NDSS SYMPOSIUM/2024

Secret-shared Shuffle with Malicious Security

Xiangfu Song[1], Dong Yin[2], Jianli Bai[3], Changyu Dong[4], Ee-Chien Chang[1]

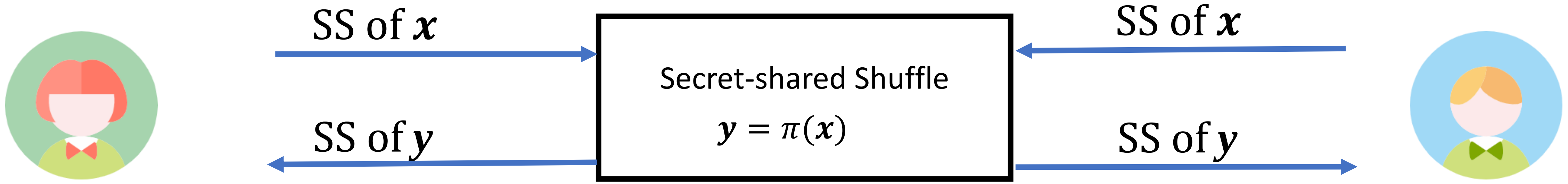
[1]National University of Singapore [2]Ant Group [3]University of Auckland [4]Guangzhou University

Table of Contents

Background: Shuffle

- When playing cards and mah-jong...

Background: Secret-shared Shuffle (SSS)



- Privacy goals:
  - No party learns any information about $\boldsymbol{x}$ or $\boldsymbol{y}$.
  - No party learns any information about π.

Background: Applications of SSS

- Privacy-preserving data analysis over secret-shared data

T

Name	diabetes	weight
Alice	1	120
Bob	0	70
Cindy	0	60
David	1	90
Edward	0	80

Background: Applications of SSS

- Privacy-preserving data analysis over secret-shared data

T

Name	diabetes	weight
Alice	1	120
Bob	0	70
Cindy	0	60
David	1	90
Edward	0	80

Background: Applications of SSS

- Privacy-preserving data analysis over secret-shared data

T

Name	diabetes	weight
Alice	1	120
Bob	0	70
Cindy	0	60
David	1	90
Edward	0	80

Background: Applications of SSS

- Privacy-preserving data analysis over secret-shared data

T

Name	diabetes	weight
Alice	1	120
Bob	0	70
Cindy	0	60
David	1	90
Edward	0	80

Background: Applications of SSS

- Privacy-preserving data analysis over secret-shared data

T

Name	diabetes	weight
Alice	1	120
Bob	0	70
Cindy	0	60
David	1	90
Edward	0	80

→ Secret-shared Shuffle →

T’

diabetes	weight
0	70
0	80
1	90
0	60
1	120

Background: Applications of SSS

- Privacy-preserving data analysis over secret-shared data

T

Name	diabetes	weight
Alice	1	120
Bob	0	70
Cindy	0	60
David	1	90
Edward	0	80

Secret-shared Shuffle

T'

diabetes	weight
0	70
0	80
1	90
0	60
1	120

Background: Applications of SSS

- Privacy-preserving data analysis over secret-shared data

T

Name	diabetes	weight
Alice	1	120
Bob	0	70
Cindy	0	60
David	1	90
Edward	0	80

→ Secret-shared Shuffle →

T'

diabetes	weight
0	70
0	80
1	90
0	60
1	120

Background: More Applications

- Private database Join and aggregation [MRR20, ACDG+21, JSZD+22]
- Secure graph analysis [AFOP+21]
- Secure sorting [AHIK+22]
- Anonymous communication [EB22, LK23]
- …

[MRR20] P. Mohassel, P. Rindal, and M. Rosulek, “Fast database joins and psi for secret shared data”, ACM CCS 2020.

[ACDG+21] E. Anderson, M. Chase, F. B. Durak, E. Ghosh, K. Laine, and C. Weng, “Aggregate measurement via oblivious shuffling,” Cryptology ePrint Archive, 2021.

[JSZD+22] Y. Jia, S. Sun, H. Zhou, J. Du, and D. Gu, “Shuffle-based private set union: Faster and more secure”, USENIX Security 2022.

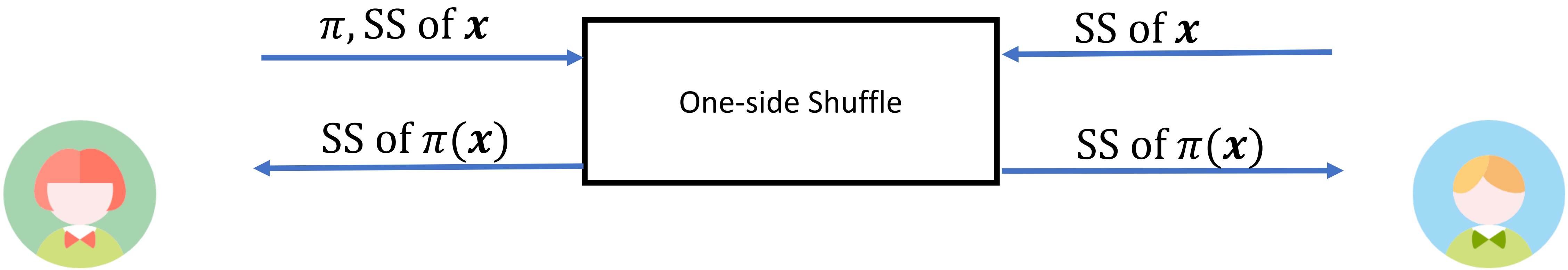
[AFOP+21] T. Araki, J. Furukawa, K. Ohara, B. Pinkas, H. Rosemarin, and H. Tsuchida, “Secure graph analysis at scale”, ACM CCS 2021.

[AHIK+22] G. Asharov, K. Hamada, D. Ikarashi, R. Kikuchi, A. Nof, B. Pinkas, K. Takahashi, and J. Tomida, “Efficient secure three-party sorting with applications to data analysis and heavy hitters”, ACM CCS 2022.

[EB22] S. Eskandarian and D. Boneh, “Clarion: Anonymous communication from multiparty shuffling protocols”, NDSS 2022.

[LK23] D. Lu and A. Kate, “Rpm: Robust anonymity at scale”, PoPETs 2023.

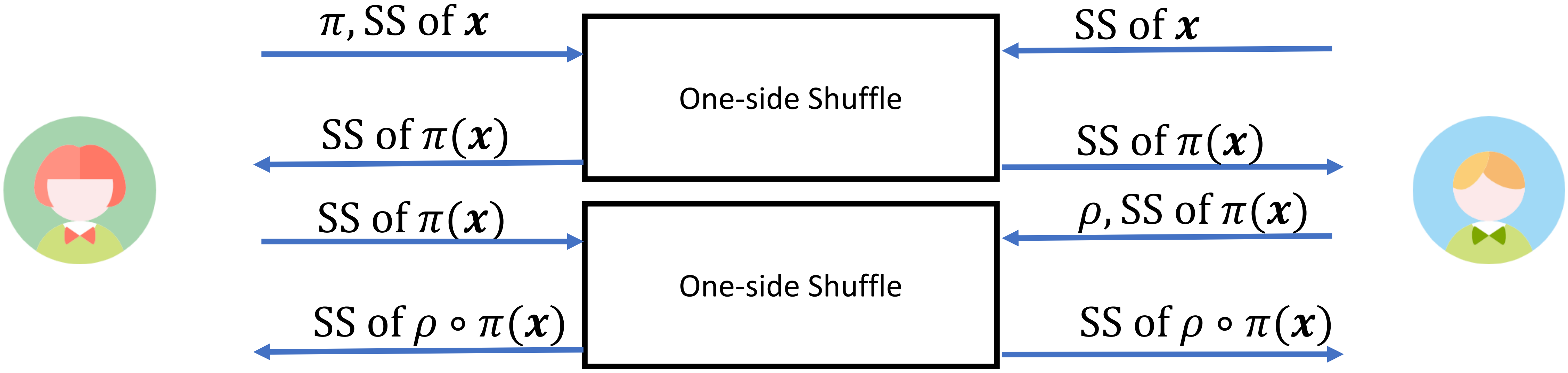
CGP Shuffle Protocol: Overview



- Proposed by Chase-Ghosh-Poburinnaya [CGP20]

