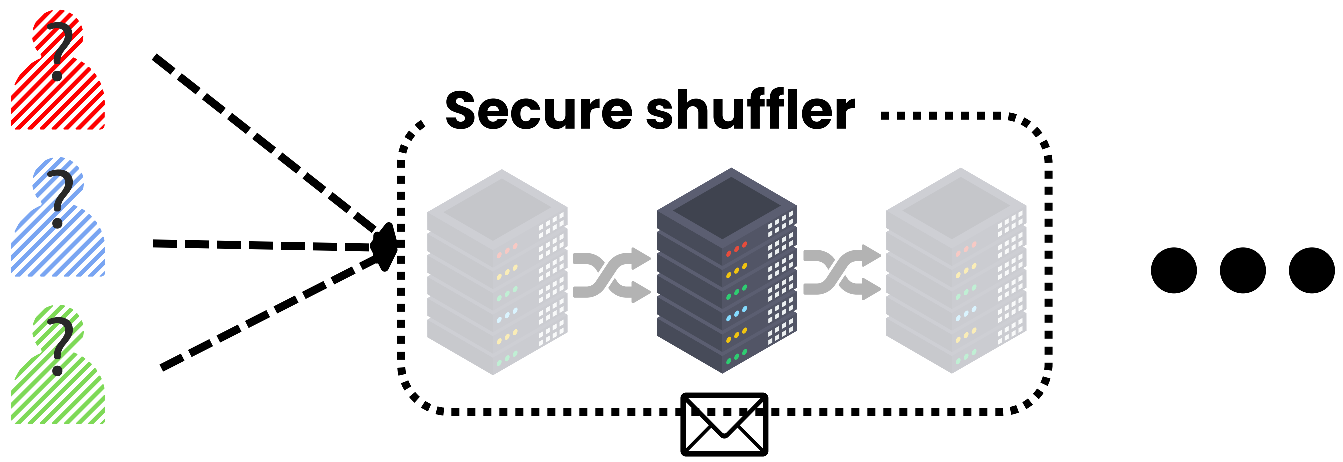
Status Quo: Mix/DC-nets Approaches

Traditional solutions: mix-nets (or DC-nets)



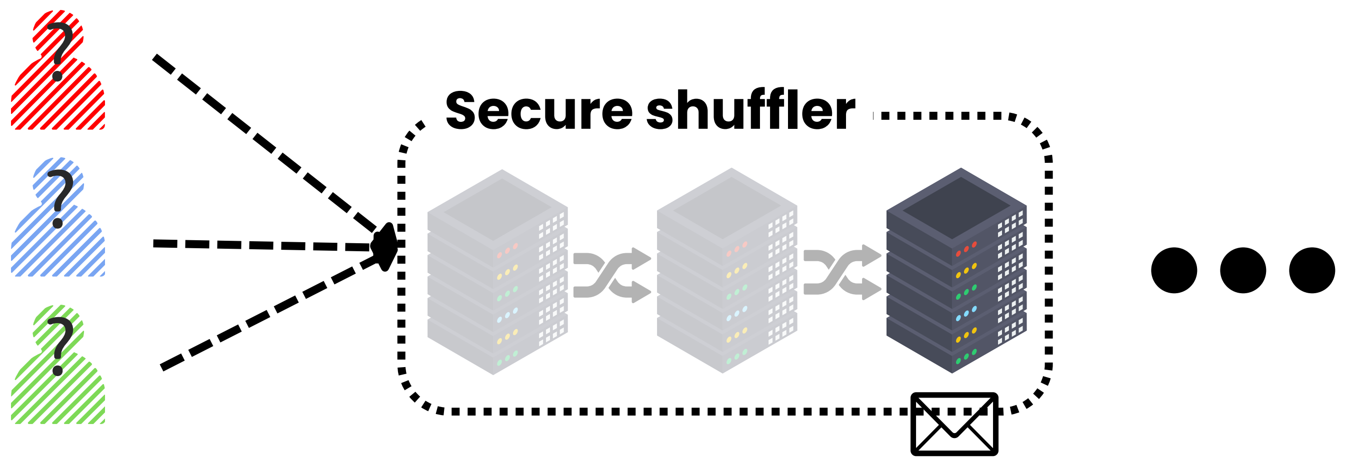
Status Quo: Mix/DC-nets Approaches

Traditional solutions: mix-nets (or DC-nets)



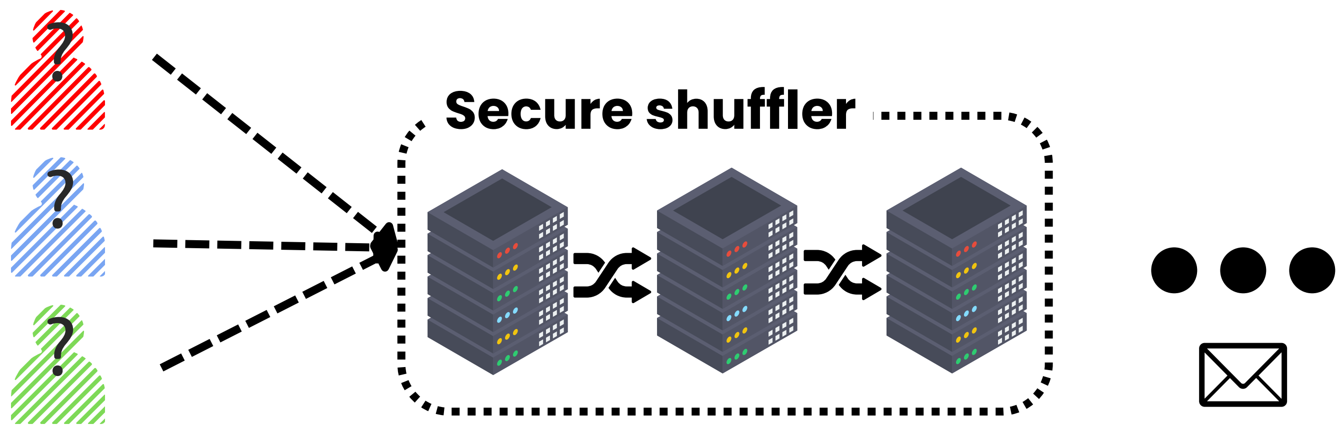
Status Quo: Mix/DC-nets Approaches

Traditional solutions: mix-nets (or DC-nets)



Status Quo: Mix/DC-nets Approaches

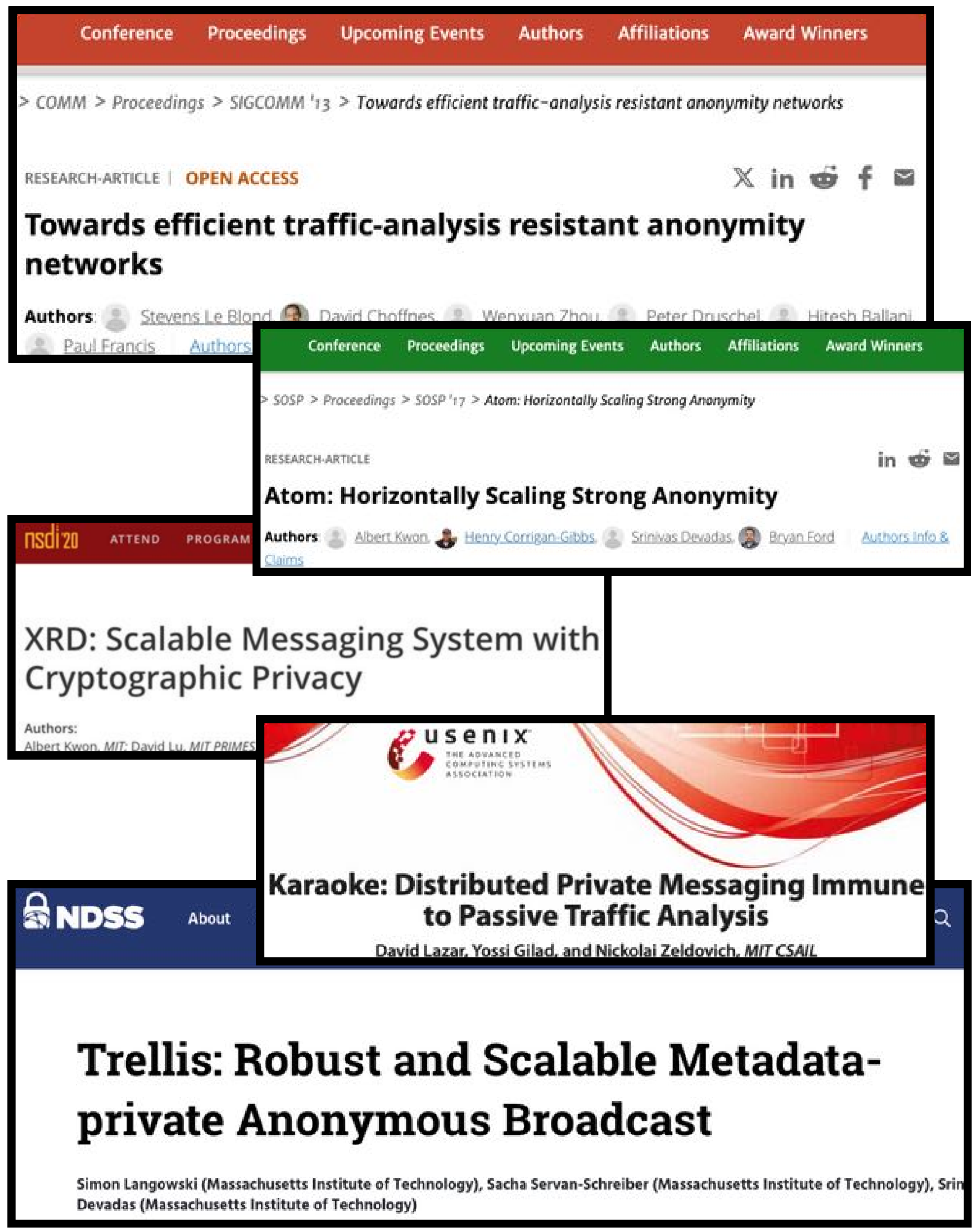
Traditional solutions: mix-nets (or DC-nets)



On the downsides [NDSS'22, Usenix Security'24, ...]:

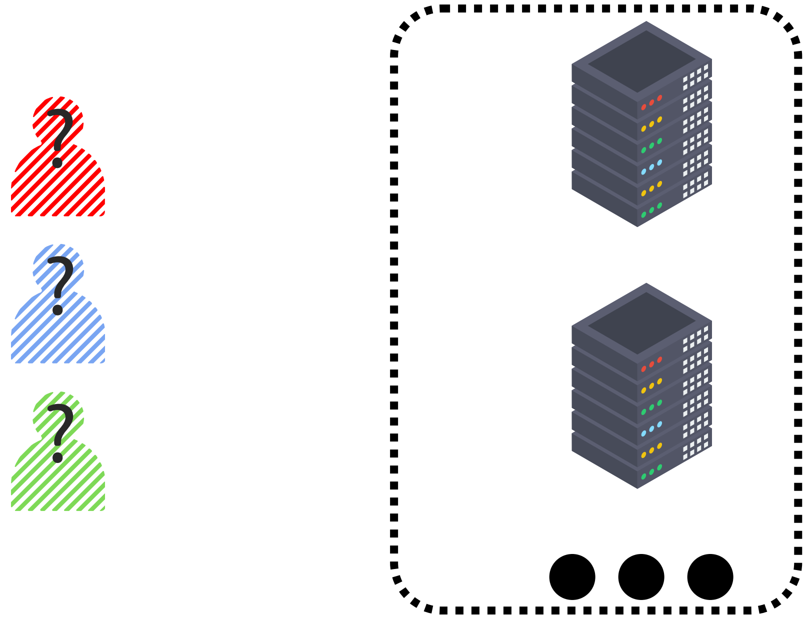
- Heavy public-key cryptoprahy.
- Intensive network hops.
- Subtle security definitions.
- ...

Mix-nets-based anonymous communication systems:



An Alternative: MPC Approaches

Multi-party shuffle (MPS) protocols:

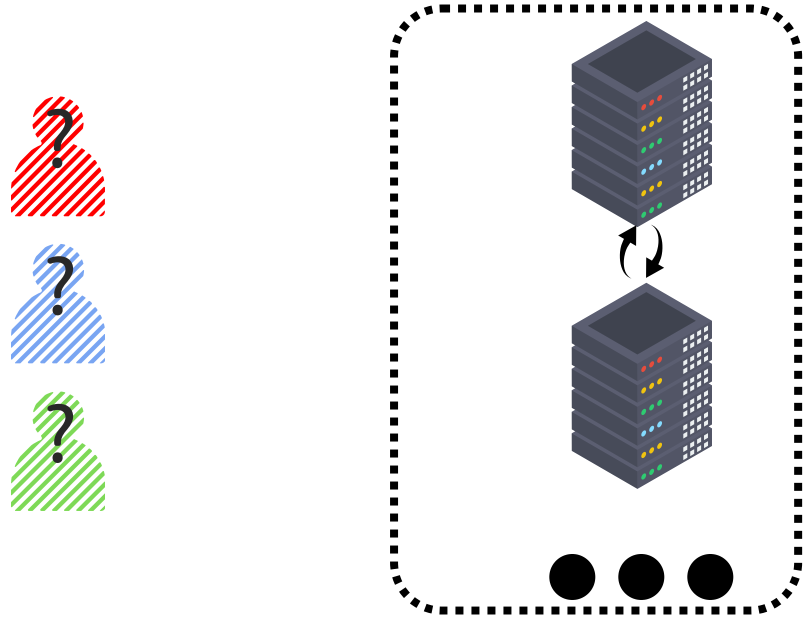


Common assumption:

- Non-colluding servers.

An Alternative: MPC Approaches

Multi-party shuffle (MPS) protocols:



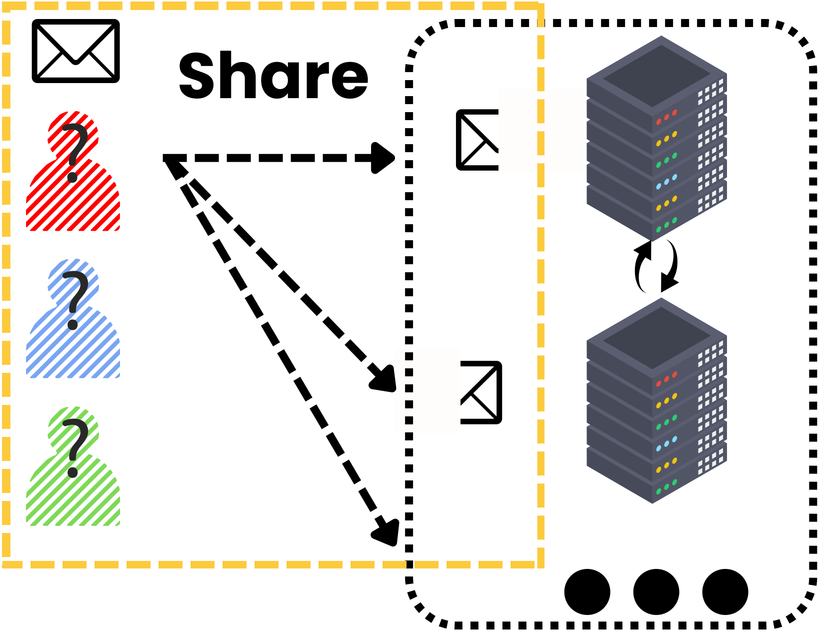
Offline:
pre-compute
shuffle correlation.

Common assumption:

- Non-colluding servers.

An Alternative: MPC Approaches

Multi-party shuffle (MPS) protocols:



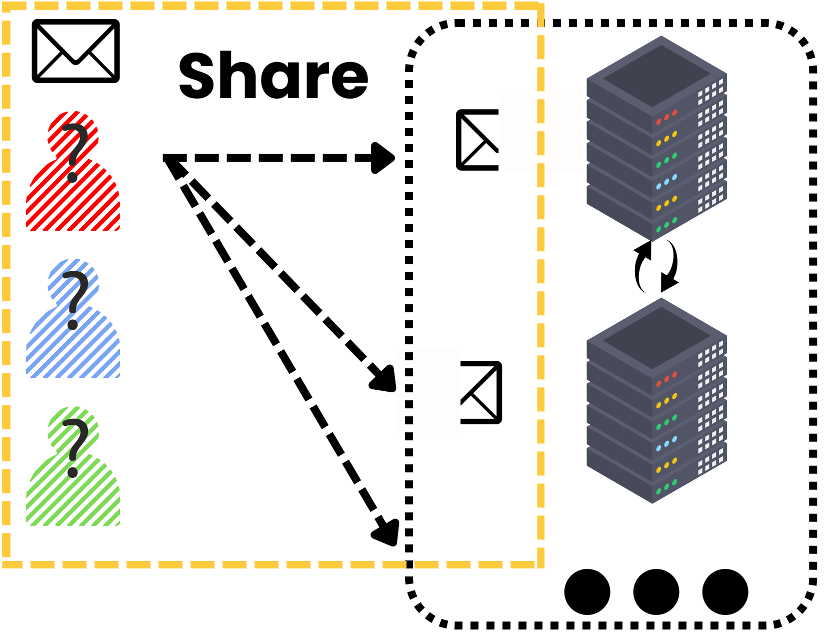
Offline:
pre-compute
shuffle correlation.

Common assumption:

- Non-colluding servers.

An Alternative: MPC Approaches

Multi-party shuffle (MPS) protocols:



Offline:
pre-compute
shuffle correlation.

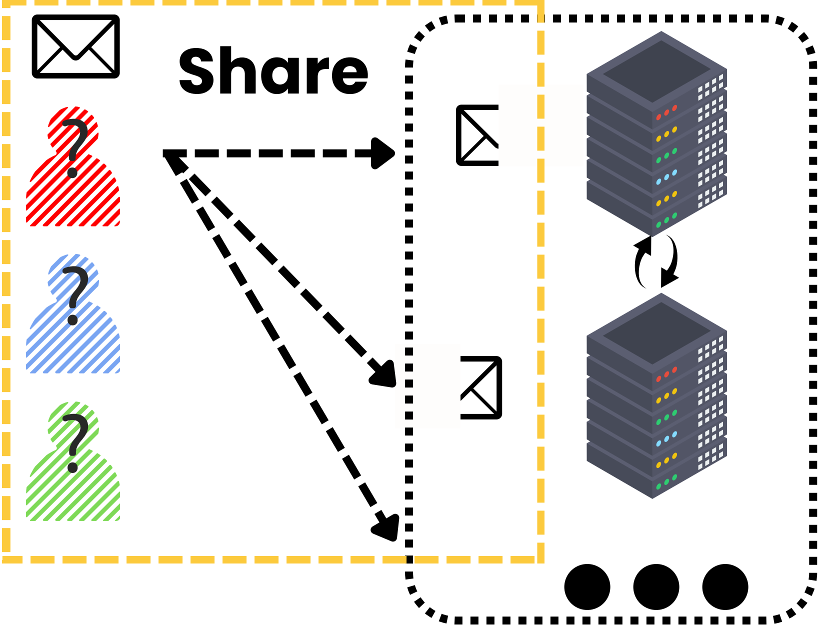
Online:
collaboratively
perform shuffle.

Common assumption:

- Non-colluding servers.

An Alternative: MPC Approaches

Multi-party shuffle (MPS) protocols:



Offline:
pre-compute *shuffle correlation.*

Online:
collaboratively perform shuffle.

Common assumption:

- Non-colluding servers.

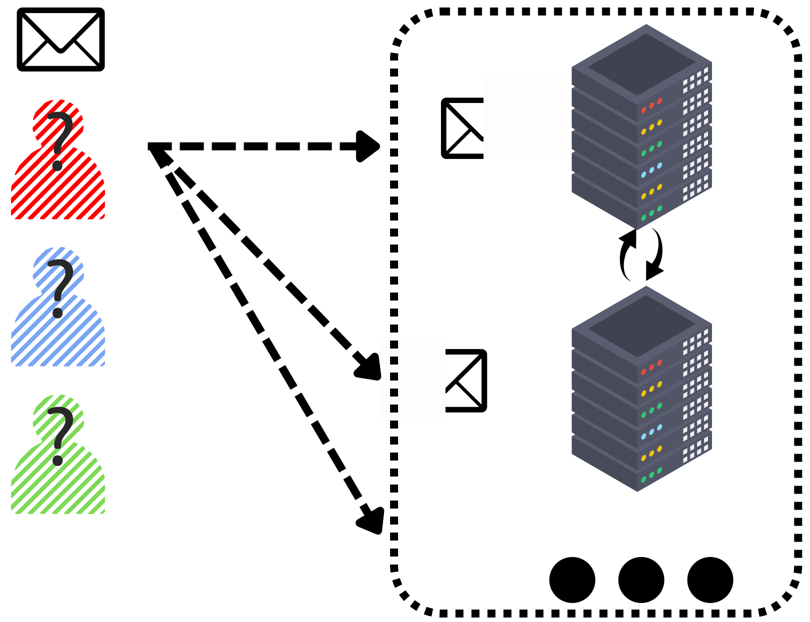
Why MPS?

- Cleaner privacy/security definitions.
- Metadata-private.
- Low computational overhead.
- ...

MPC-based anonymous communication systems:

So Far So Good, But...

Multi-party shuffle (MPS) protocols:



Offline:
pre-compute
shuffle correlation.

Online:
collaboratively
perform shuffle.

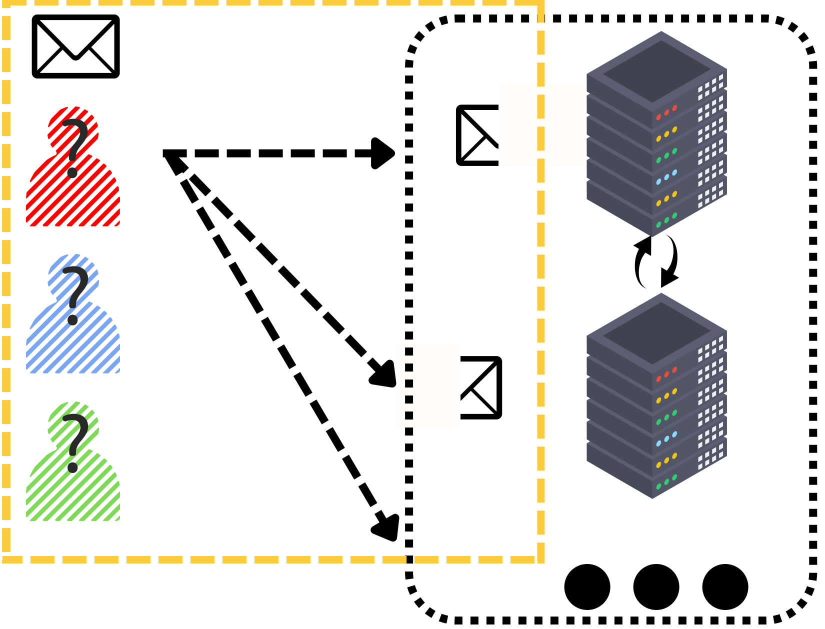
On the downsides: communication remains sub-optimal:

- **CGP paradigm:** constant 2/6 rounds [NDSS'22, NDSS'24].
- **Permutation matrix paradigm:** constant 2/4 rounds [CCS'20, PETs'23].

Ref.	Round	Comm.
AsynchroMix [CCS'19]	$O(\log^2 N)$	$O(N\log^2 N\ell)$
Blinder [CCS'20]	$O(1)$	$O(N\ell)$
Clarion [NDSS'22]	$O(1)$	$O(N\ell)$
RPM [PETs'23]	$O(1)$	$O(N\ell)$

A Starting Point: Permutation Matrix

Multi-party shuffle (MPS) protocols:



Offline:
pre-compute
shuffle correlation.

Online:
collaboratively
perform shuffle.

permutation matrix as shuffle correlation *user input*

$$\mathcal{X}' = \boxed{M} \cdot \boxed{\mathcal{X}}$$

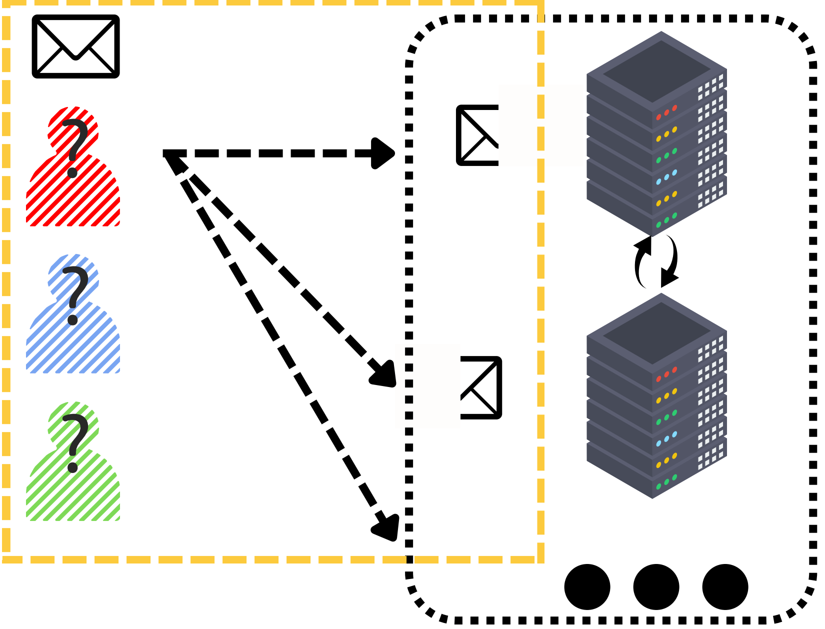
- **Permutation matrix paradigm:** constant 2/4 rounds [CCS'20, PETs'23].

Ref.	Round	Comm.
AsynchroMix [CCS'19]	$O(\log^2 N)$	$O(N\log^2 N\ell)$
Blinder [CCS'20]	$O(1)$	$O(N\ell)$
Clarion [NDSS'22]	$O(1)$	$O(N\ell)$
RPM [PETs'23]	$O(1)$	$O(N\ell)$

**Clarion adopts another CGP paradigm, see details in paper*

Break It Down: Shuffle → Multiplication

Multi-party shuffle (MPS) protocols:



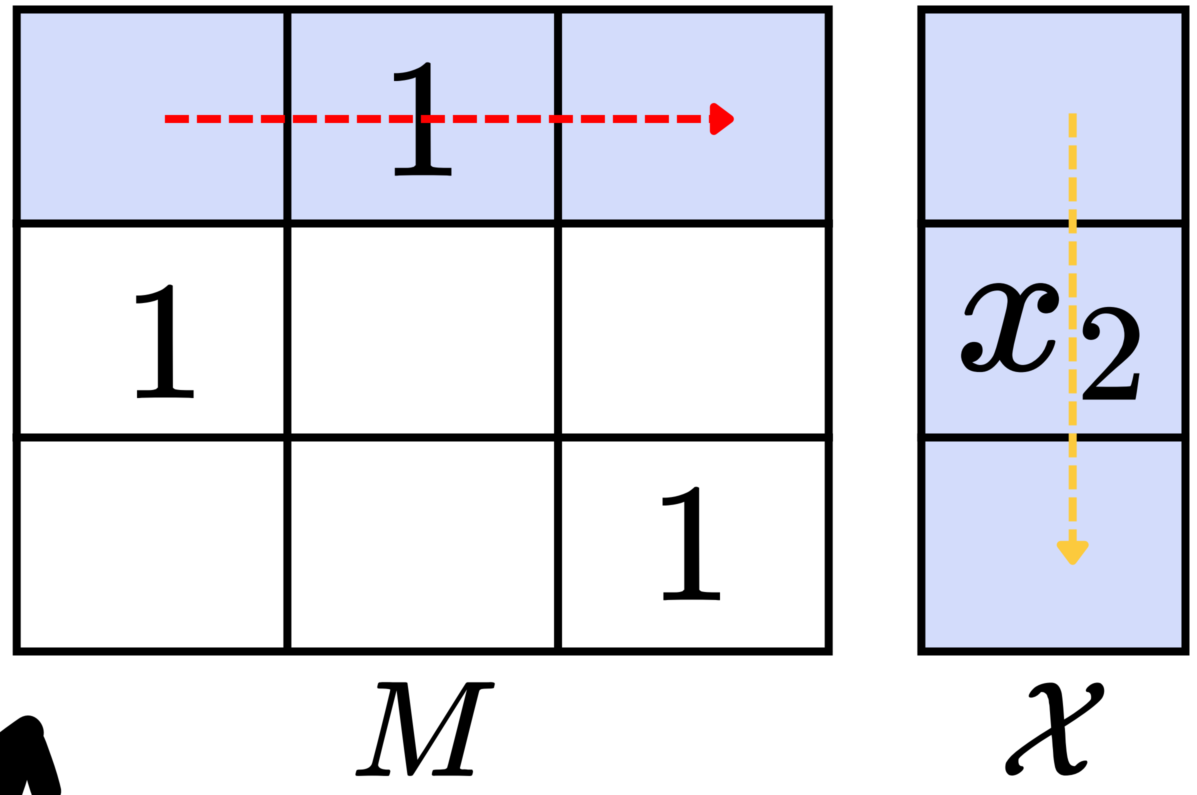
Offline:
pre-compute
shuffle correlation.

Online:
collaboratively
perform shuffle.

permutation matrix as shuffle correlation *user input*

$$\mathcal{X}' = \boxed{M} \cdot \boxed{\mathcal{X}}$$

Shuffle reduces to a secret-shared (matrix-vector) multiplication



$$\sum_{k=1}^{k=N}\left(\sum_{i=1}^{i=n}[x]_i \cdot \sum_{j=1}^{j=n}[m]_j\right)$$

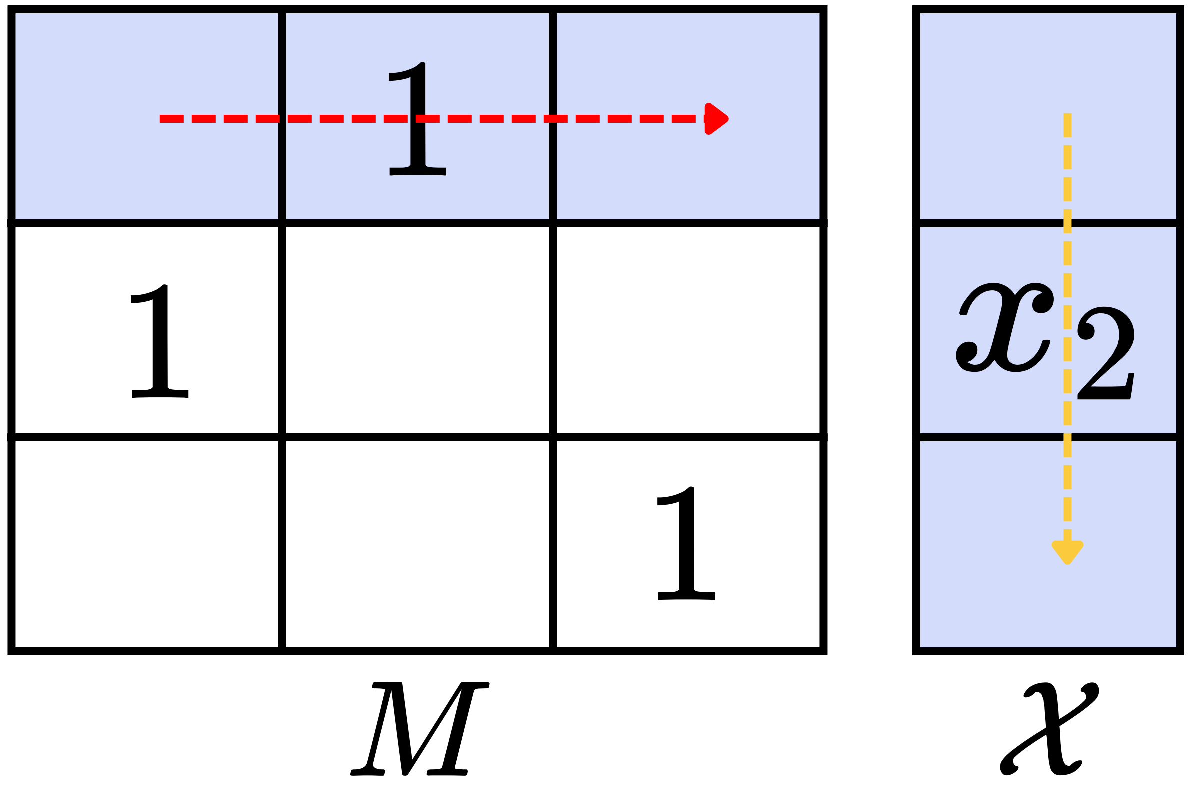
N is the size of message batch;
n is the number of servers; and $m \in M, x \in \mathcal{X}$.

Break It Down: Shuffle → Multiplication

Traditionally, multiplication gate takes
<u>constant interaction round:</u>

- e.g., Matrix triple [Oakland'17].
- e.g., FantasticFour [Usenix Security'21].
- e.g., SWIFT [Usenix Security'21].
- ...

<u>necessary interaction round for degree reduction (or else)</u>



$$\sum_{k=1}^{k=N}\left(\sum_{i=1}^{i=n}[x]_i \cdot \sum_{j=1}^{j=n}[m]_j\right)$$

N is the size of message batch;
n is the number of servers; and $m \in M, x \in \mathcal{X}$.

A Minimal System Setup

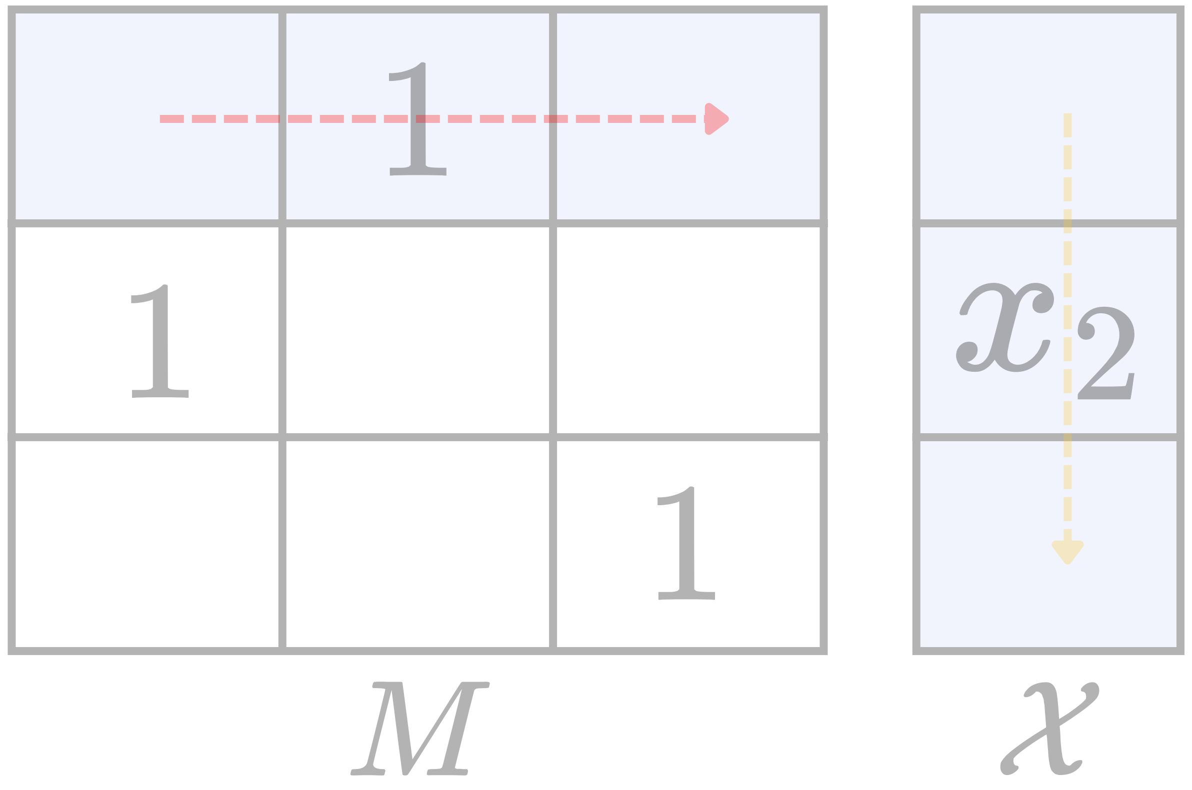
Traditionally, multiplication gate takes <u>constant interaction round:</u>

- e.g., Matrix triple [Oakland'17].
- e.g., FantasticFour [Usenix Security'21].
- e.g., SWIFT [Usenix Security'21].
- ...

<u>necessary interaction round for degree reduction (or else)</u>

To explore optimal communication boundaries, we consider an *<u>honest-majority</u>*, *<u>small-party</u> setup, over the *<u>finite ring,</u>* under the *<u>preprocessing model</u>*: i.e., a minimal 4-party setting.

- Also for efficiency, robustness, and deployment [Usenix Security'21, PETs'23, NDSS'22]

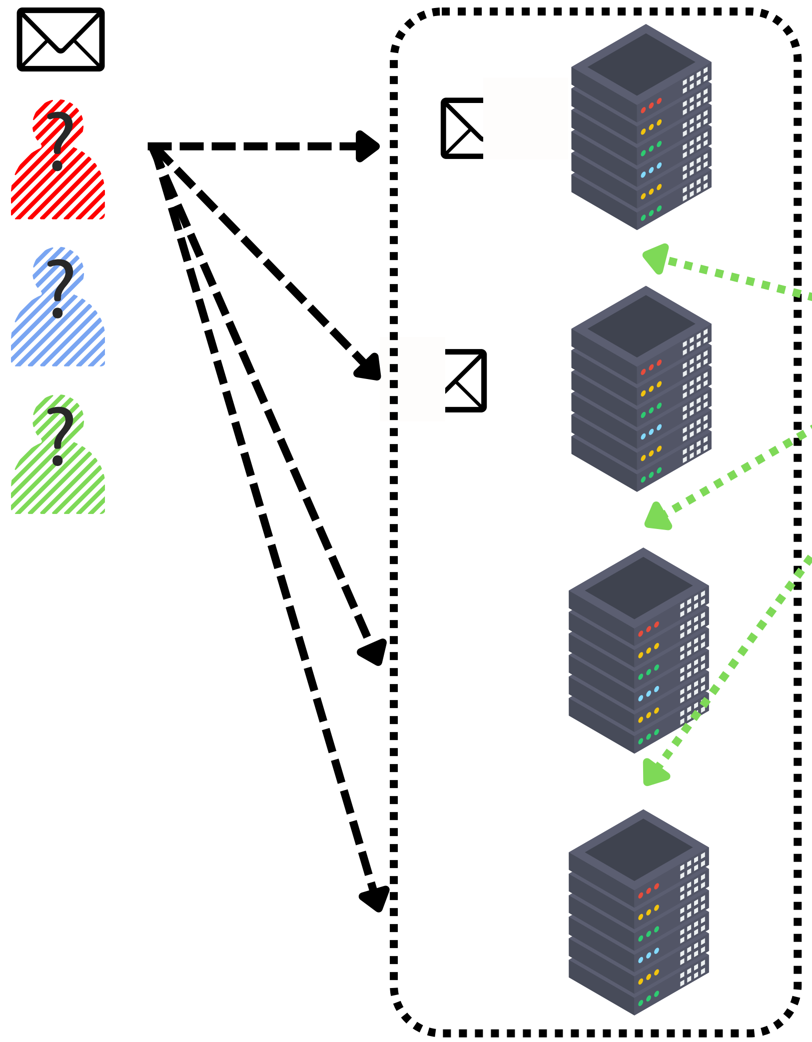


$$\sum_{k=1}^{k=N}\left(\sum_{i=1}^{i=n}[x]_i \cdot \sum_{j=1}^{j=n}[m]_j\right)$$

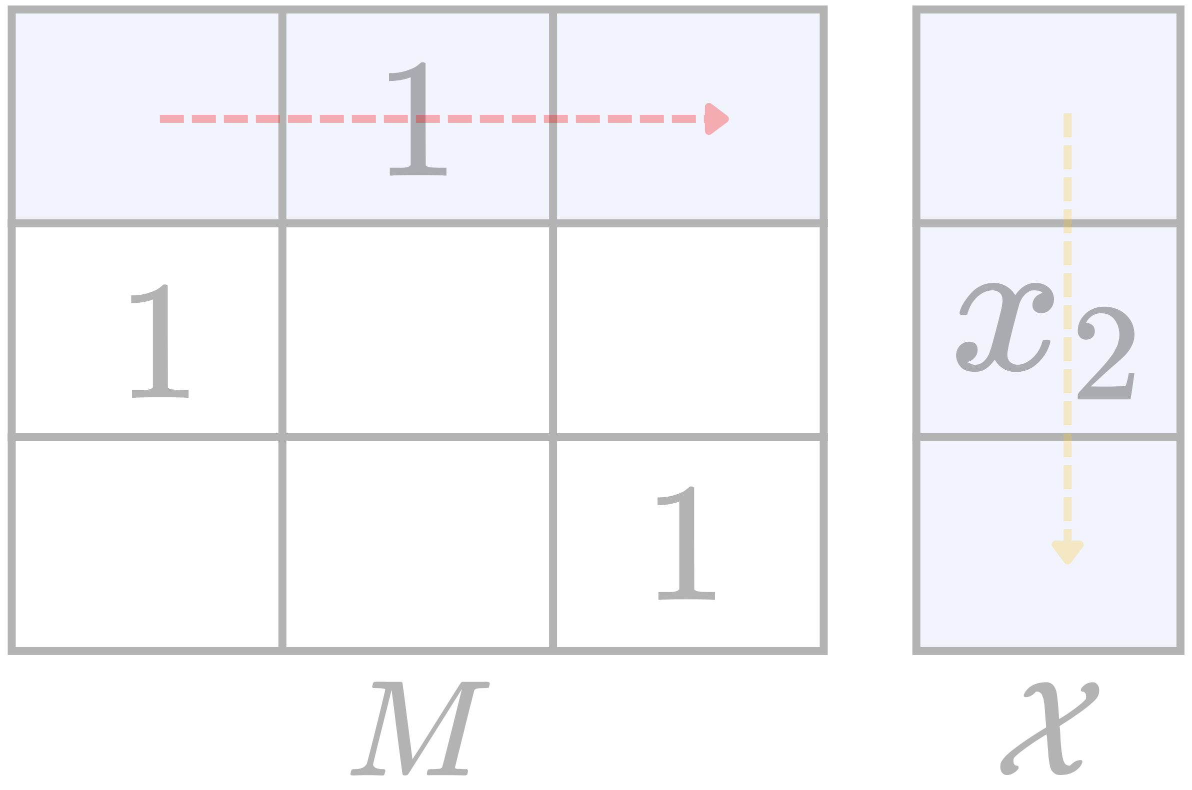
N is the size of message batch;
n is the number of servers; and $m \in M, x \in \mathcal{X}$.