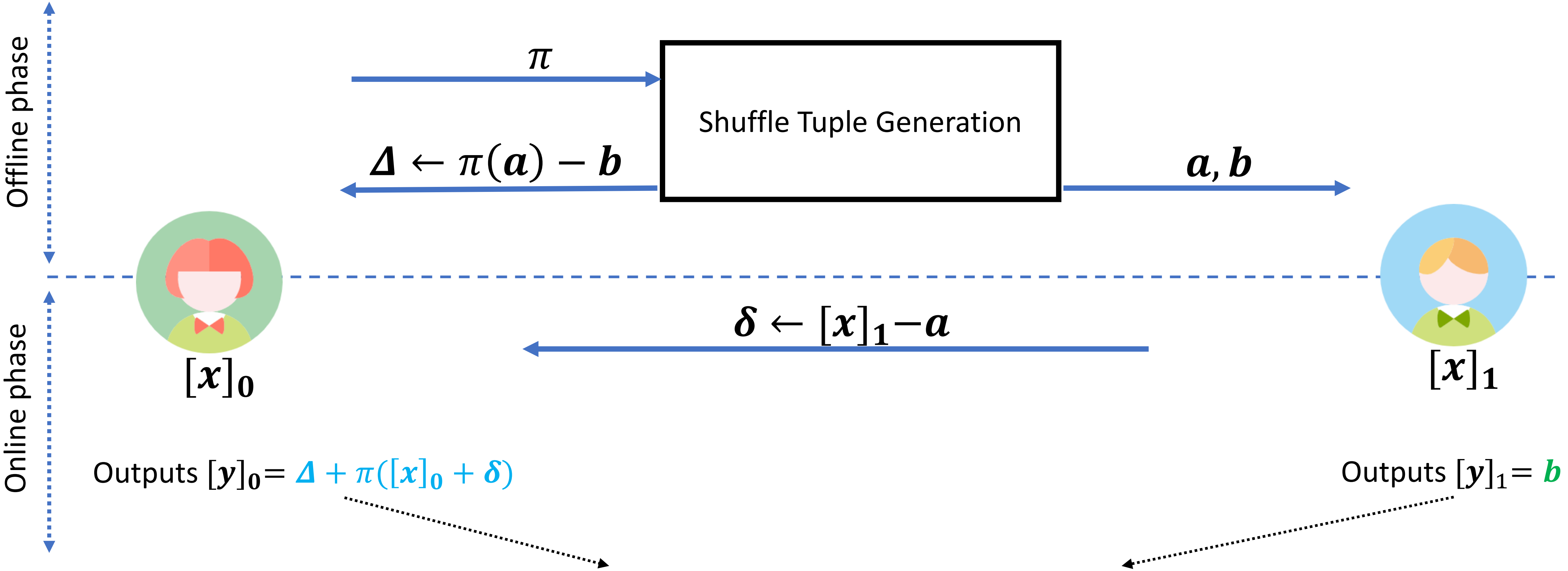
[CGP20] M. Chase, E. Ghosh, and O. Poburinnaya, “Secret-shared shuffle,” ASIACRYPT 2020.

CGP Shuffle Protocol: Overview



- Two one-side shuffle → (two-side) secret-shared shuffle

One-side Shuffle Protocol



$$[\boldsymbol{y}]_0 + [\boldsymbol{y}]_1 = \boldsymbol{\Delta} + \pi(\boldsymbol{\delta} + [\boldsymbol{x}]_0) + \boldsymbol{b} = \pi(\boldsymbol{a}) - \boldsymbol{b} + \pi([\boldsymbol{x}]_1 - \boldsymbol{a} + [\boldsymbol{x}]_0) + \boldsymbol{b} = \pi(\boldsymbol{x})$$

Semi-honest One-side Shuffle: Offline Phase

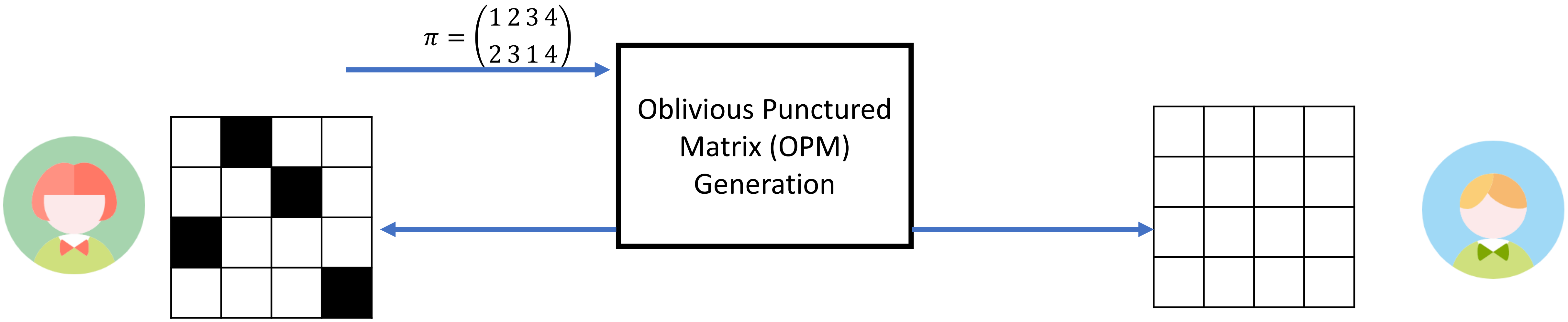
Oblivious Punctured Matrix (OPM) Generation

Semi-honest One-side Shuffle: Offline Phase

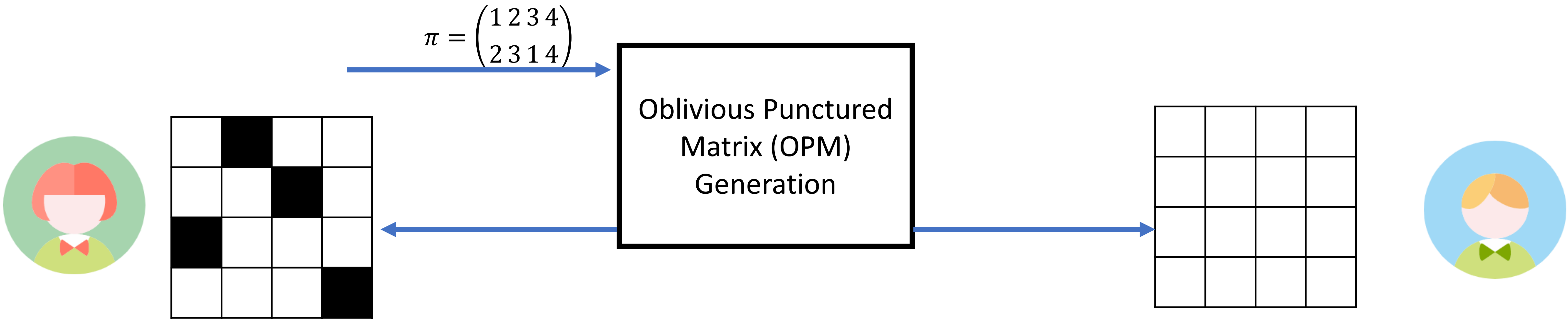
$$\pi = \begin{pmatrix} 1\,2\,3\,4 \\ 2\,3\,1\,4 \end{pmatrix}$$

Oblivious Punctured Matrix (OPM) Generation

Semi-honest One-side Shuffle: Offline Phase

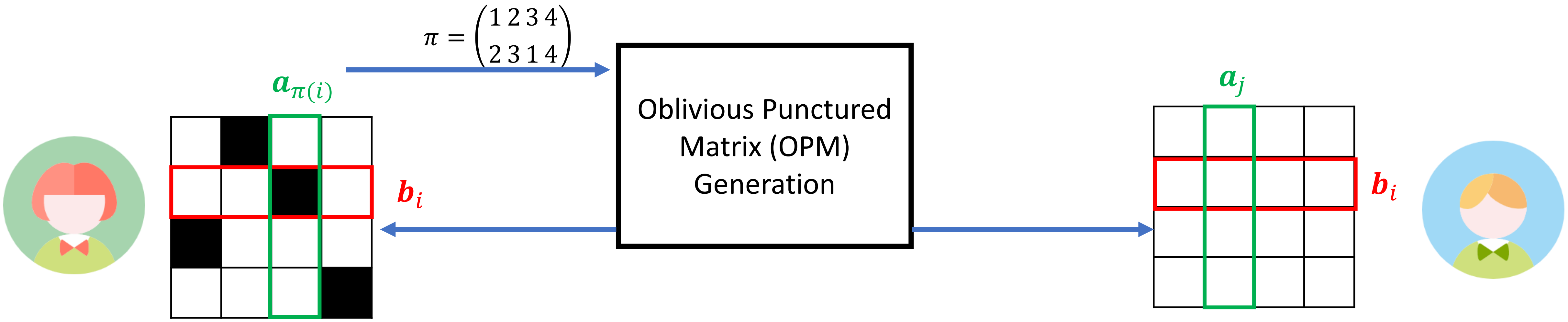


Semi-honest One-side Shuffle: Offline Phase



- OPM generation: Oblivious transfer (OT) + Puncturable pseudorandom function (PPRF)

Semi-honest One-side Shuffle: Offline Phase



Alice doesn't know $\mathbf{M}_{i,\pi(i)}$ for row i

Outputs $\boldsymbol{\Delta}$ such that $\boldsymbol{\Delta}_i = \text{sum of column } \pi(i) - \text{sum of row } i$
$= \boldsymbol{a}_{\pi(i)} - \boldsymbol{b}_i$

Outputs $\boldsymbol{a}, \boldsymbol{b}$ such that $\boldsymbol{b}_i = \text{sum of row } i$ and $\boldsymbol{a}_j = \text{sum of column } j$,

$\boldsymbol{\Delta} = \pi(\boldsymbol{a}) - \boldsymbol{b}$ as required

Maliciously Secure SSS

Semi-honest (Honest-but-Curious) adversary

vs.

Malicious adversary

Maliciously Secure SSS

Semi-honest (Honest-but-Curious) adversary

vs.

Malicious adversary

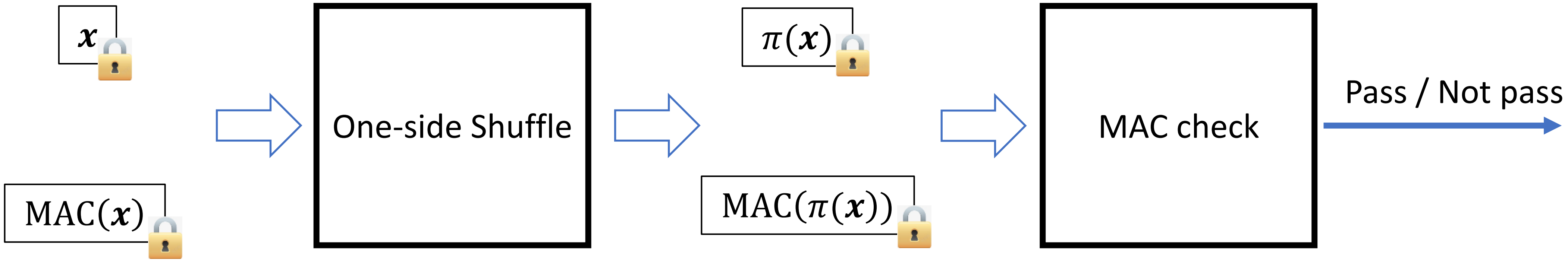
- Two existing maliciously secure CGP shuffle protocols [Lau21, EB22]
- Security goals
  - *Privacy*: hiding 1) the shared secrets and 2) the permutation being used.
  - *Correctness*: ensuring 1) integrity of the shared secrets and 2) a correct shuffling.

[Lau21] P. Laud, "Linear-time oblivious permutations for spdz", CANS 2021.

[EB22] S. Eskandarian and D. Boneh, "Clarion: Anonymous communication from multiparty shuffling protocols", NDSS 2022.

Existing Maliciously Secure SSS: Online Phase

- Ensuring correctness using MACs [Lau21, EB21]

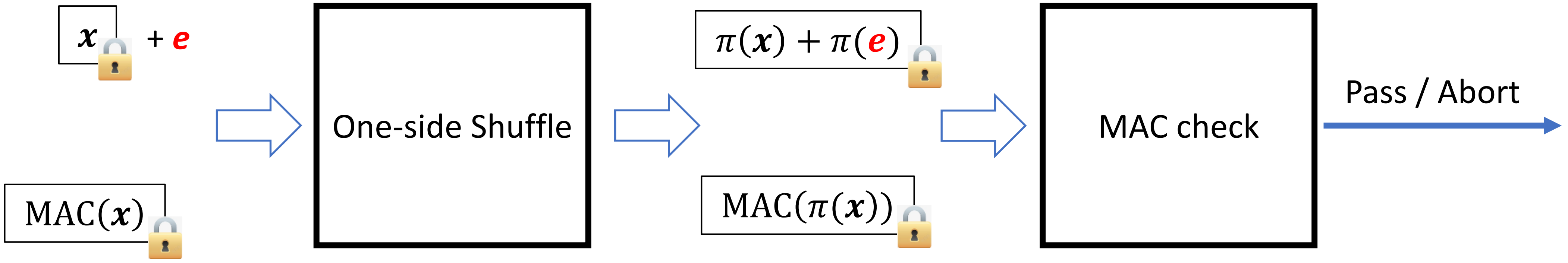


[EB21] S. Eskandarian and D. Boneh, "Clarion: Anonymous communication from multiparty shuffling protocols", NDSS 2022.

[Lau21] P. Laud, "Linear-time oblivious permutations for spdz", CANS 2021.

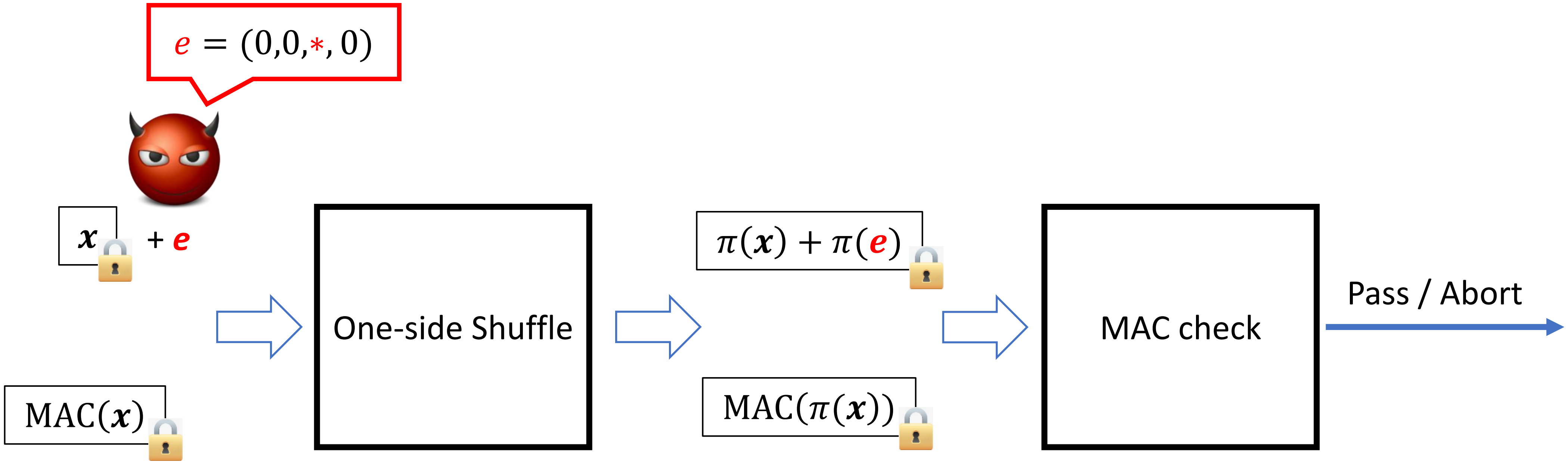
Existing Maliciously Secure SSS: Online Phase