Towards Non-Interactive, Silent Shuffle

A minimal 4-server example:



Goal: make it non-interactive

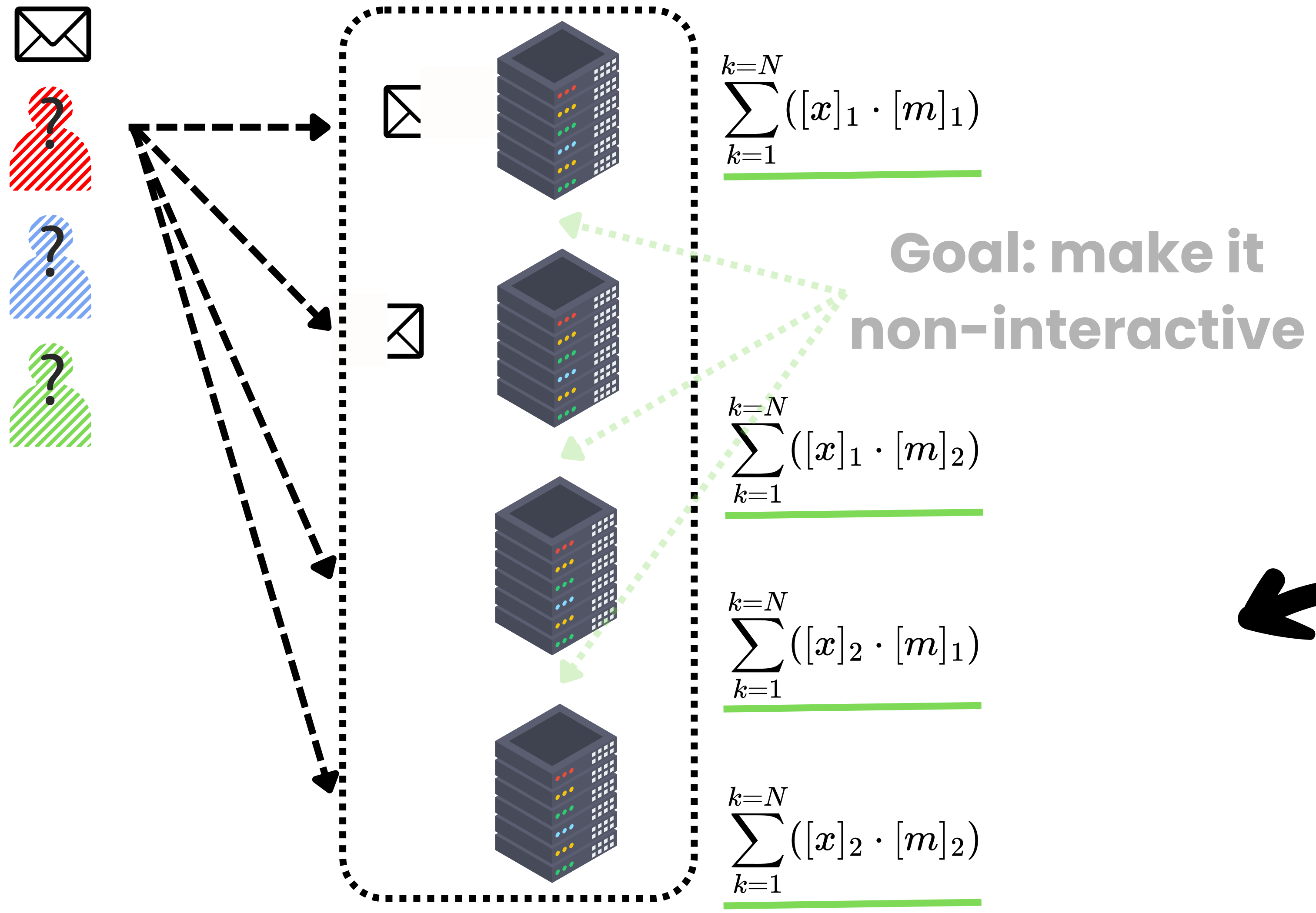


$$\sum_{k=1}^{k=N}\left(\sum_{i=1}^{i=n}[x]_i \cdot \sum_{j=1}^{j=n}[m]_j\right)$$

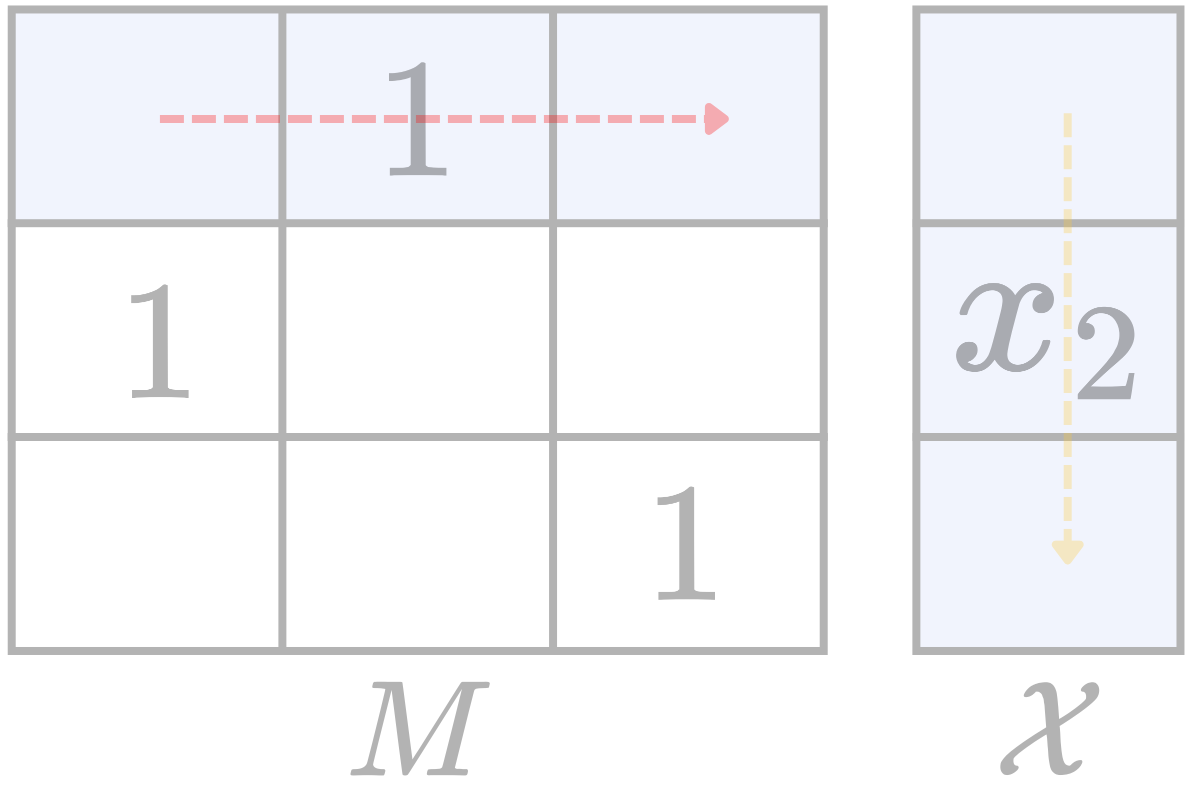
N is the size of message batch;
n is the number of servers; and $m \in M, x \in \mathcal{X}$.

Towards Non-Interactive, Silent Shuffle

A minimal 4-server example:



Locally multiply the distributed message shares and permutation matrix shares

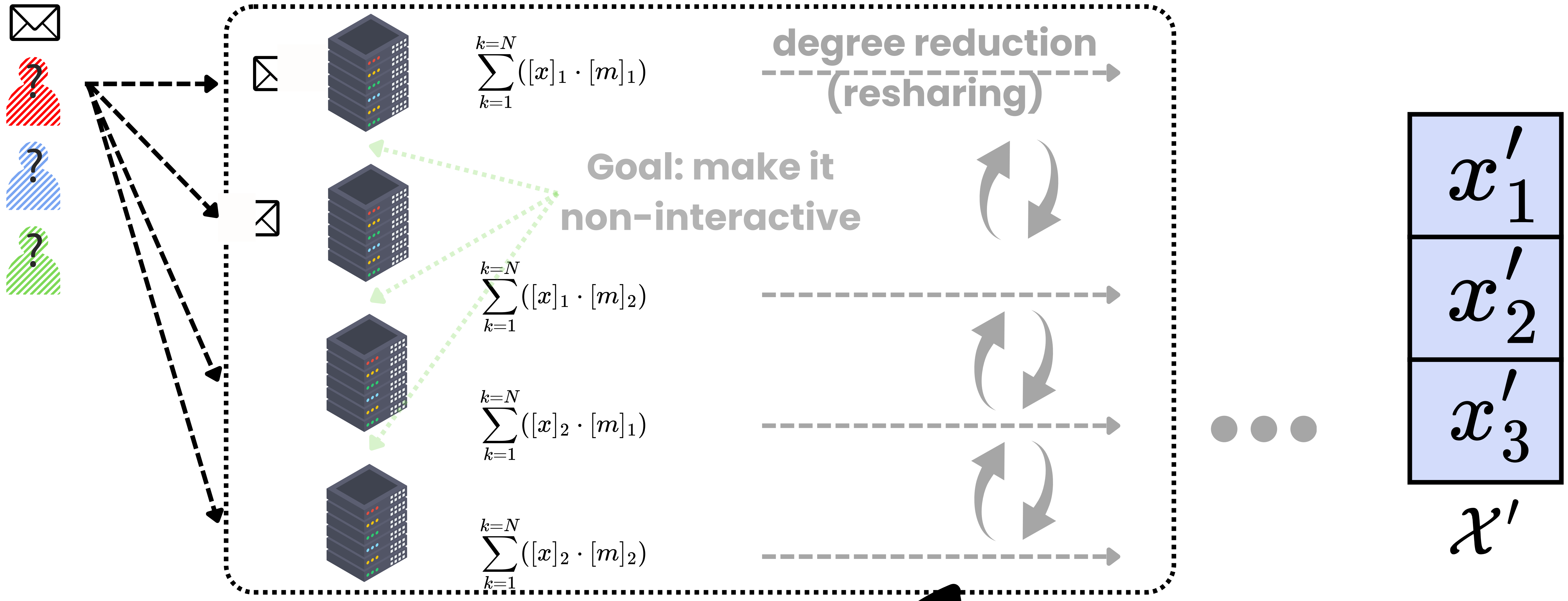


$$\sum_{k=1}^{k=N}\left(\sum_{i=1}^{i=n}[x]_i \cdot \sum_{j=1}^{j=n}[m]_j\right)$$

N is the size of message batch;
n is the number of servers; and $m \in M, x \in \mathcal{X}$.

Towards Non-Interactive, Silent Shuffle

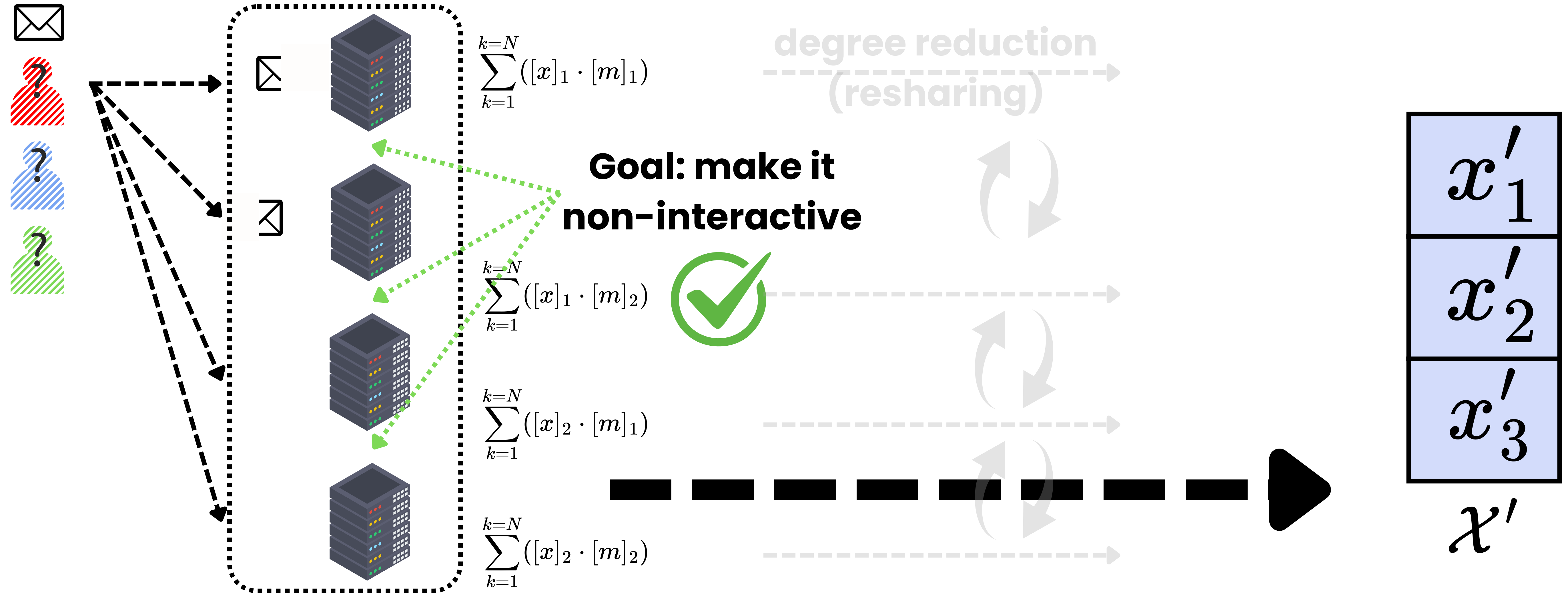
A minimal 4-server example:



Typically, one more sharing round to perform the degree reduction (or else)

Towards Non-Interactive, Silent Shuffle

A minimal 4-server example:

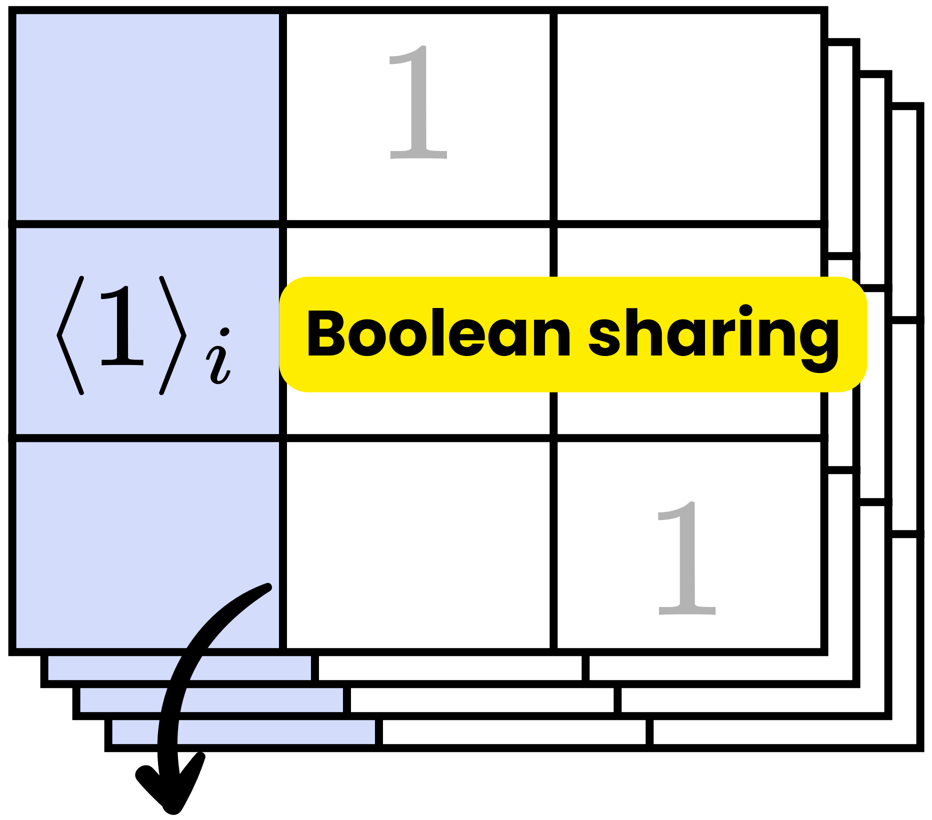


Directly combine all sub-terms after local computations to perform shuffle

Key observation: we need standalone shuffle in boradcast services.

Sparsity-Aware Optimization

Key observation: permutation matrix is sparse, containing only 0 s and 1 s, thus can be aptly encoded with Boolean share.

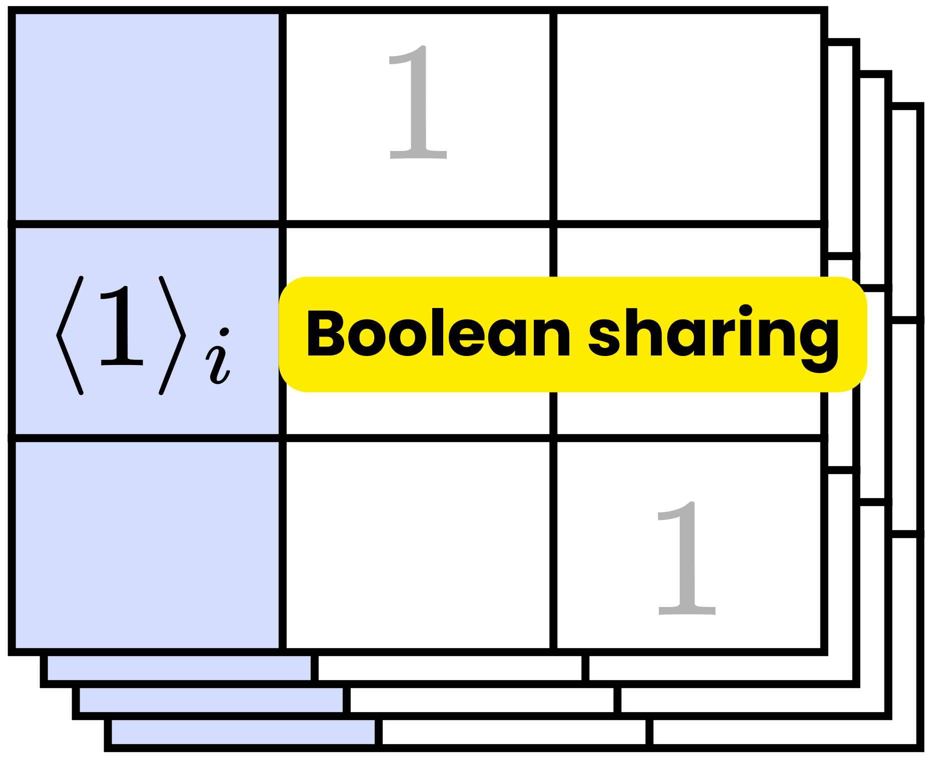


Boolean permutation correlation (BPC)

- Compress communication
- Accelerate computation
- …

Sparsity-Aware Optimization

Key observation: permutation matrix is sparse, containing only 0 s and 1 s, thus can be aptly encoded with Boolean share.



A DPF key tuple encodes a one-hot vector, i.e., one BPC column.

How to generate BPC:

4-server 1-bit DPF

(1) DPF. Gen()

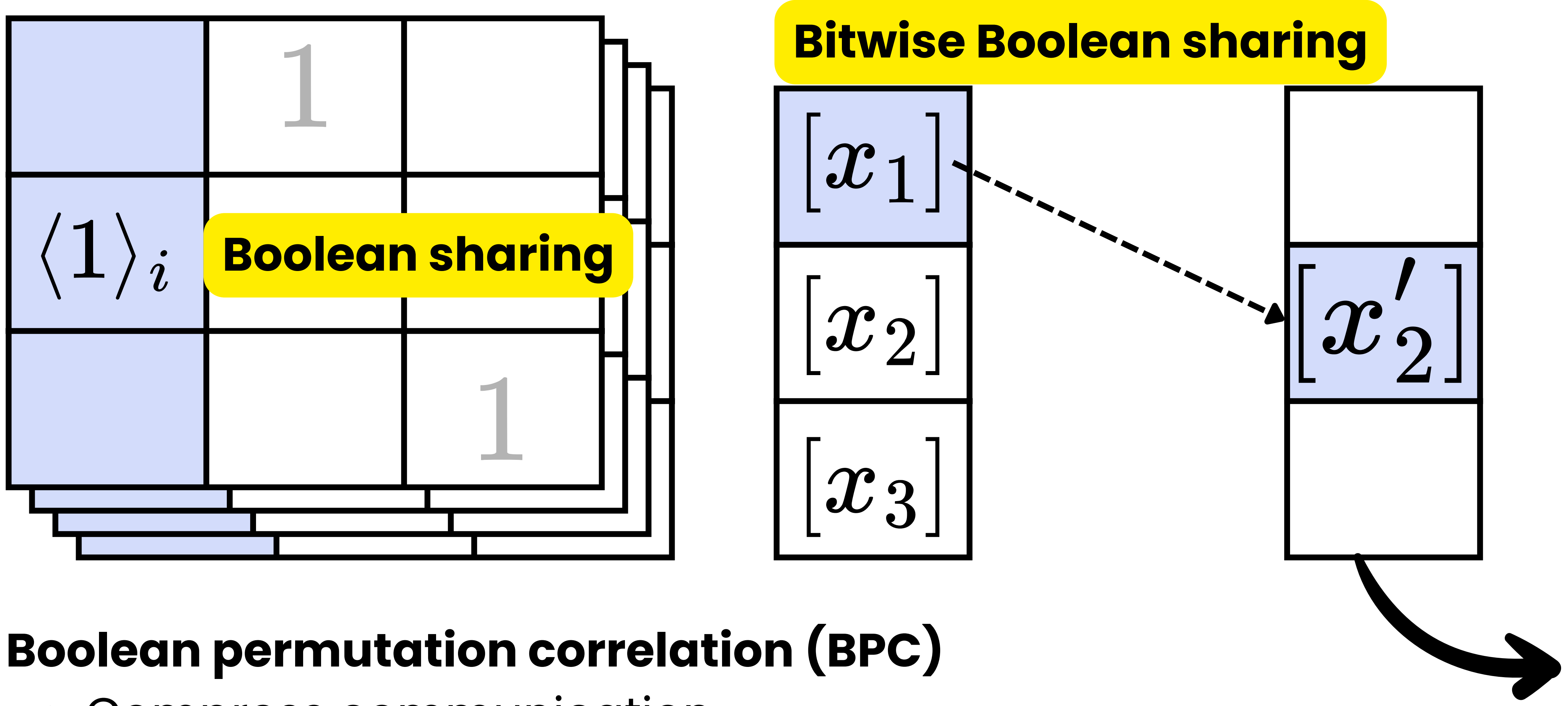
(2) DPF. EvalAll()

Boolean permutation correlation (BPC)

- Compress communication
- Accelerate computation
- ...

Sparsity-Aware Optimization

Key observation: permutation matrix is sparse, containing only 0 s and 1 s, thus can be aptly encoded with Boolean share.



How to generate BPC:

4-server 1-bit DPF

(1) DPF. Gen()

(2) DPF. EvalAll()

How to perform shuffle with BPC:

$$\bigoplus_{k=1}^{k=N}\left(\bigoplus_{i=1}^{i=n}[x]_i \cdot \bigoplus_{j=1}^{j=n}\langle m\rangle_j\right)$$

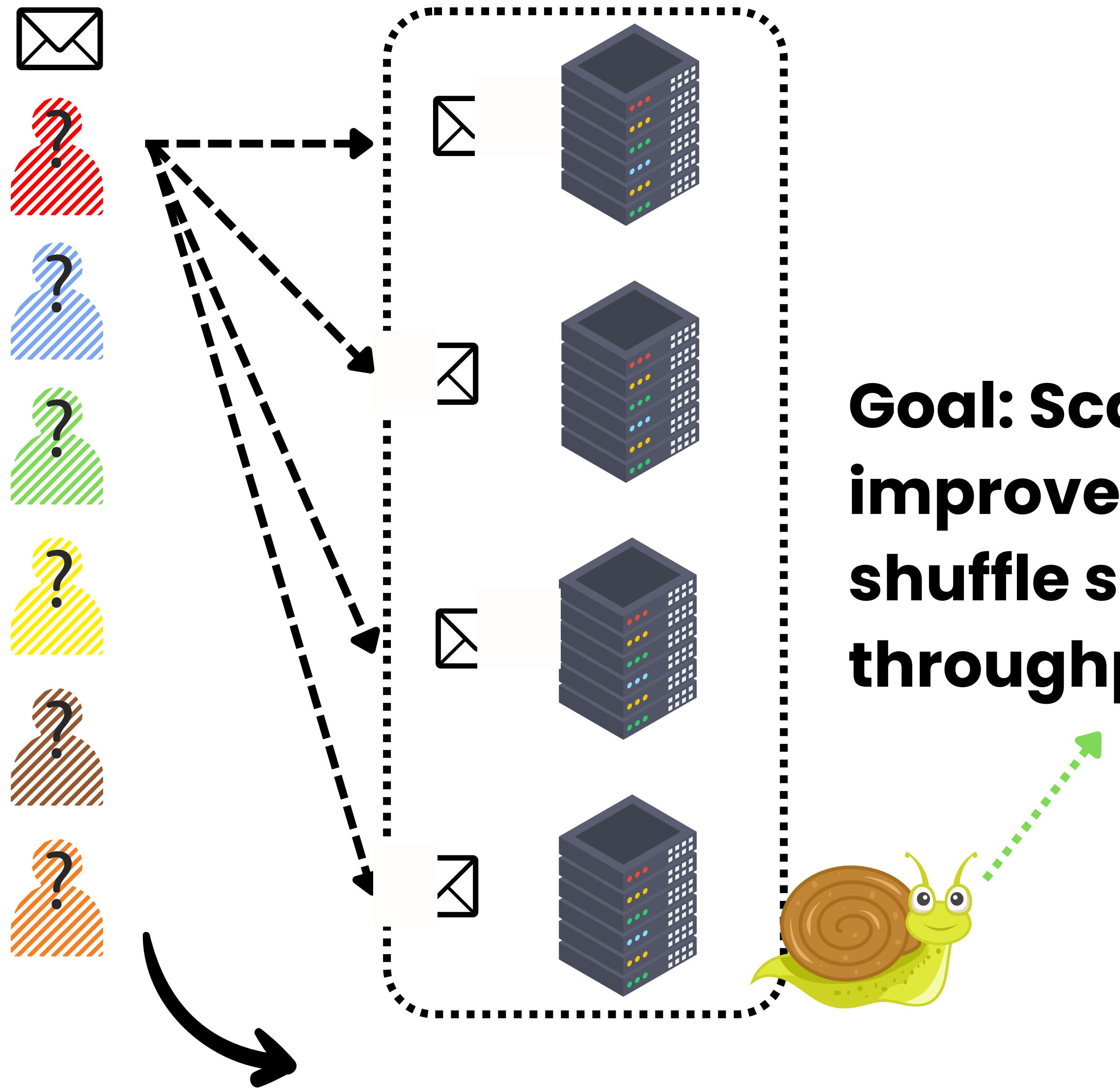
Here we slightly abuse the XOR notation, the arithmetic sharing semantic, and the "multiplication" dot.

Boolean permutation correlation (BPC)

- Compress communication
- Accelerate computation
- …

Anonymity Loves Company, But...

A minimal 4-server example:

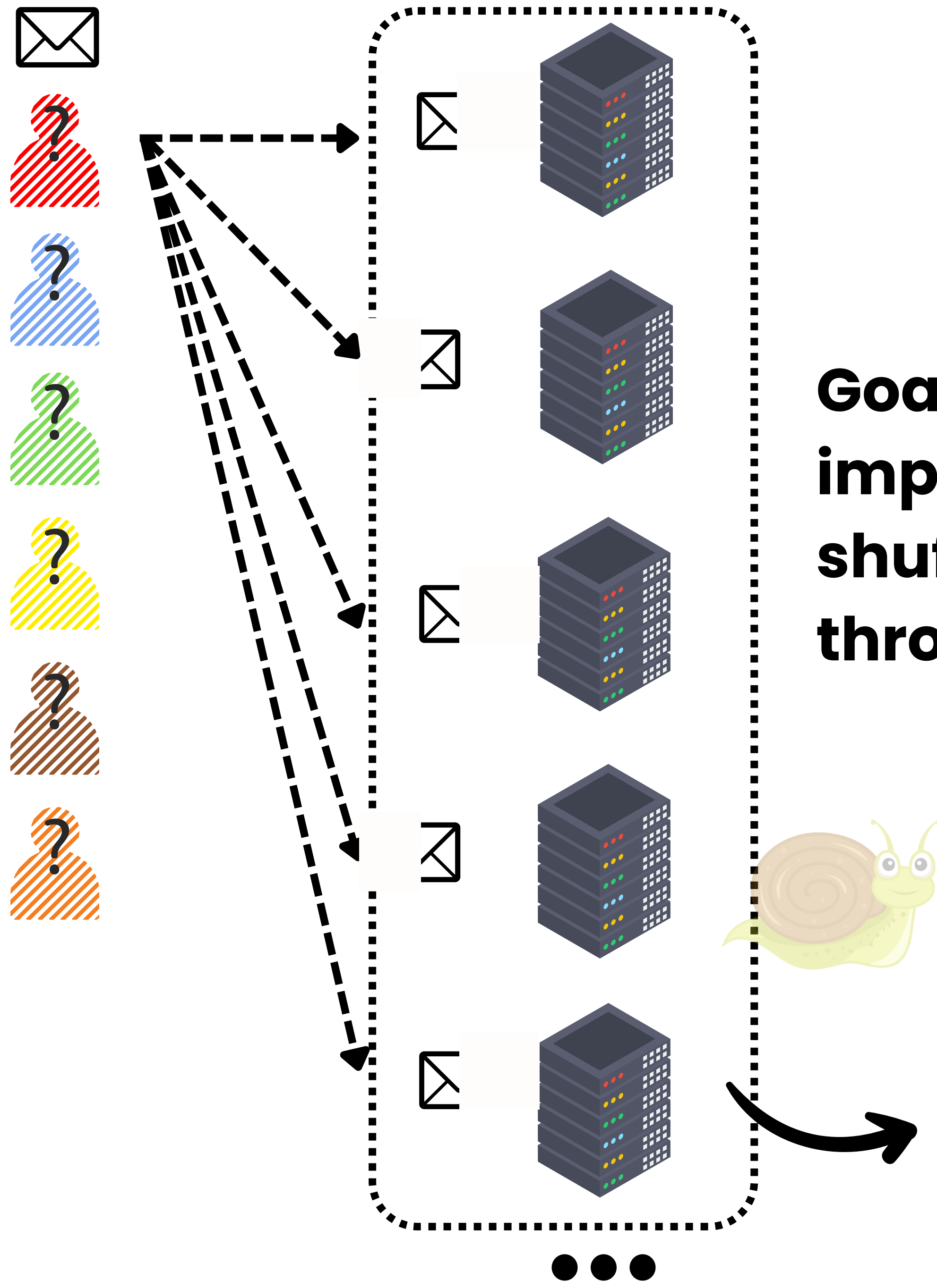


Goal: Scaling to improve the shuffle service throughput

The larger anonymity set ensures stronger privacy, yet incurs higher overhead.

Anonymity Loves Company, But...

A traditional MPC-style scaling example:

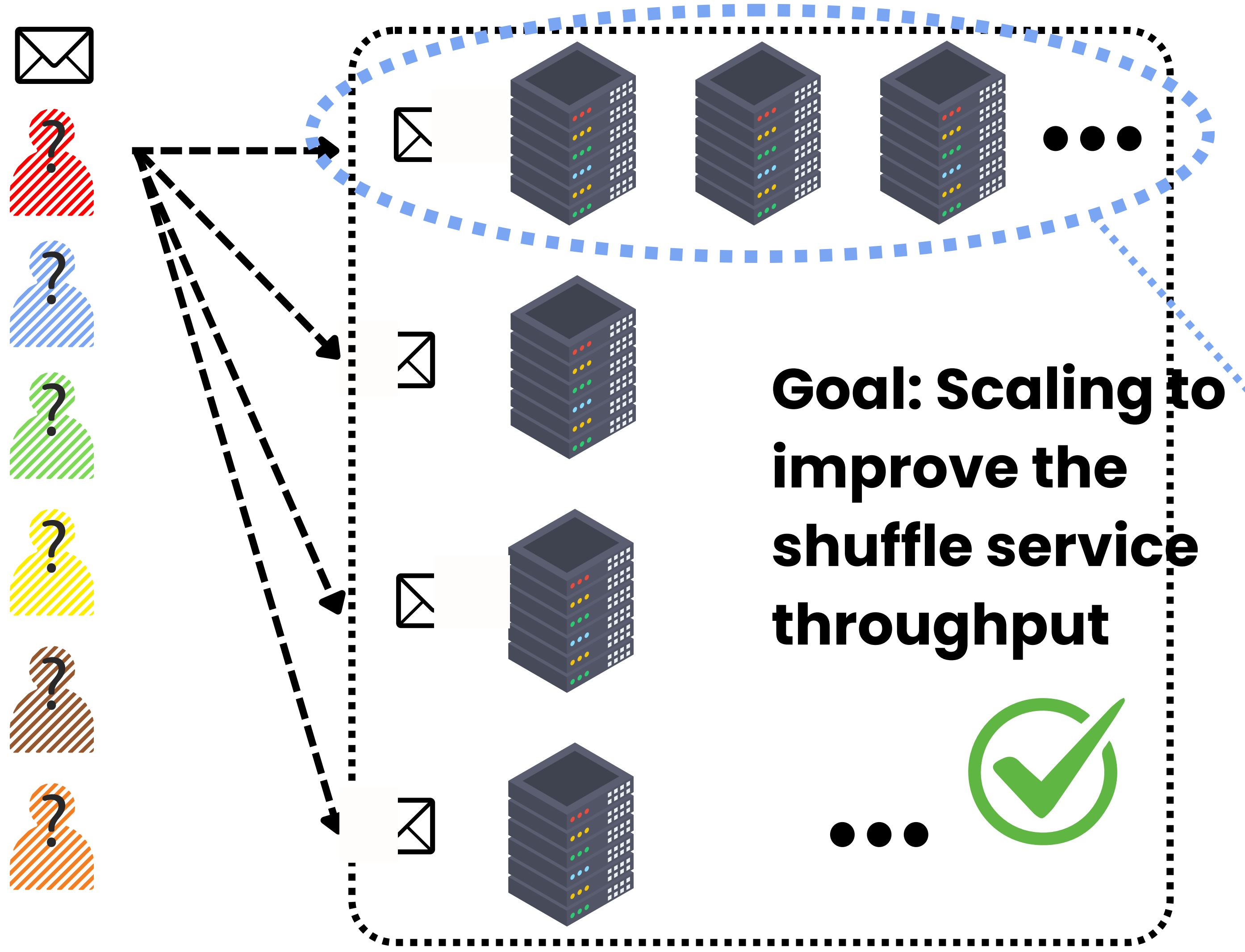


Goal: Scaling to improve the shuffle service throughput

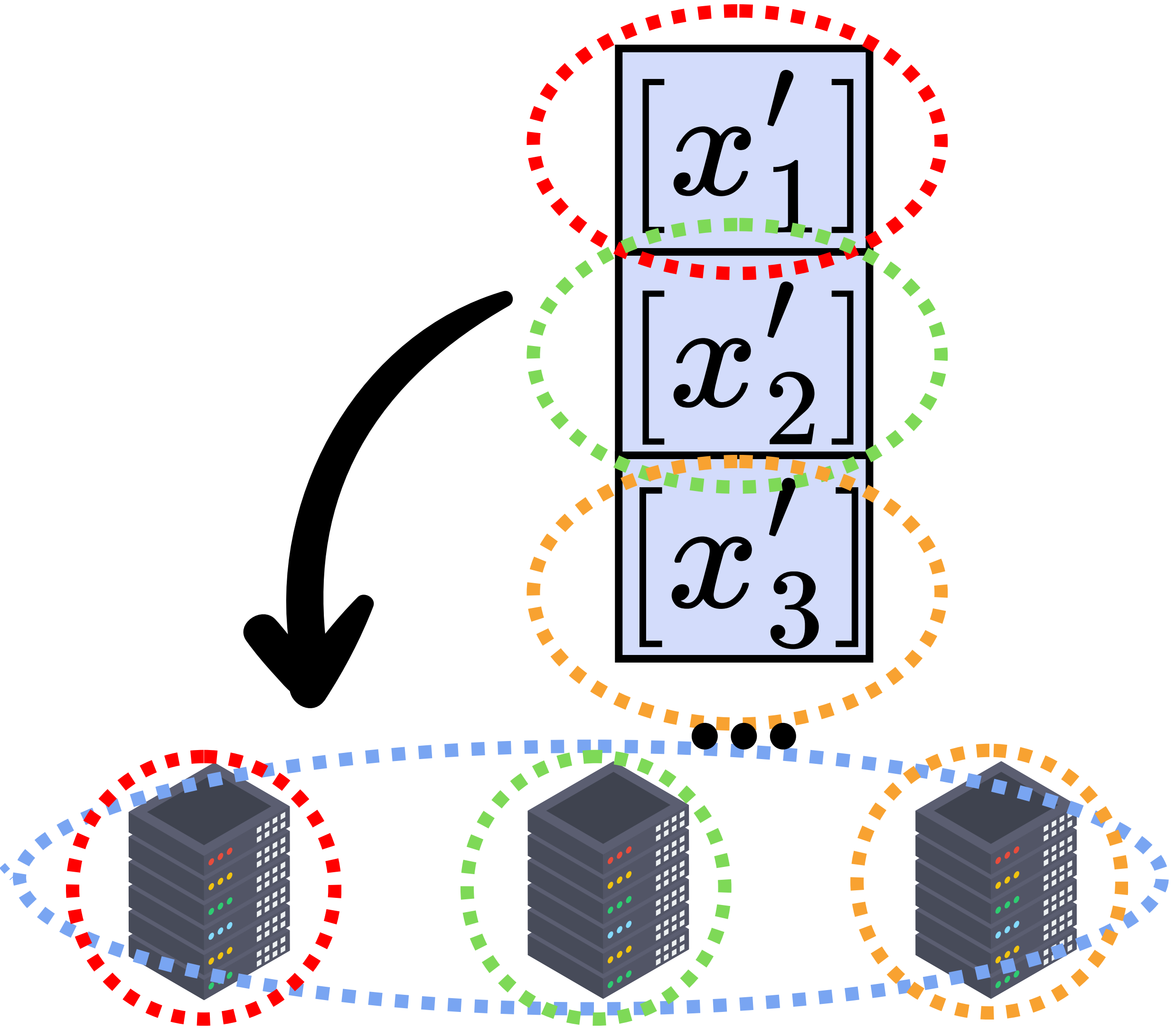
e.g., Shamir sharing, m-party additive sharing.