- Ensuring correctness using MACs



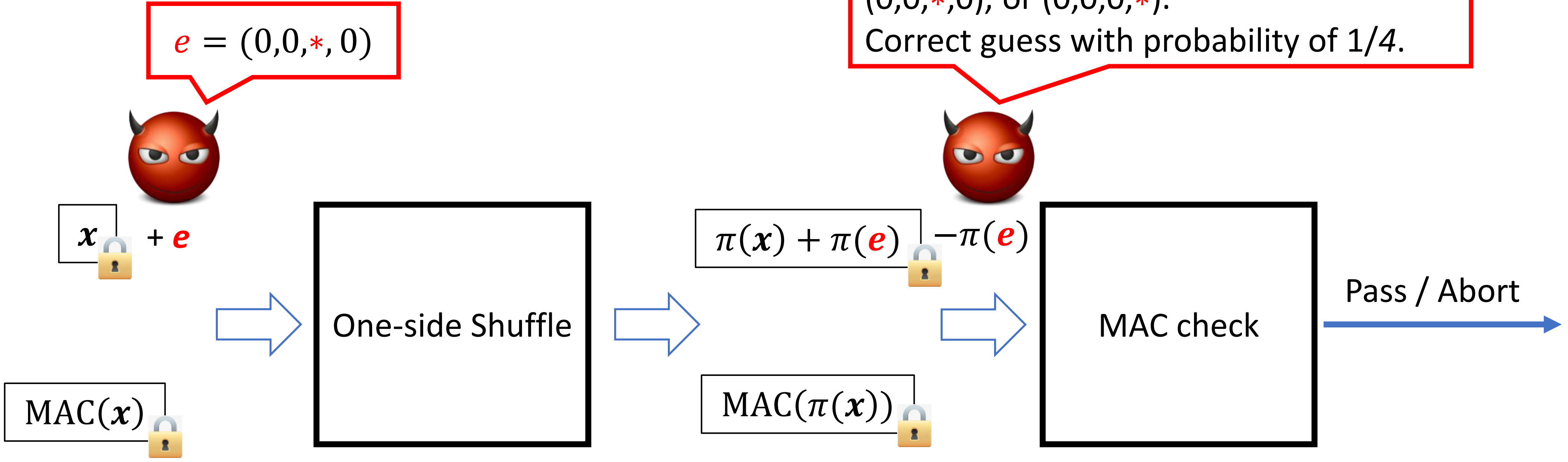
Existing Maliciously Secure SSS: Online Phase

- Ensuring correctness using MACs

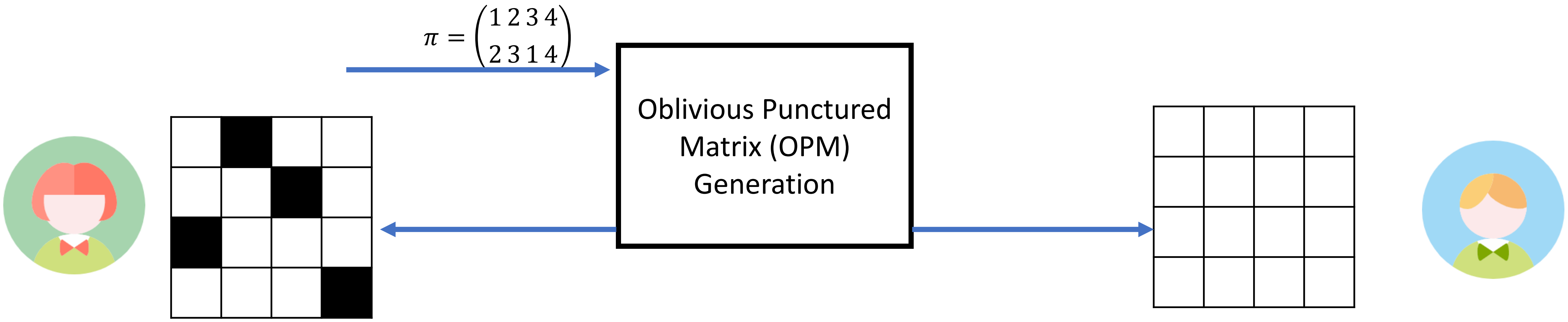


Existing Maliciously Secure SSS: Online Phase

- Ensuring correctness using MACs



Existing Maliciously Secure SSS: Offline Phase

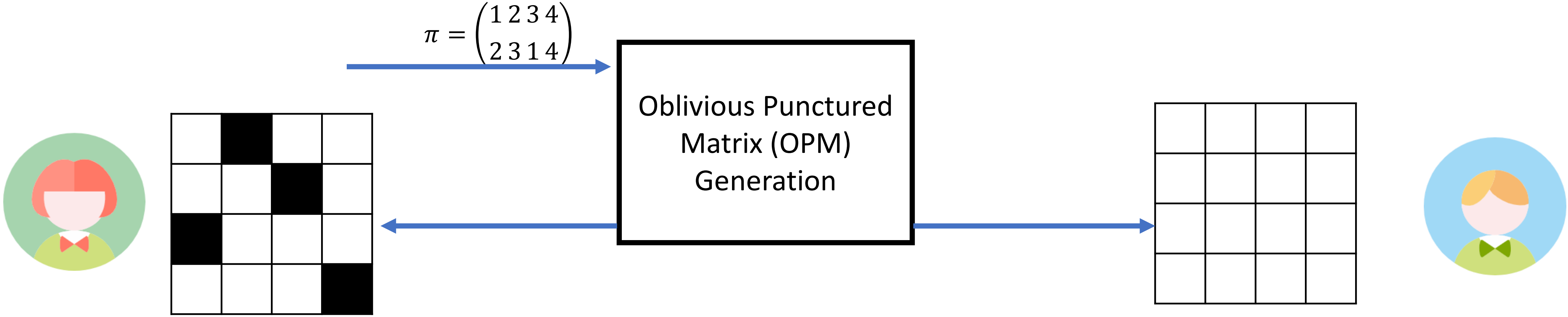


- Previous works [Lau21, EB22]
  - Use maliciously secure OT in the correlation generation protocol
  - Other parts remains unchanged as the semi-honest version

[Lau21] P. Laud, “Linear-time oblivious permutations for spdz”, CANS 2021.

[EB22] S. Eskandarian and D. Boneh, “Clarion: Anonymous communication from multiparty shuffling protocols”, NDSS 2022.

Existing Maliciously Secure SSS: Offline Phase



- Previous works [Lau21, EB22]
  - Use maliciously secure OT in the correlation generation protocol
  - Other parts remains unchanged as the semi-honest version
  - Selective failure attacks from the sender
  - An attack exploiting incorrect OPM from the receiver

[Lau21] P. Laud, "Linear-time oblivious permutations for spdz", CANS 2021.