Horizontal Scaling for A Larger User Base

A minimal 4-party <u>horizontal scaling</u> example:



Divide, compute, and aggregate their local computations

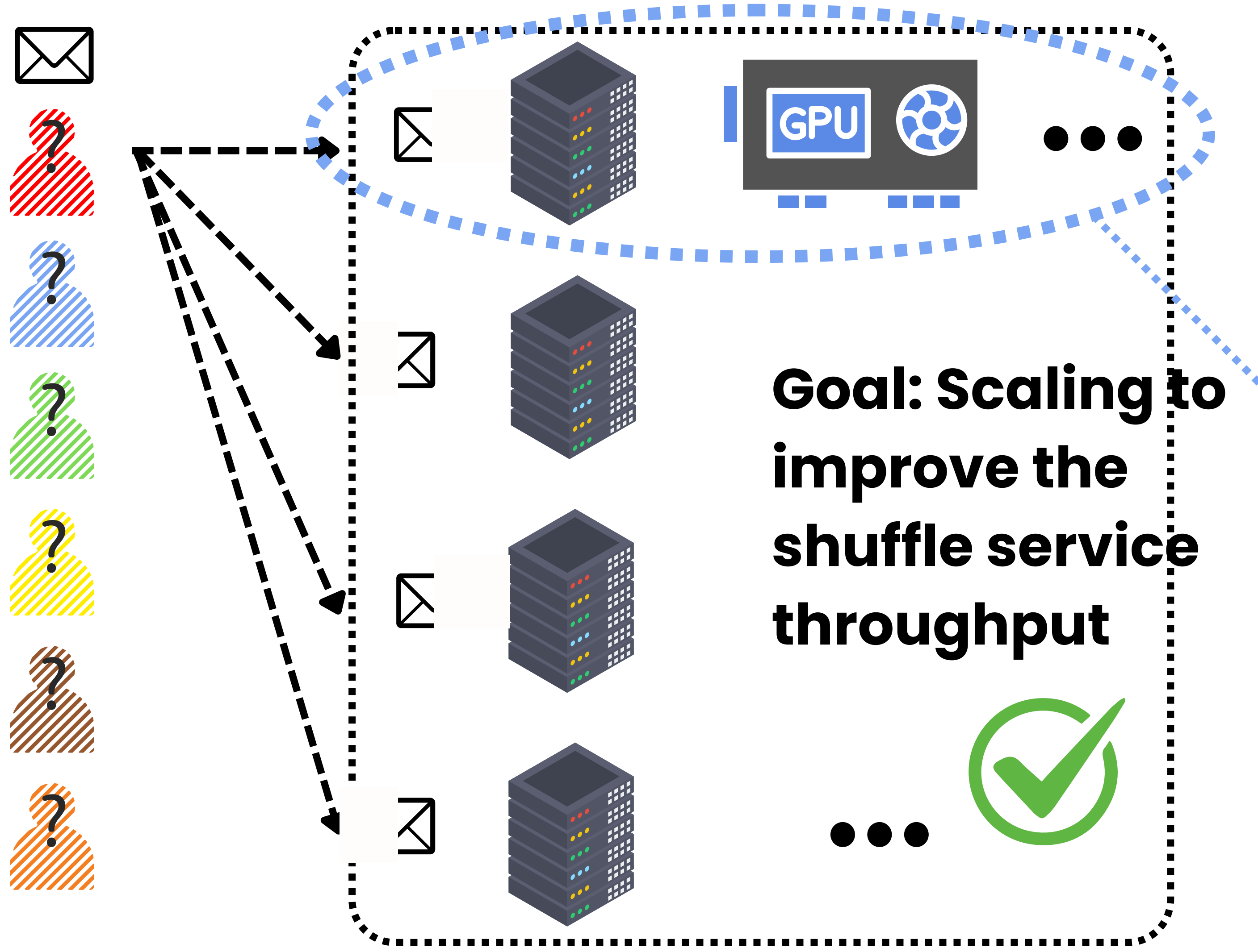


Non-interactive, parallelization-friendly

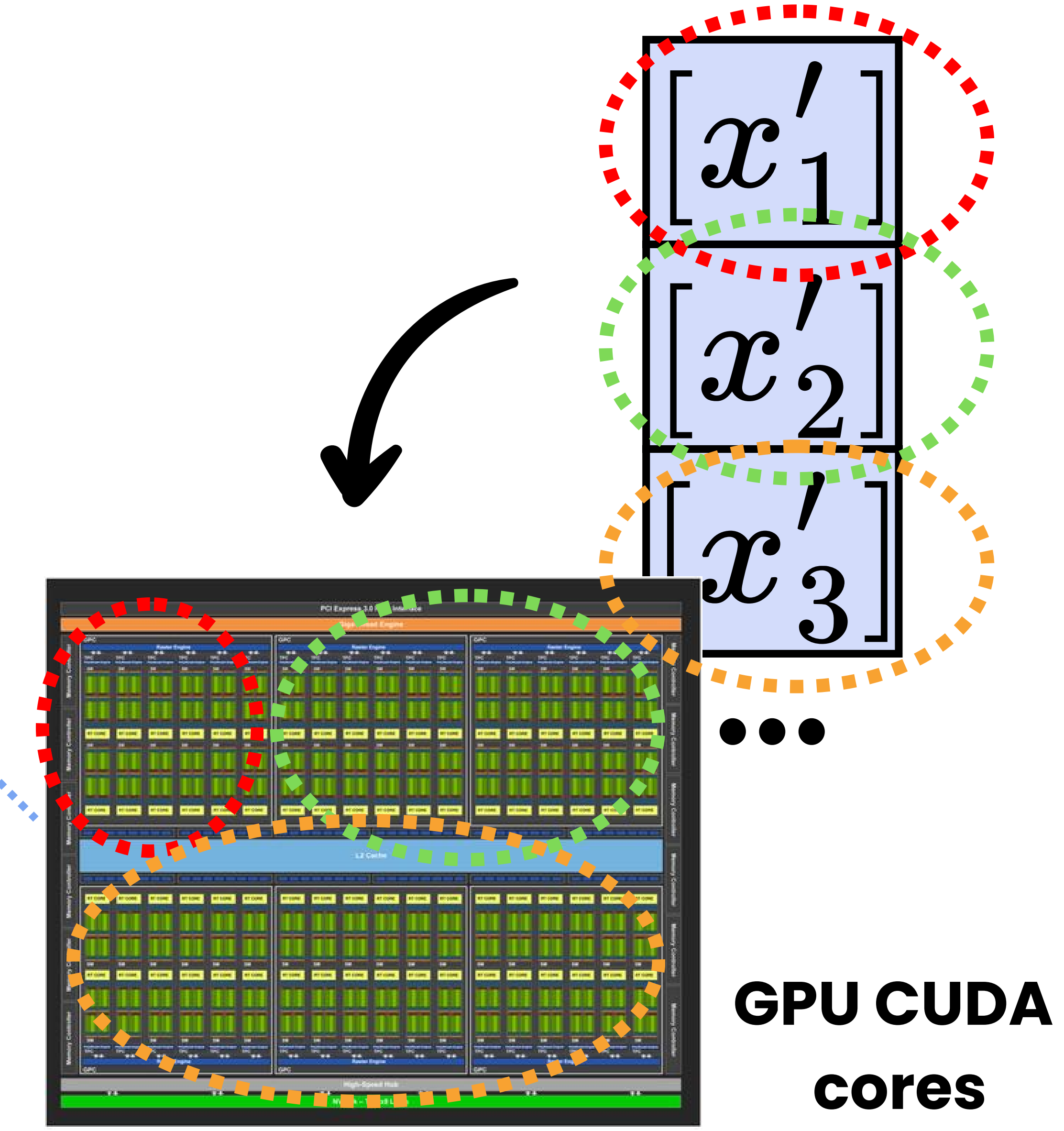
Improve message shuffling throughput by adding more servers

Vertical Scaling for A Larger User Base

A minimal 4-party vertical scaling example:



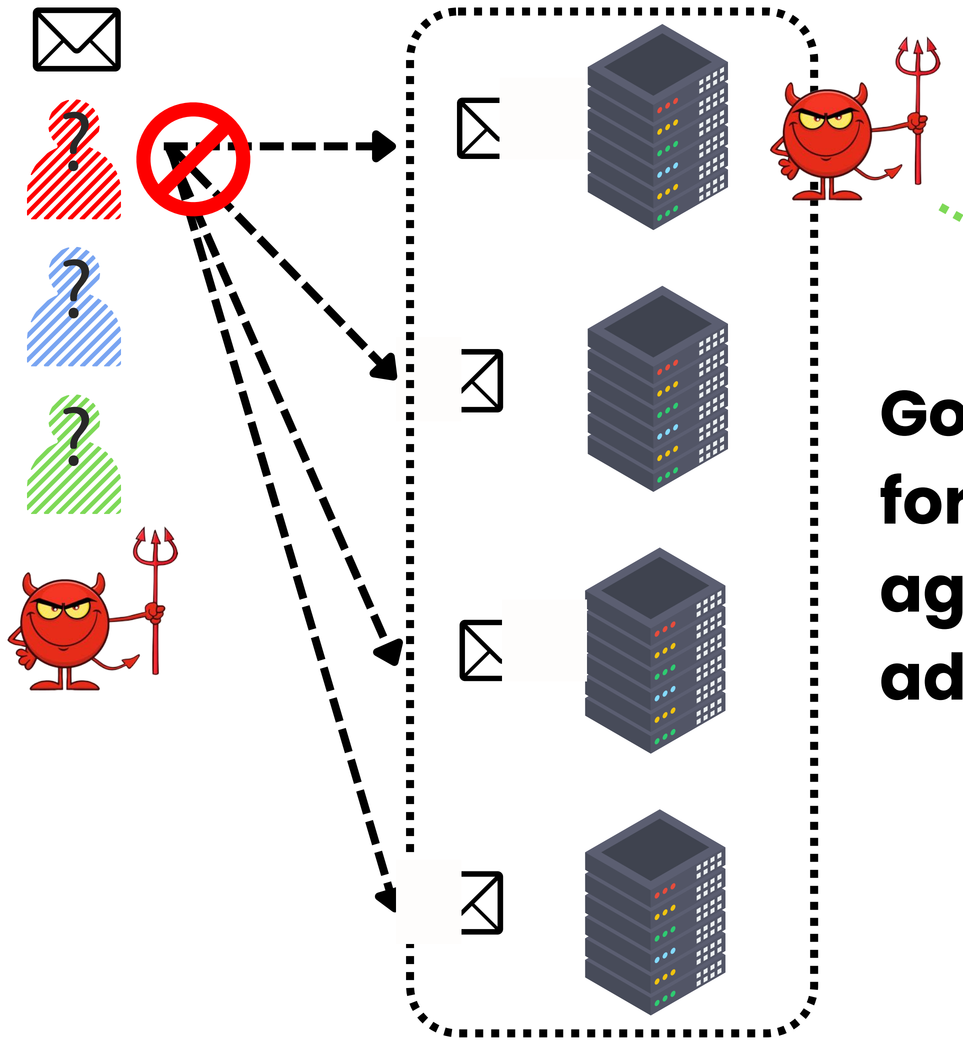
Divide, compute, and aggregate their local computations



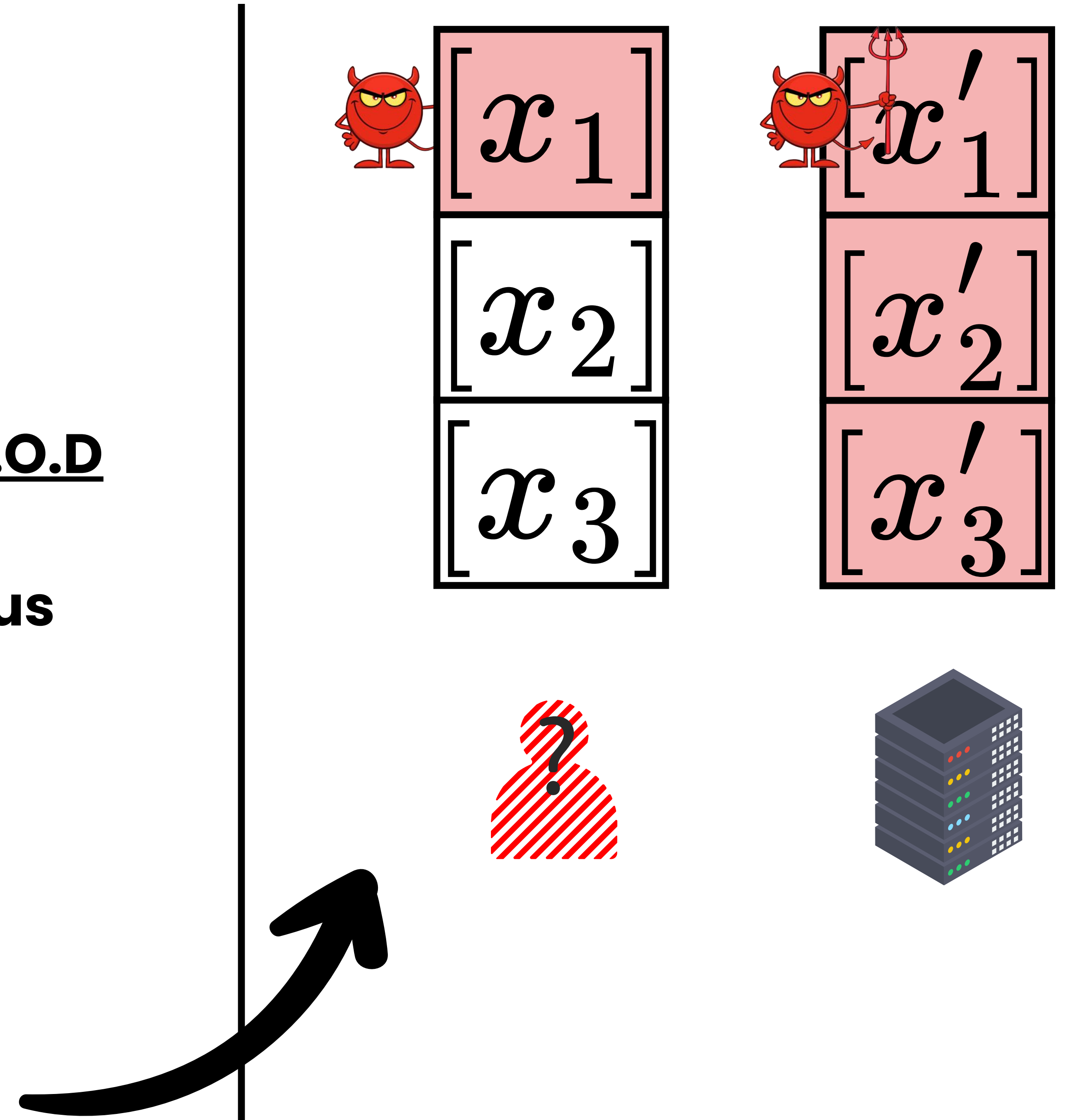
Improve message shuffling throughput by using more powerful devices

Also, Anonymity Loves Robustness

Selective failure attacks:



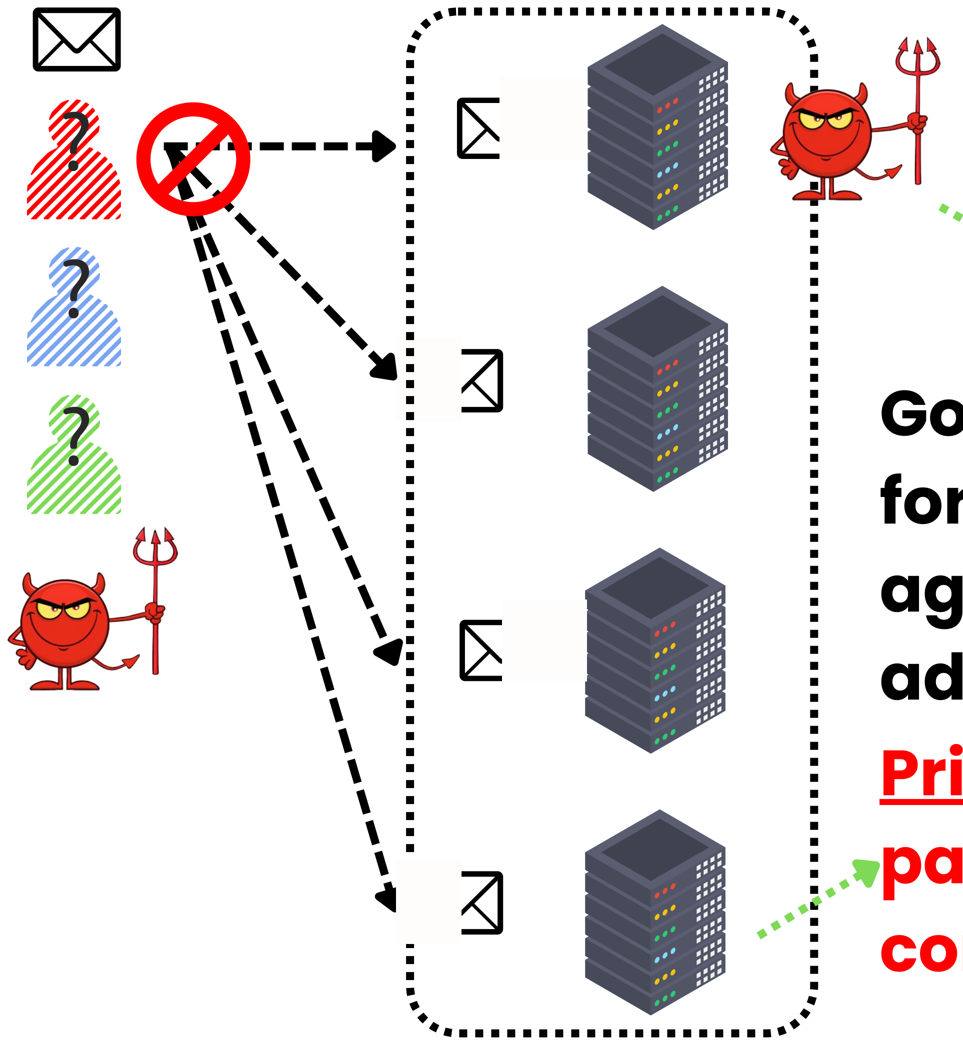
Goal: <u>efficient G.O.D</u> for honest users against malicious adversaries



Adversary (either server or user) attempt to launch selective failure attacks

A Step Further: Private Robustness

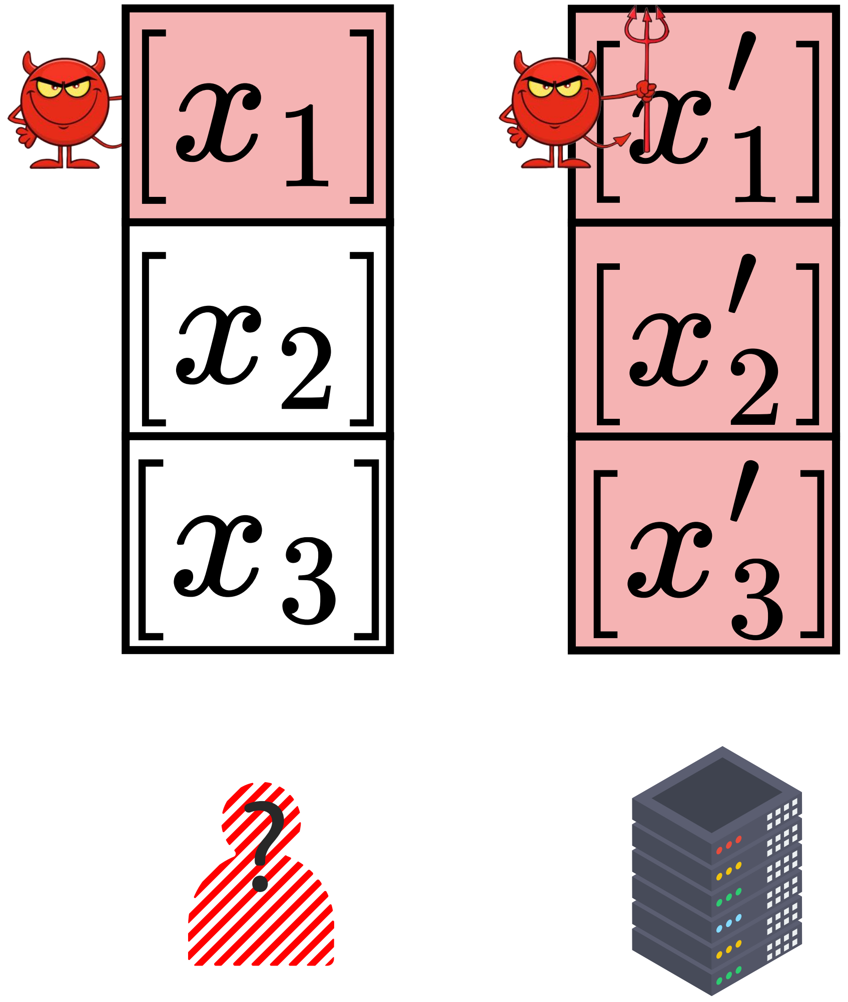
Private robustness [USENIX Security'21]:



Goal: <u>efficient G.O.D</u> for honest users against malicious adversaries

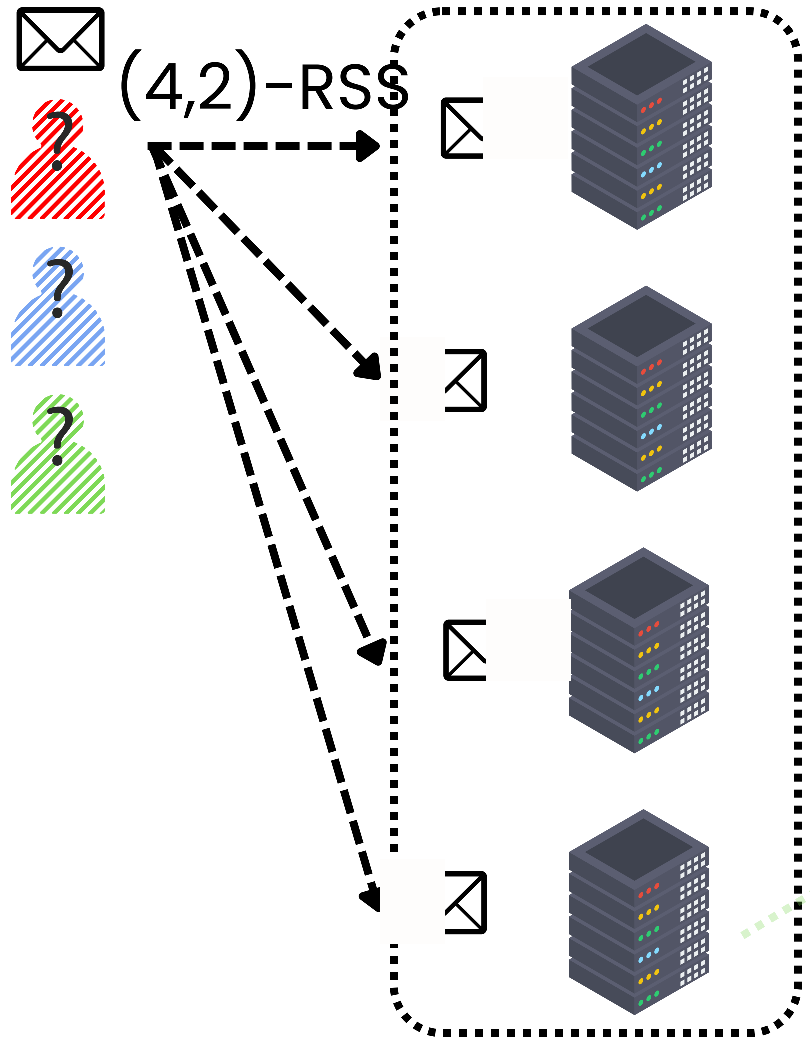
<u>Private G.O.D</u>: honest parties also cannot compromise anonymity!

Also see "MPC with Friends & Foes" [CRYPTO'20]



A Useful Tool: Replicated Sharing

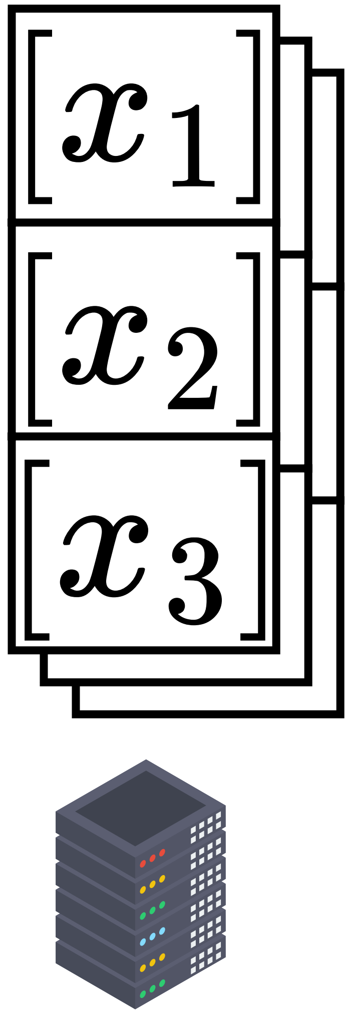
A minimal 4-server example:



Goal: <u>efficient G.O.D</u> for honest users against malicious adversaries

<u>Private G.O.D</u>: honest parties also cannot compromise anonymity!

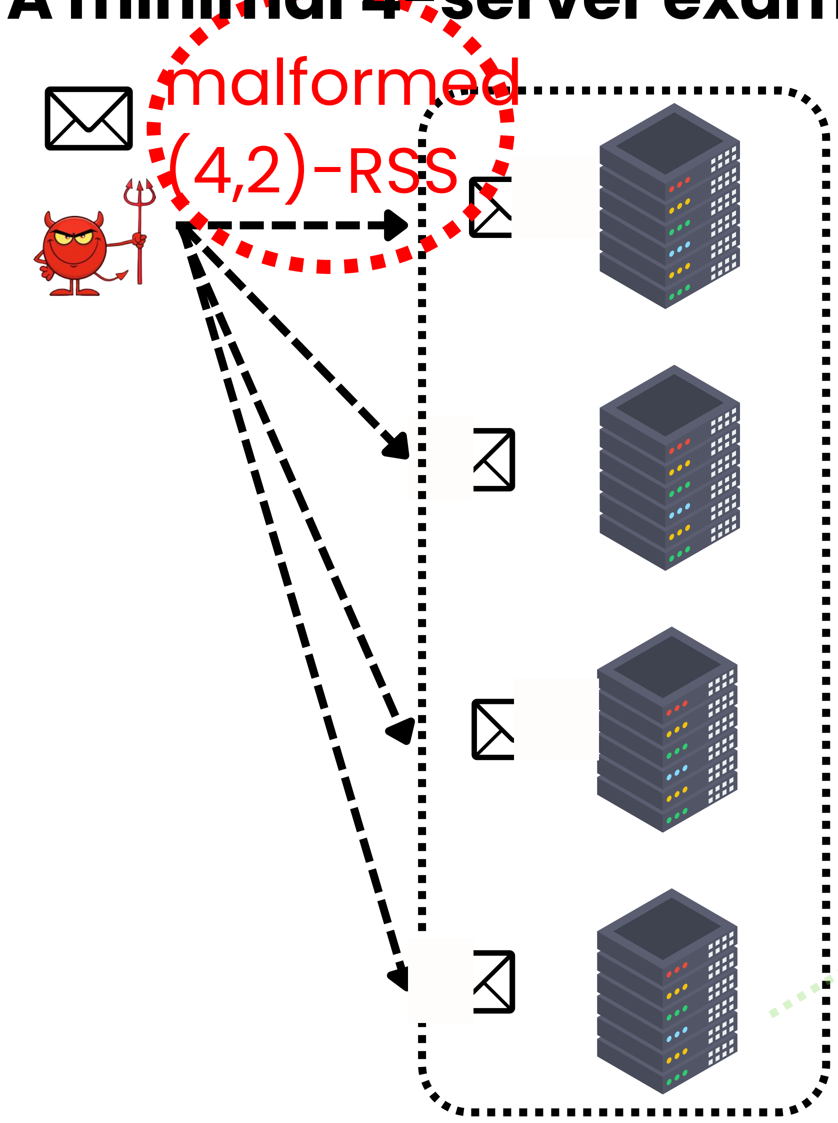
User submit:
(4,2)-replicated secret sharing (RSS)



Key observation: (4,2)-RSS is indeed an error correcction code.

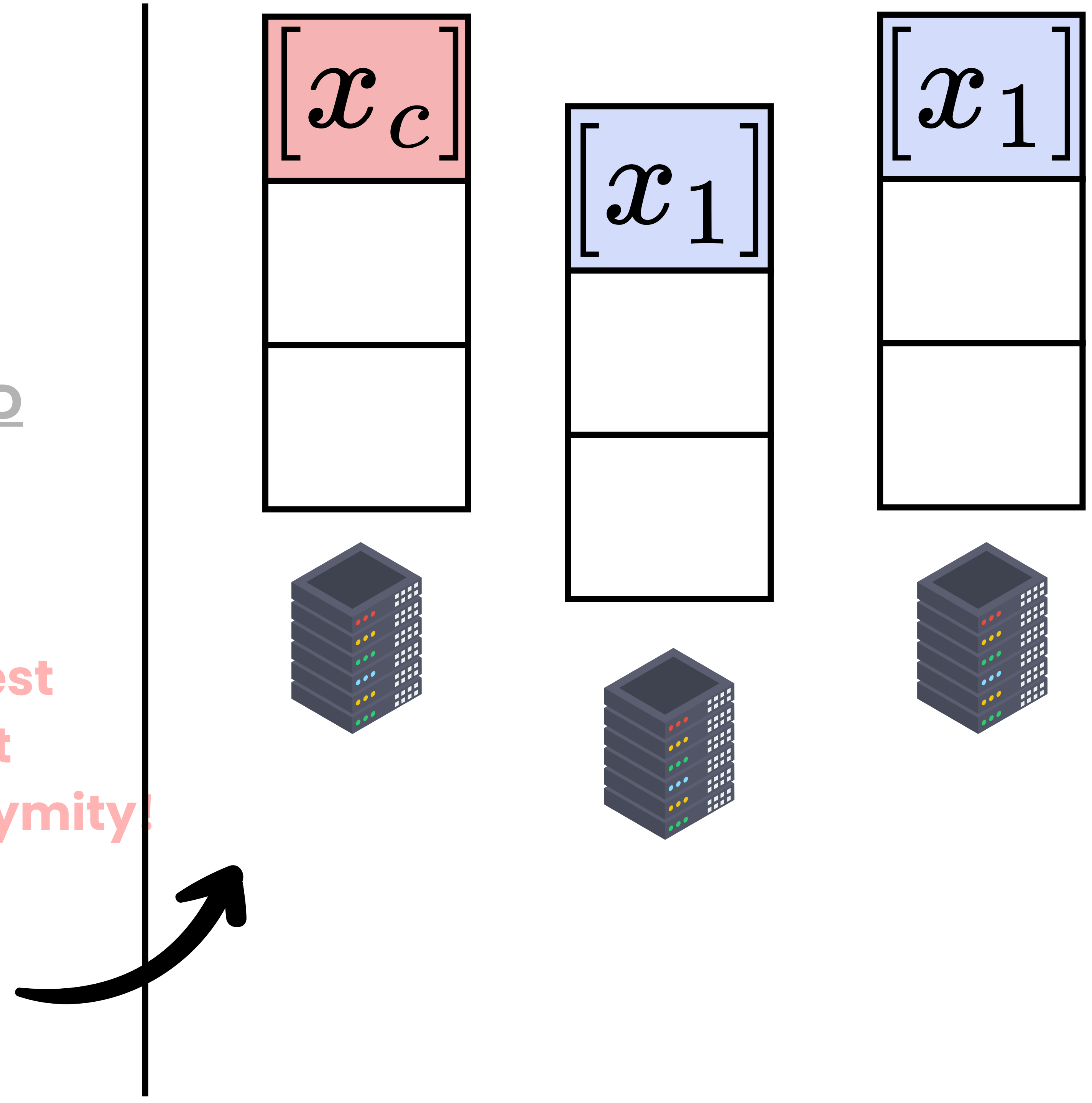
Malicious Users: Submit Malformed Data

A minimal 4-server example:



Goal: efficient G.O.D for honest users against malicious adversaries

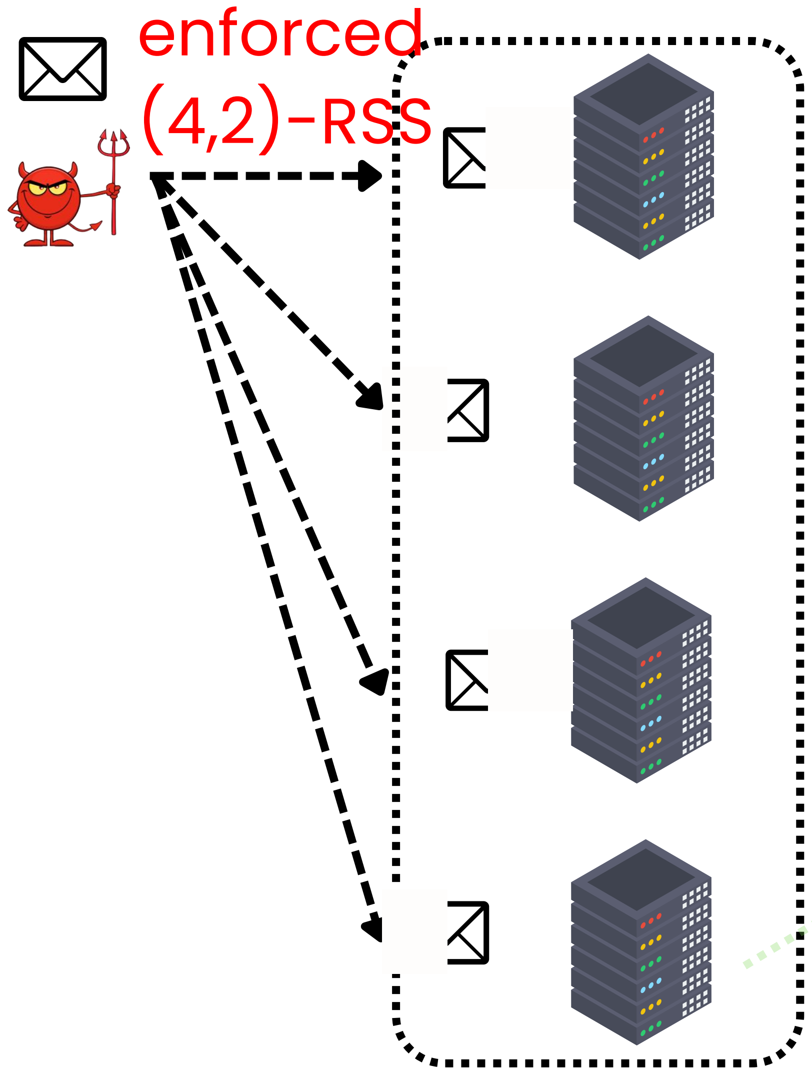
Private G.O.D: honest parties also cannot compromise anonymity!



Malicious users submit:
malformed message share.

Weak Verifiable Secret Sharing (WSS)

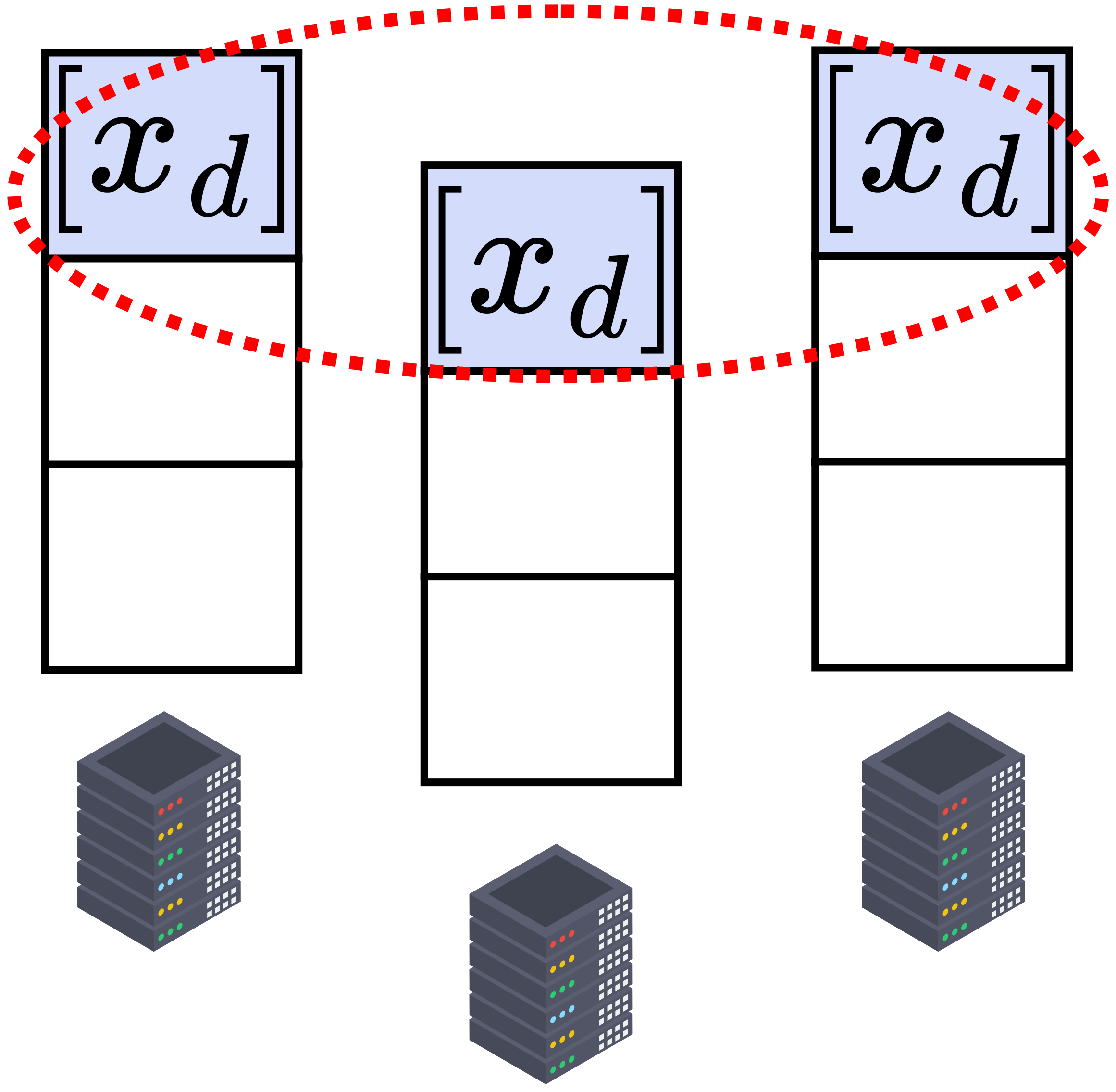
A minimal 4-server example:



Goal: efficient G.O.D for honest users against malicious adversaries

Private G.O.D: honest parties also cannot compromise anonymity!