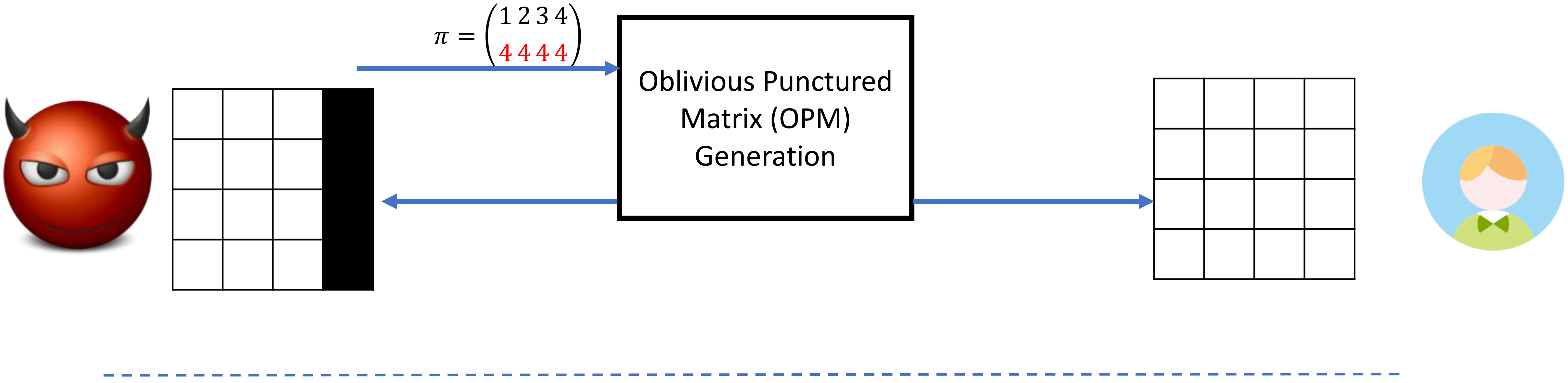
[EB22] S. Eskandarian and D. Boneh, "Clarion: Anonymous communication from multiparty shuffling protocols", NDSS 2022.

An OPM Attack From the Receiver

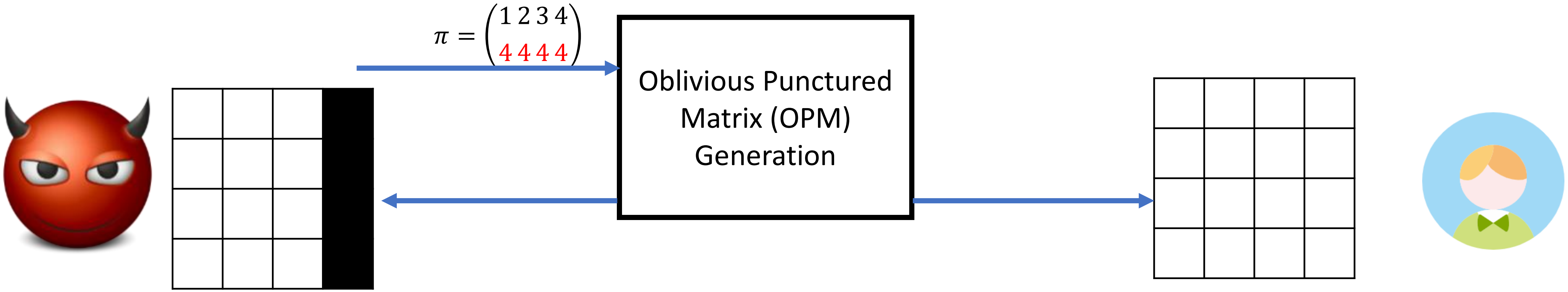
$$\pi = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 4 & 4 & 4 \end{pmatrix}$$