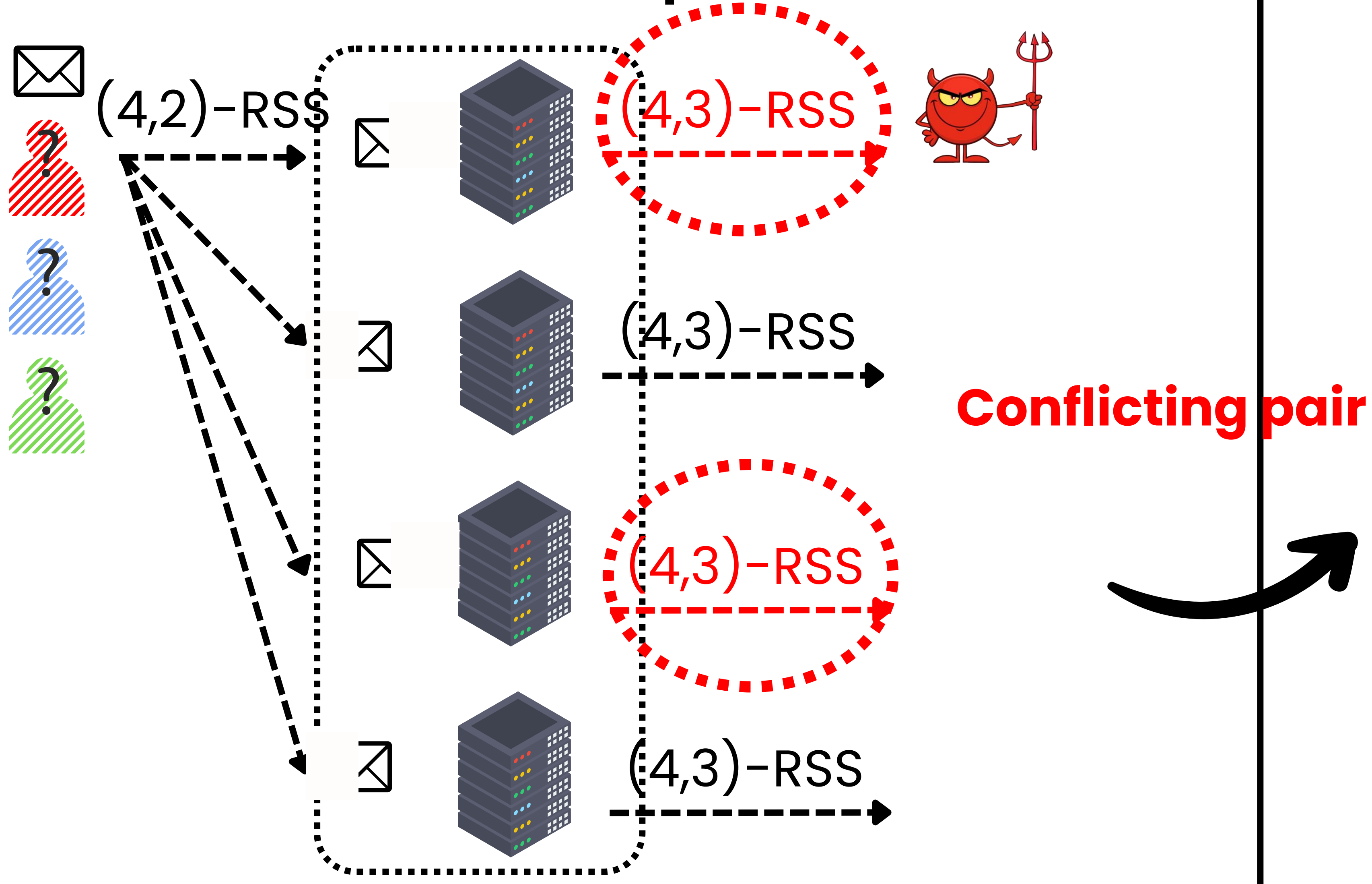
Enforce well-formed sharing via WSS.



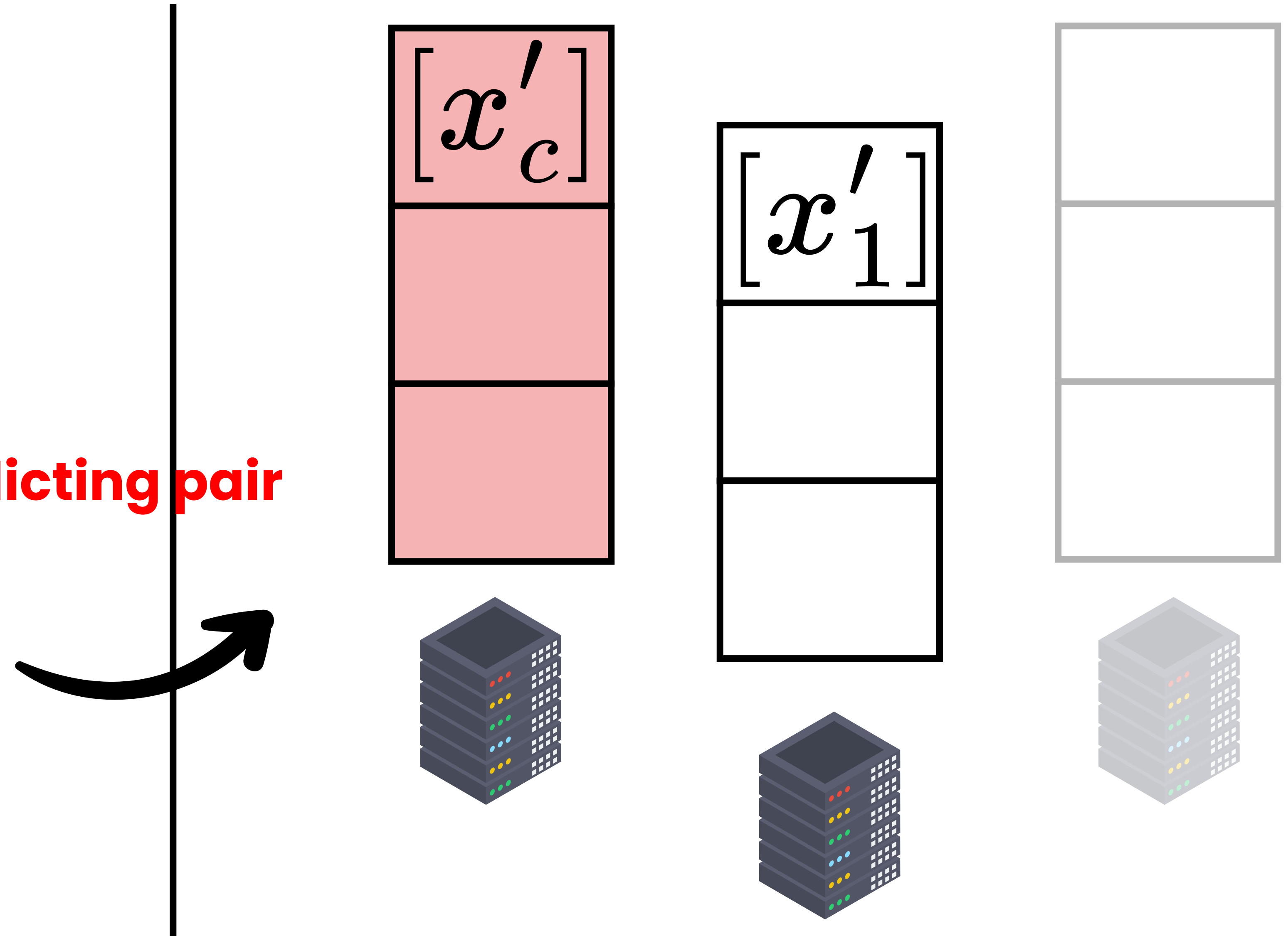
Key observation: No need to guarantee the integrity for malicious input, also see Rabin et al. [STOC'89]

Malicious Server: Send Malformed Data

A minimal 4-server example:



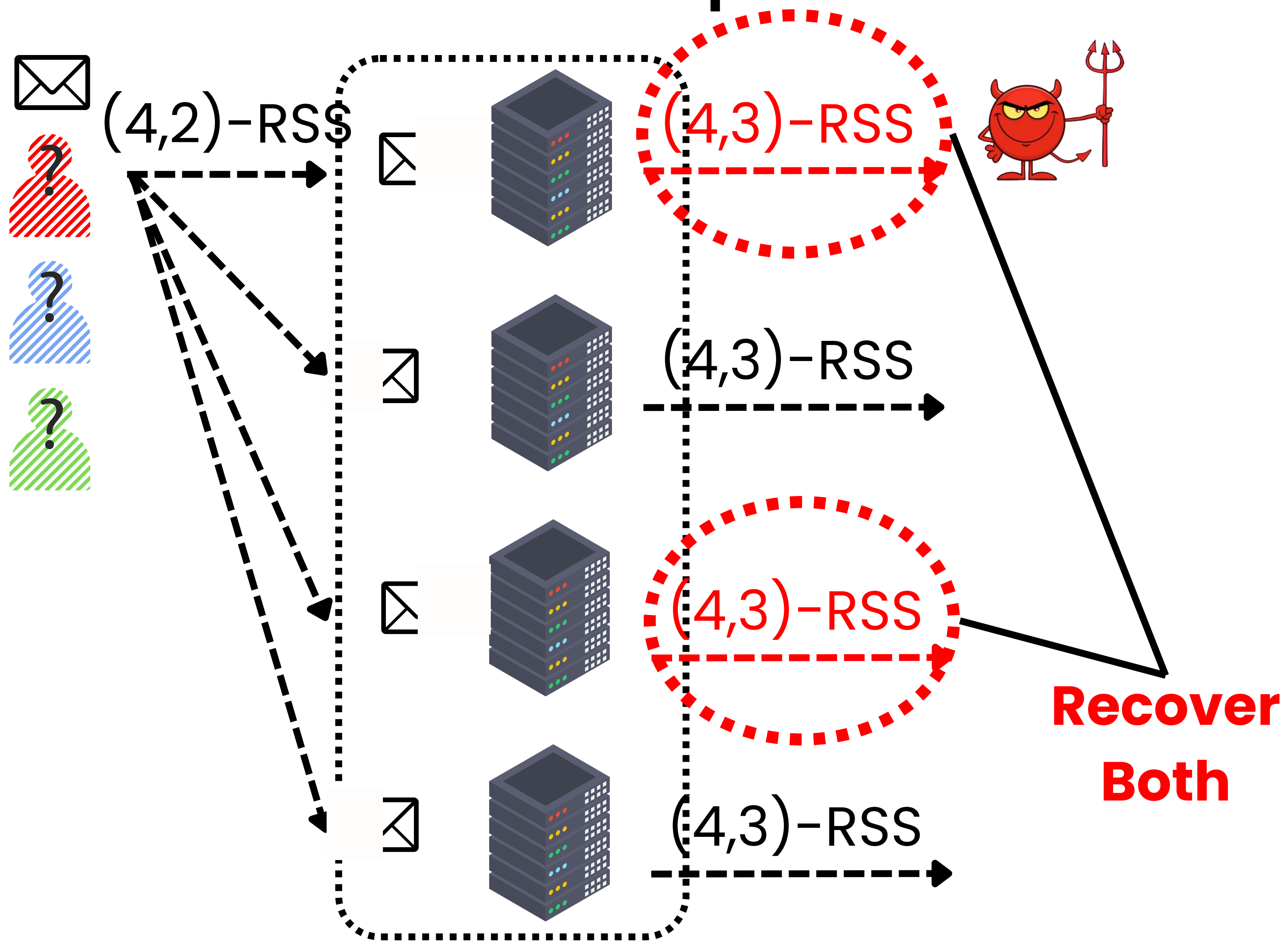
User submit:
(4,2)-replicated secret sharing (RSS)



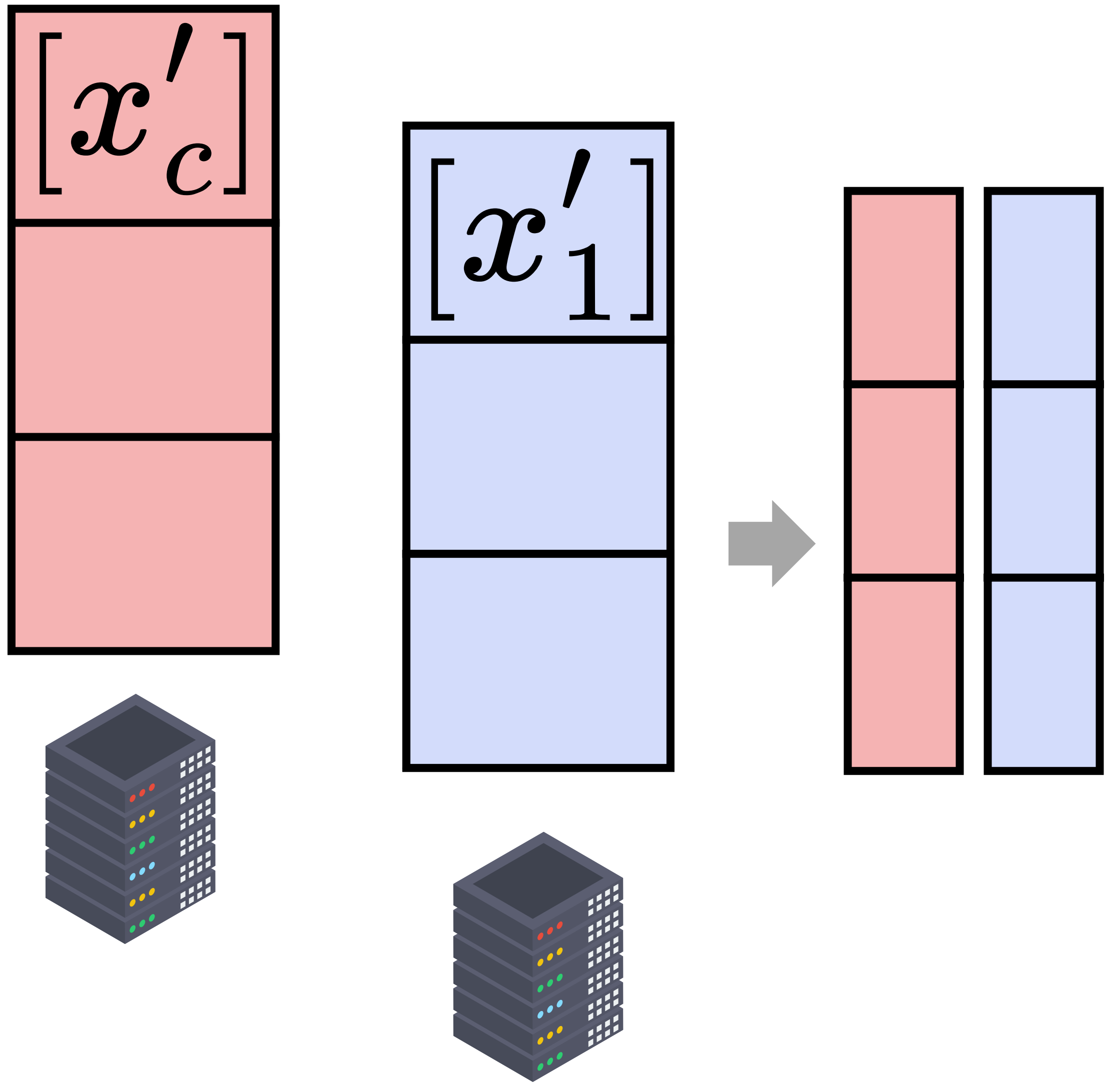
After silent shuffle:
(4,3)-replicated secret sharing (RSS)

Robustly Reconst Shuffled Messages

A minimal 4-server example:



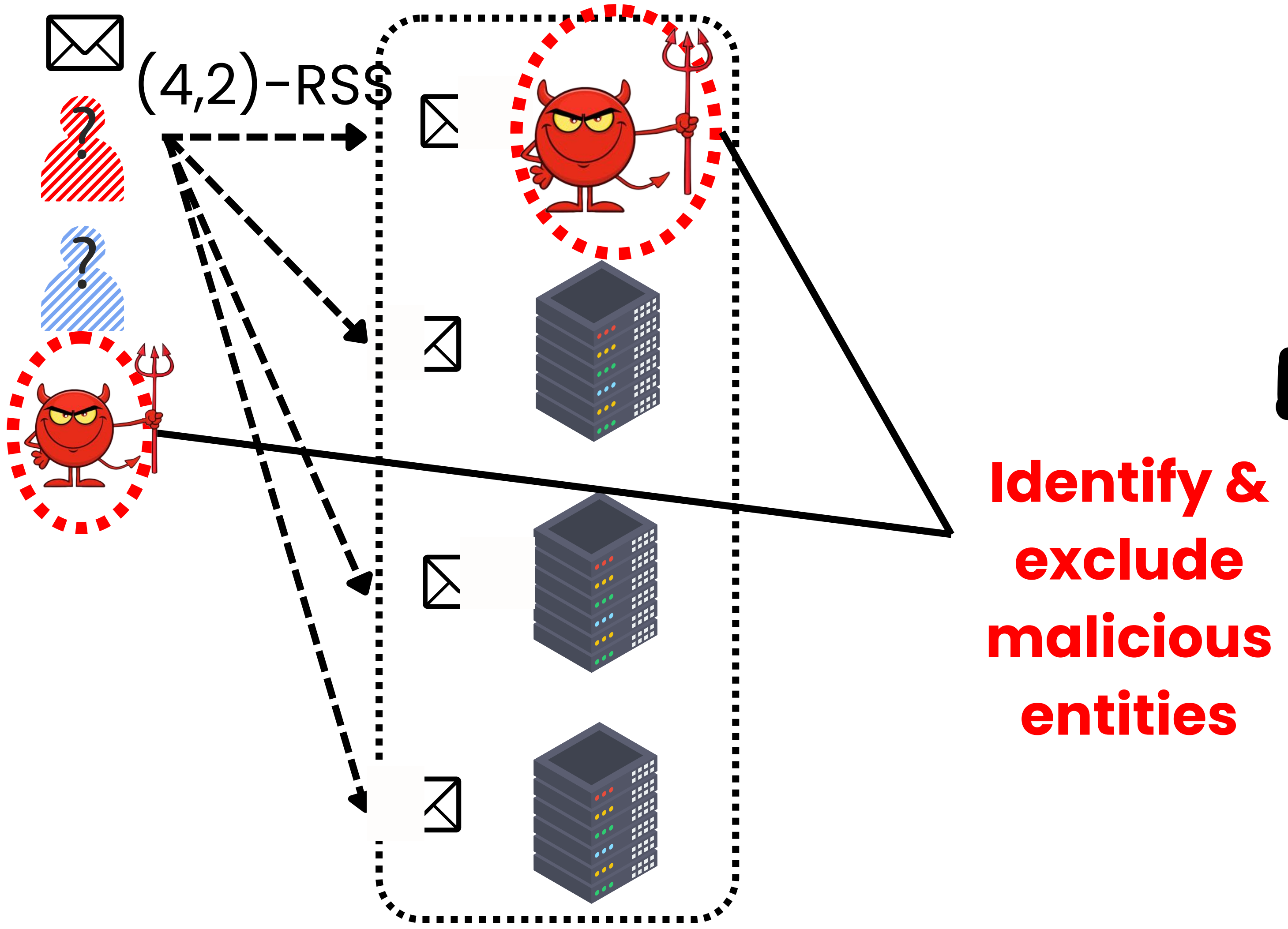
User submit:
(4,2)-replicated secret sharing (RSS)



The meaningful one should be correct message.

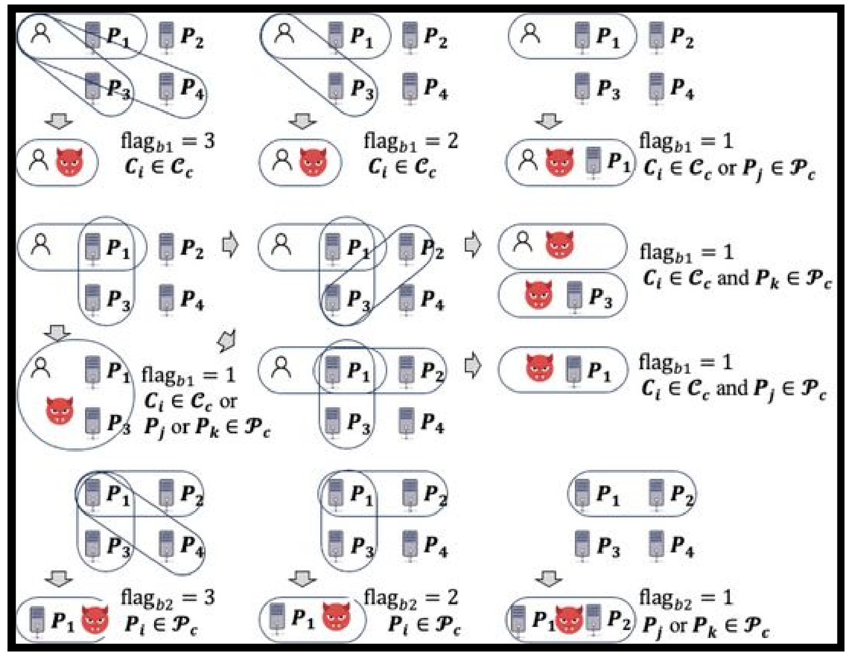
One More Thing: Blame Game & Reduction

A minimal 4-server example:



Run blame game to narrow down the malicious entities.