Oblivious Punctured Matrix (OPM) Generation

An OPM Attack From the Receiver

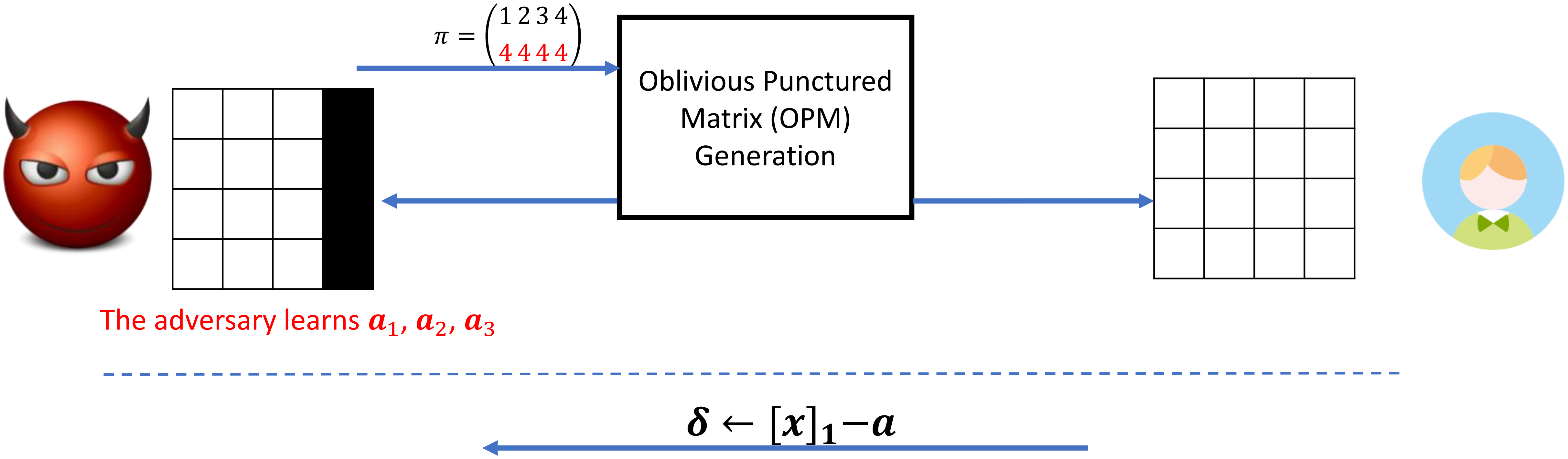


An OPM Attack From the Receiver

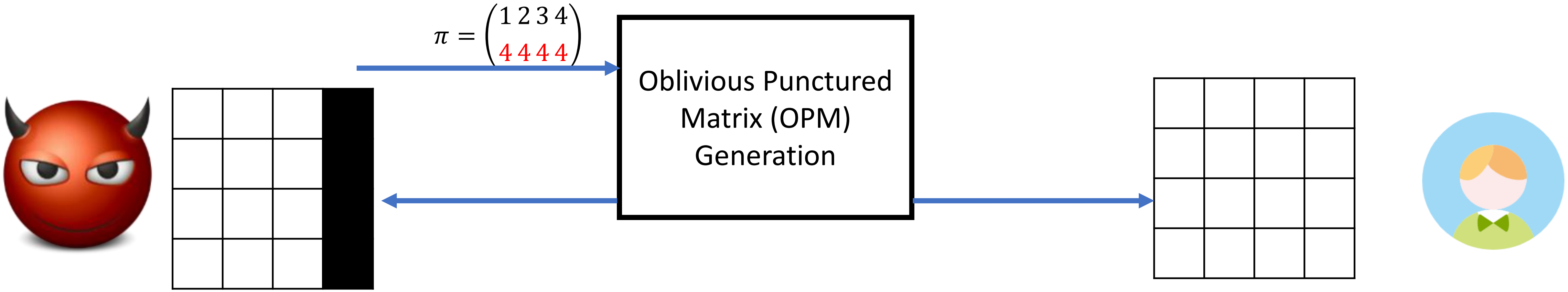


The adversary learns $\boldsymbol{a}_1$, $\boldsymbol{a}_2$, $\boldsymbol{a}_3$

An OPM Attack From the Receiver



An OPM Attack From the Receiver

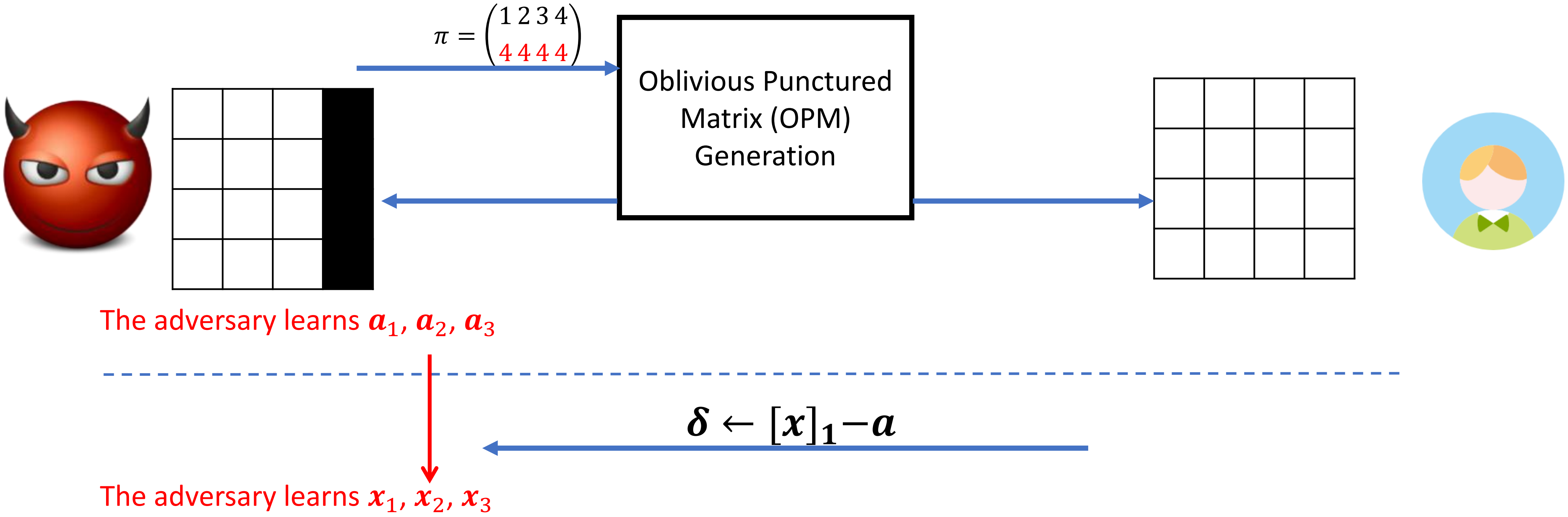


The adversary learns $\boldsymbol{a}_1, \boldsymbol{a}_2, \boldsymbol{a}_3$

$$\boldsymbol{\delta} \leftarrow [\boldsymbol{x}]_1 - \boldsymbol{a}$$

The adversary learns $\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3$

An OPM Attack From the Receiver



- Incorrect correlations → privacy breach
- Do perform well-formness check before using correlations

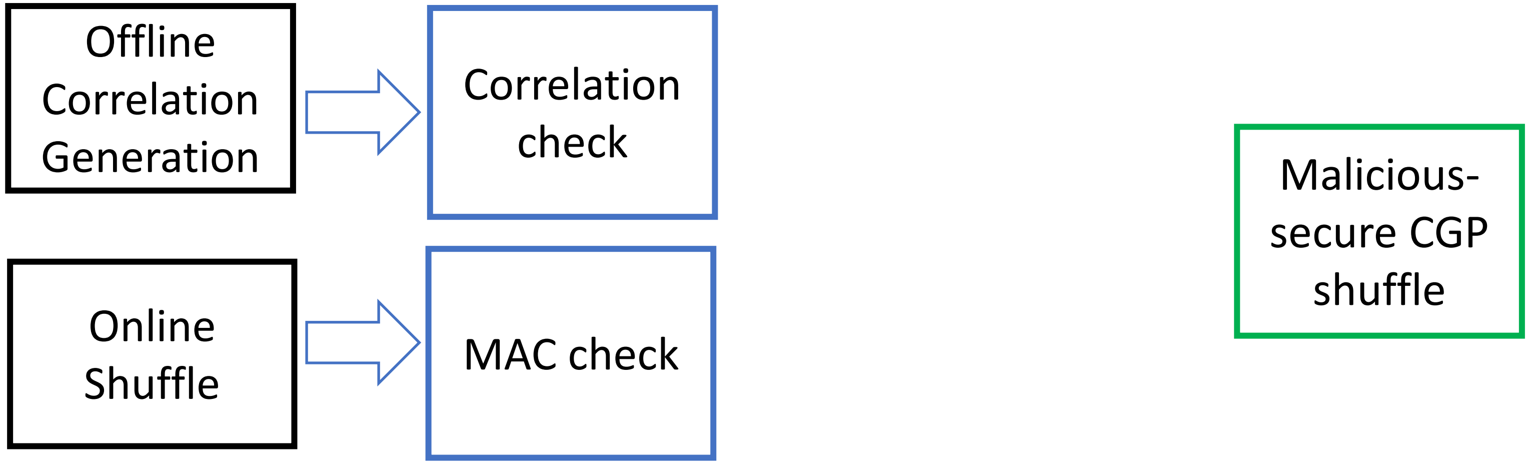
Our Solution

Offline Correlation Generation

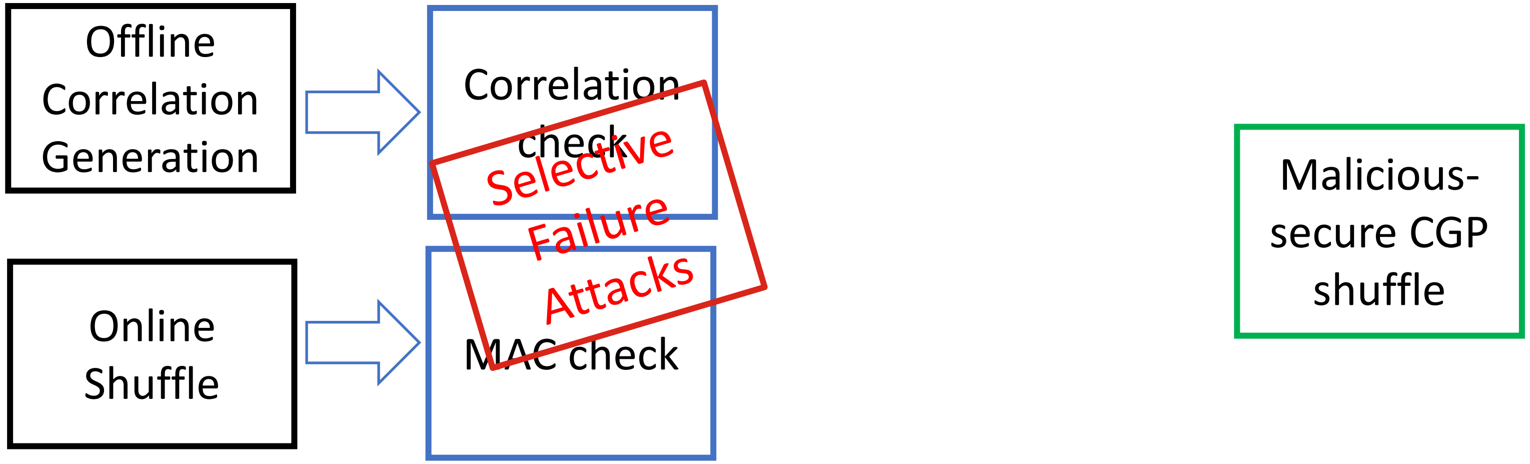
Online Shuffle

Malicious-secure CGP shuffle

Our Solution



Our Solution

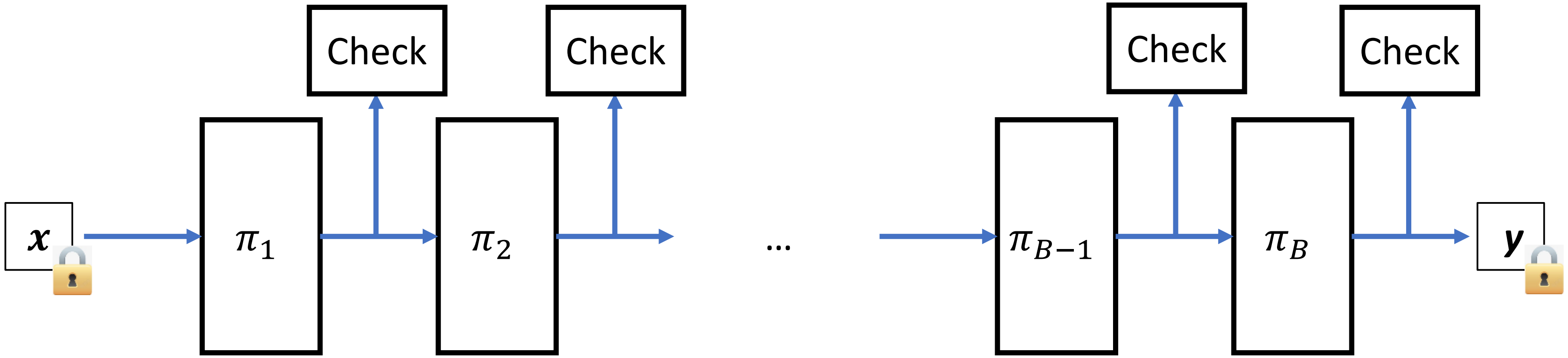


Leakage reduction

- Two kinds of selective-failure attack
  - Offline Selective-failure attacks (from correlation check)
  - Online Selective-failure attacks (from MAC check)

Leakage reduction

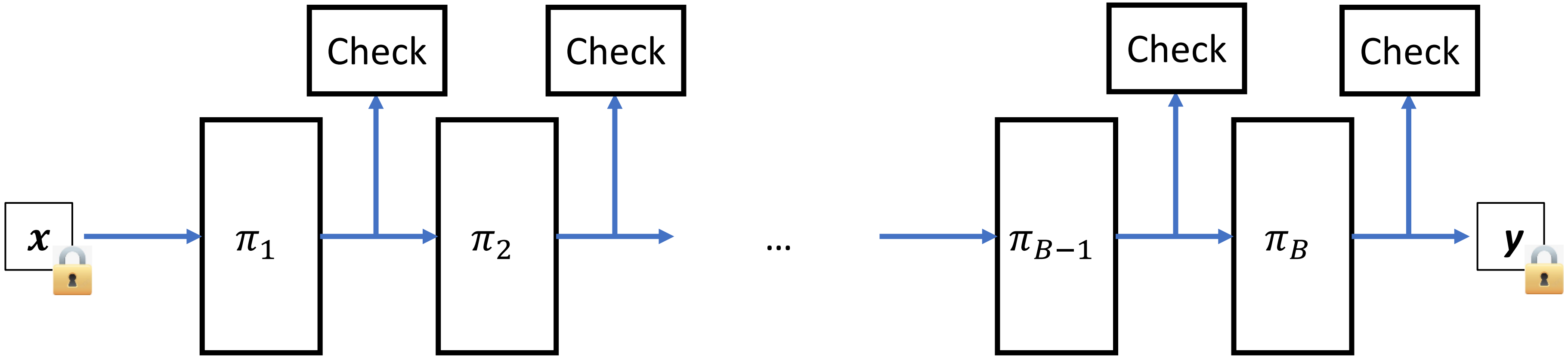
- Two kinds of selective-failure attack
  - Offline Selective-failure attacks (from correlation check)
  - Online Selective-failure attacks (from MAC check)
- Intuition: repeated shuffle + check



- $\boldsymbol{y} = \pi_B \circ \pi_{B-1} \circ \cdots \circ \pi_2 \circ \pi_1(\boldsymbol{x})$

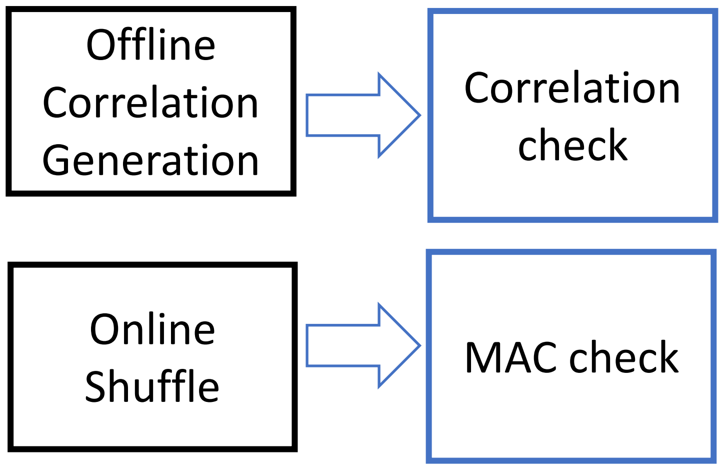
Leakage reduction

- Two kinds of selective-failure attack
  - Offline Selective-failure attacks (from correlation check)
  - Online Selective-failure attacks (from MAC check)
- Intuition: repeated shuffle + check

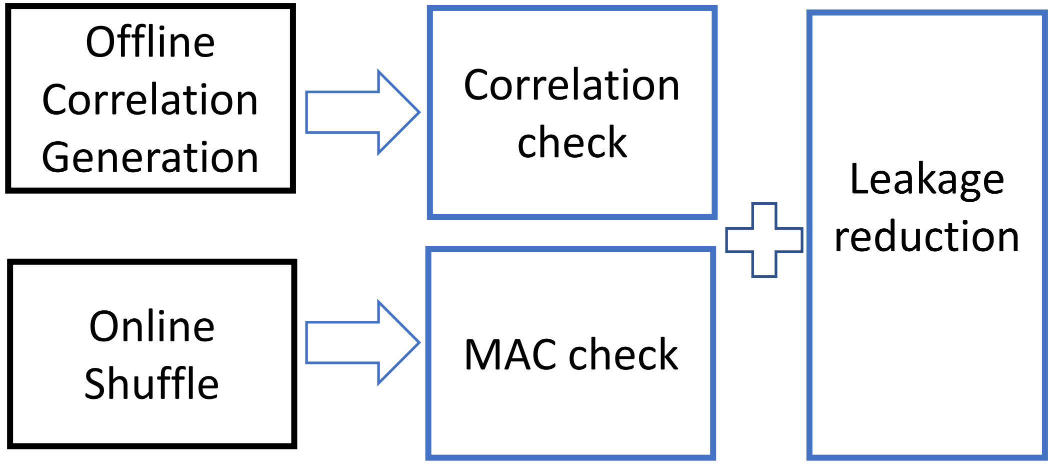


- $\boldsymbol{y} = \pi_B \circ \pi_{B-1} \circ \cdots \circ \pi_2 \circ \pi_1(\boldsymbol{x})$
- Our contribution
  - A new cut-and-choose leakage reduction mechanism
  - A new combinatorial analysis method for the cut-and-choose game

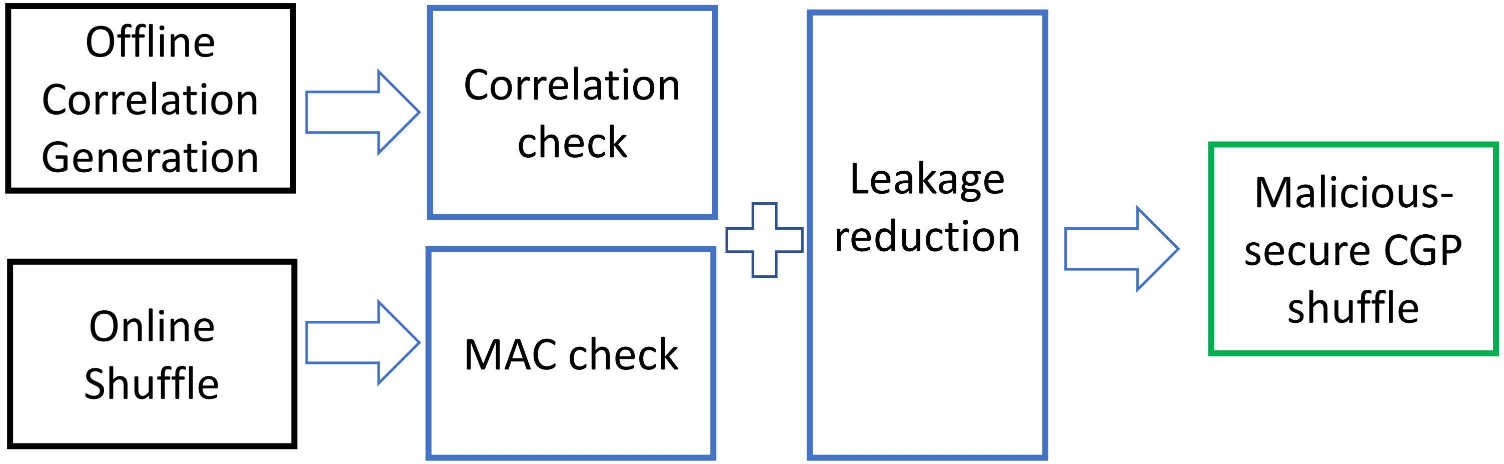
Summary of Roadmap



Summary of Roadmap



Summary of Roadmap



Performance: Correlation Generation Phase

- Setting
  - LAN: 0.2 ms RTT, 1 Gbps
  - WAN: 80 ms RTT, 40 Mbps
- Optimizations
  - Generalized Benes Networks [CGP20]
  - Decompose a big permutation into many small ones
- Running time
  - 1.1-2.9x slower in the LAN setting
  - 1.01-2.3x slower in the WAN setting
- Communication
  - 20% more communication