Reduction rules:

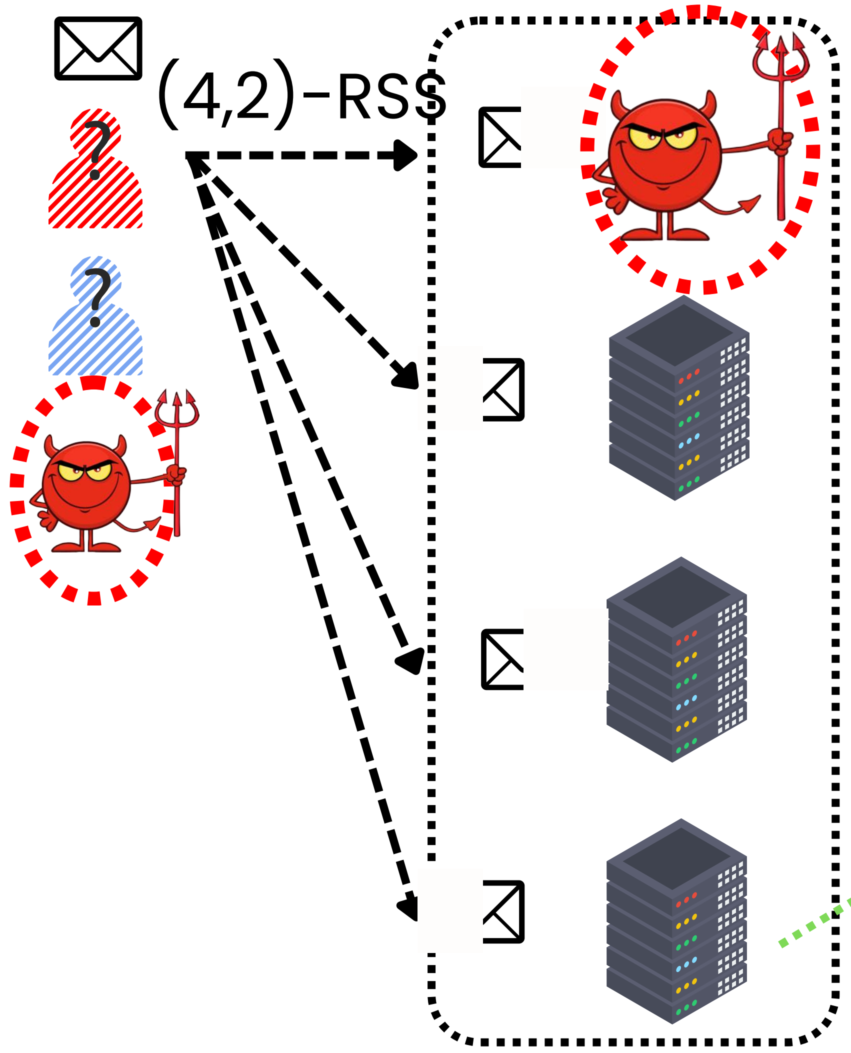


Key observation:

- once there is a conflict, at least one entity in the conflict group must be malicious.
- honest server never blame honest user.

Silent Shuffle with Private Robustness

A minimal 4-server example:



Goal: <u>efficient G.O.D</u> for honest users against malicious adversaries

<u>Private G.O.D</u>: honest parties also cannot compromise anonymity!

Not Done Yet: Anonymity is Not Utopia...

YikYak, a once-popular anonymous social media platform.

The Ring of Gyges

Plato's "Republic" (Book II, 359d–360b)

The Ring of Gyges unveils a truth:

unfettered anonymity, though cloaked in the guise of freedom, breeds not utopia but the unraveling of virtue. For when men act without the eyes of society upon them, even the just may become tyrants.

Not Done Yet: Anonymity is Not Utopia...

YikYak, a once-popular anonymous social media platform.

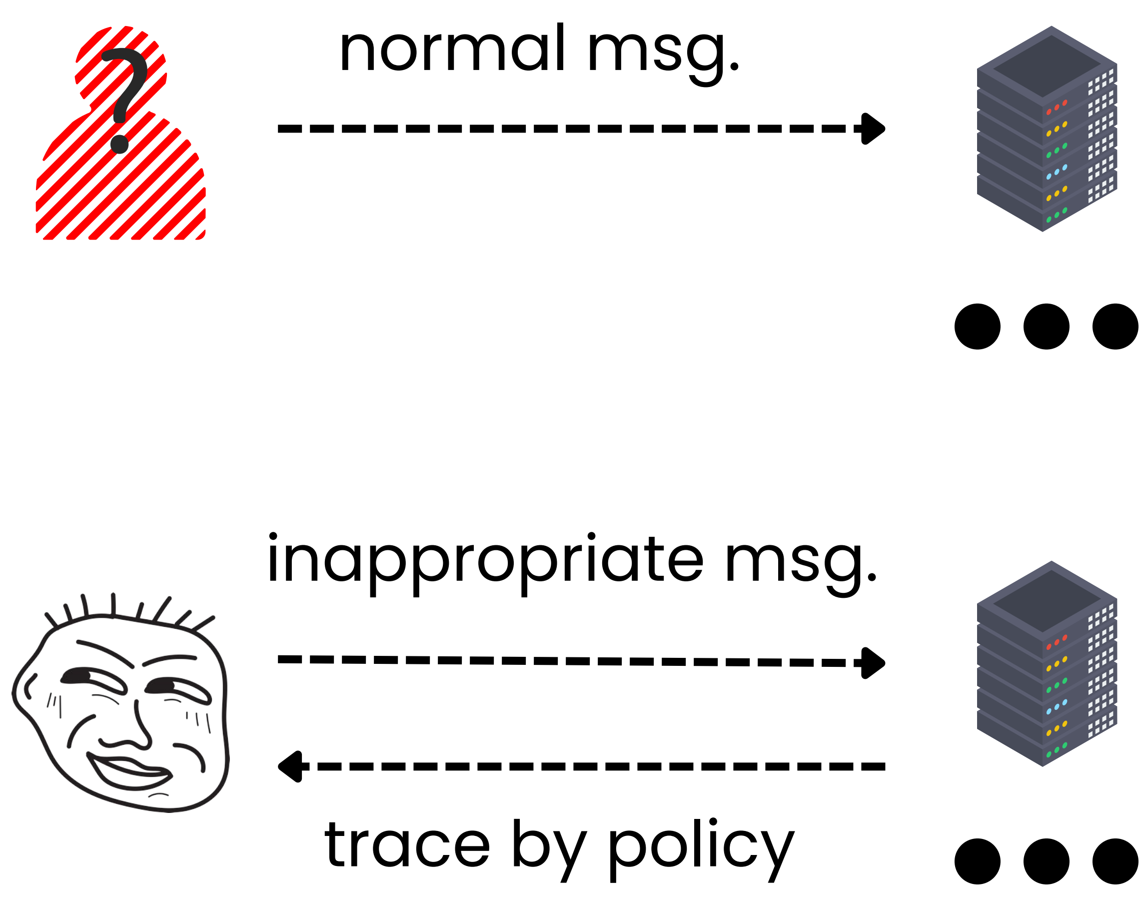
Anonymity may fail for lack of accountability against:

- Cyberbullying.
- Terrorist propaganda.
- Fake news
- ...

We May Expect: Accountable Anonymity

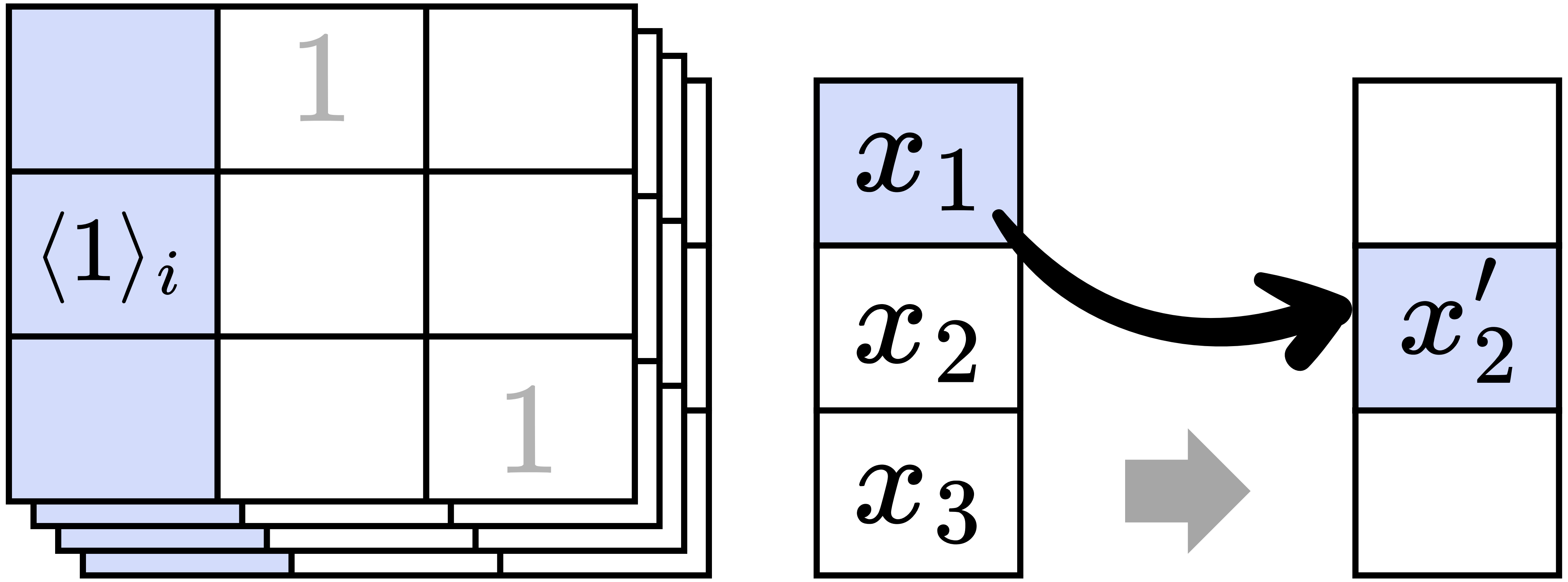
Goal:

- **Selectively trace** inappropriate messages back to misbehaving users.



How We Do This? Secret-Shared Trace

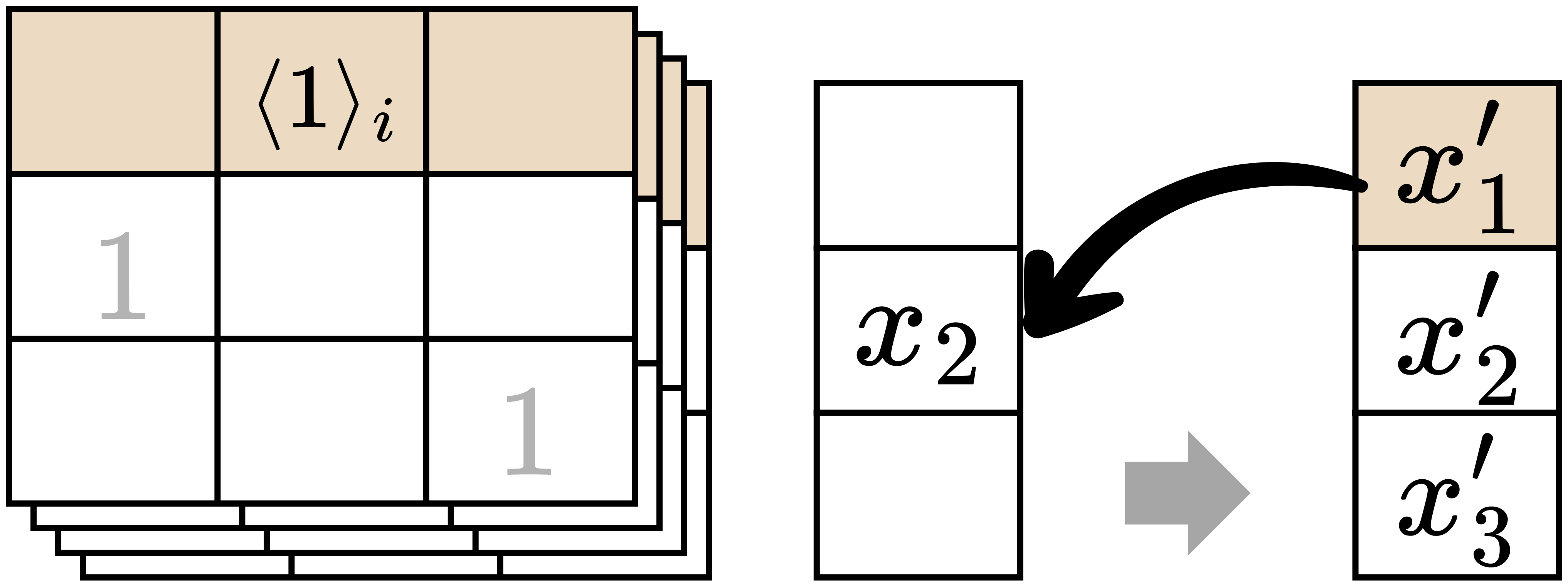
Recall: Boolean permutation correlation (BPC)



- For secure shuffle, we care the index of **1** in each column of BPC matrix.

How We Do This? Secret-Shared Trace

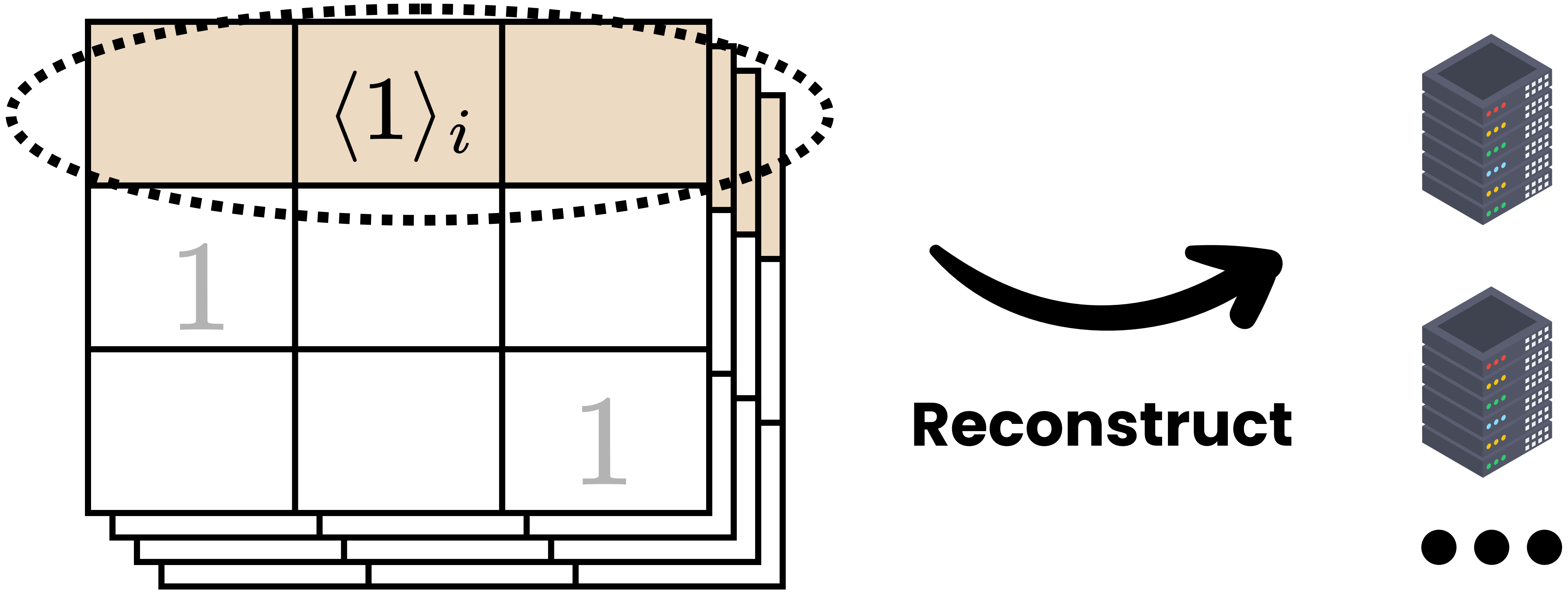
Recall: Boolean permutation correlation (BPC)



- For secure shuffle, we care the index of **1** in each ***column*** of BPC matrix.
- For selective trace, we care the index of **1** in each ***row*** of BPC matrix

How We Do This? Secret-Shared Trace

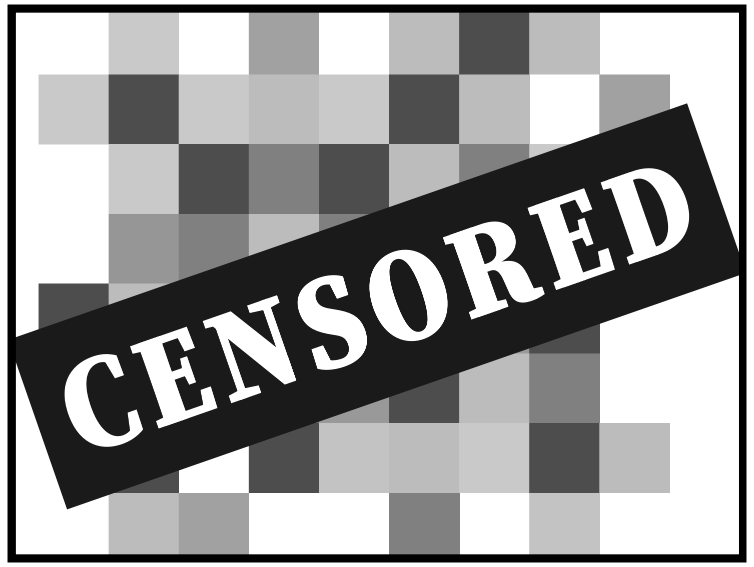
Recall: Boolean permutation correlation (BPC)



- For secure shuffle, we care the index of **1** in each ***column*** of BPC matrix.
- For selective trace, we care the index of **1** in each ***row*** of BPC matrix

Wait, Accountability Can Also Be Misused...

Misuse of accountability for censorship and others



- False accusations & reputation damage.
- Overly strict accountability that damage free of expression & innovation.
- Retaliation.
- ...

Moderation Policy to Mitigate It

Trigger trace by strict moderation policy:

- <u>Mechanism 1:</u> user reports over a threshold.
- <u>Mechanism 2:</u> all servers jointly audit the content.

If and only if both satisfy:

- servers jointly perform selective secret-shared trace protocol.
- manage misbehaving users.

Else:

- abort & report.

*The tracing (accountability) mechanism is exclusively activated for messages that violate predefined moderation policies, ensuring it is only applied to content deemed inappropriate by societal consensus.

Evalution

Offline microbenckmark:

Ref.	N msg no.	ℓ msg size	**Clarion [NDSS'22]**	**RPM-I [PETs'23]**	**Gyges**
Comm. [MB]	10^4	8 B	2.0	1600	1.4
		32 B	2.6	6400	
		160 B	5.8	32000	

Key observation:

- Boolean representated correlation compress offline communication, independent of ℓ .
- perform well when message size is large.

Evalution

Online microbenckmark:

Ref.	N msg no.	Clarion [NDSS'22]	RPM-I [PETs'23]	RPM-II [PETs'23]	RPM-II [PETs'23]	Gyges
Comm. [MB]	10^4	2.3	0.7	1.0	2.8	0
Time [s]		0.7	1.5	0.6	0.6	0.2

Key observation:

- Silent shuffle achieve optimal communication.
- perform well when network latency is high.
- perform well when bandwidth is limited.

Evalution

Scaling microbenckmark:

Ref.	Setup	No. of servers per party		
		1	**2**	**3**
Throughput [10^7 entry / min]	CPU (w/o sparsity)	1.3	2.5	3.8
	GPU (w/ sparsity)	1.5	3.1	4.6

Explanation:
- Tested under the LAN setting for mixing 10^5 messages, each of 8 B.

Key observation:
- silent protocol can be easily scaled.

Evalution

Tracing microbenckmark:

Ref.			Traceable mixnets [PETs'24]	Gyges
Shuffle	n=4 N=10^4	Time [s]	343 / 4	0.2
Trace		Time [s]	682 / 4	0.1

Key observation:

- Compared to traceable mix-nets solutions, MPC tracing solutions are far more lightweight in execution time.

Thanks & Questions?

See paper for things not covered!
For example:

- Protocol details.
- Blaming mechanism details.
- Security analysis.
- Moderation policy.
- ...

Be invisible, also be moral