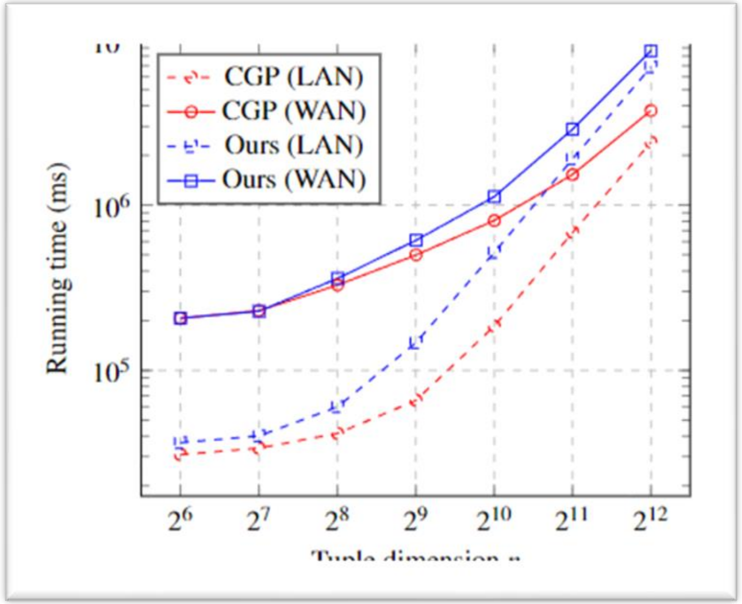


Fig. 1 Running time for correlation generation

Protocol	2^6	2^7	2^8	2^9	2^{10}	2^{11}	2^{12}
CGP	0.031	0.056	0.111	0.234	0.504	1.094	2.372
Ours	0.037	0.066	0.119	0.246	0.525	1.131	2.442

Tab. 1 Communication overhead for correlation generation (MB)

[CGP20] M. Chase, E. Ghosh, and O. Poburinnaya, “Secret-shared shuffle,” ASIACRYPT 2020.

Performance: Shuffle Phase

Protocol	LAN						WAN					
	2^{10}	2^{12}	2^{14}	2^{16}	2^{18}	2^{20}	2^{10}	2^{12}	2^{14}	2^{16}	2^{18}	2^{20}
Ours (2^4)	**0.36**	**1.22**	**6.29**	**24.78**	**132.43**	**507.21**	6.56	12.35	44.51	155.06	761.93	**3,030.09**
Ours (2^6)	0.48	1.78	11.11	44.70	178.15	986.47	4.89	**9.92**	**42.88**	147.25	**571.71**	3,213.94
Ours (2^8)	1.30	5.40	20.41	79.67	553.30	2,126.90	5.39	13.37	83.60	**145.72**	938.52	4,005.16
Ours (2^{10})	1.68	18.44	76.84	273.84	1,125.21	4,578.25	**4.33**	26.49	91.04	345.30	1,365.92	5,853.09
[4] (OT)	9.53	45.27	211.41	1,077.76	4,209.90	-	125.21	593.10	2,769.98	12,706.00	-	-
[4] (HE)	4.73	17.81	79.36	357.64	1,610.24	-	59.26	90.71	427.87	1,978.11	9,056.06	-

Tab. 2 Amortized offline running time (s)

Protocol	Offline						Online					
	2^{10}	2^{12}	2^{14}	2^{16}	2^{18}	2^{20}	2^{10}	2^{12}	2^{14}	2^{16}	2^{18}	2^{20}
Ours (2^4)	7.03	27.8	155.17	620.34	3,189.88	12,759.18	1.06	4.26	23.86	95.42	490.73	1,962.93
Ours (2^6)	4.76	**18.76**	124.65	498.16	1,992.38	11,156.93	0.49	1.97	13.11	52.43	209.72	1,174.41
Ours (2^8)	4.98	19.73	**78.71**	**314.64**	2,097.22	8,388.68	0.39	1.57	6.29	25.17	167.77	671.09
Ours (2^{10})	**1.84**	21.39	85.39	341.38	**1,365.31**	**5,461.04**	**0.11**	**1.38**	**5.51**	**22.02**	**88.08**	**352.32**
[4] (OT)	1,757.43	8,561.07	40,193.90	184,763.00	835,816.00	-	1.37	6.65	31.33	144.18	652.22	2,910.86
[4] (HE)	123.89	497.72	1,115.25	5,050.48	22,832.60	-	1.37	6.65	31.33	144.18	652.22	2,910.86

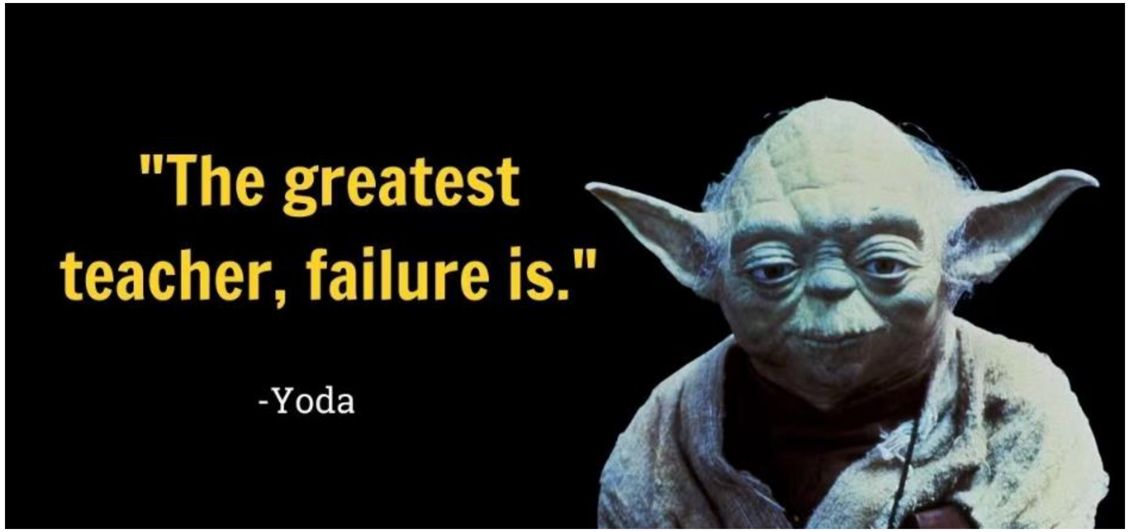
Tab. 3 Amortized communication (MB) for offline and online phases

- Compared with the SSS protocol from MP-SPDZ library [Kel20]
  - ~15x faster in the offline phase
  - ~7x faster in the online shuffle phase

[Kel20] M. Keller, “MP-SPDZ: A versatile framework for multiparty computation,” ACM CCS 2020

Conclusion

- Existing CGP shuffle protocols with malicious security are flawed
- Designing maliciously secure CGP shuffle protocol is non-trivial
- We propose correlation check and leakage reduction mechanisms to enable maliciously secure CGP shuffle protocol
- While increasing security, our enhancement introduces low overhead.



Thanks for your attentions!