On Compressing Linearly Shared Correlations

TCC 2026

Geoffroy Couteau, Alexander Koch, Nikolas Melissaris,
Peter Scholl, **Sacha Servan-Schreiber,** Xiaxi Ye














Pseudorandom Correlations[BCGI'18]

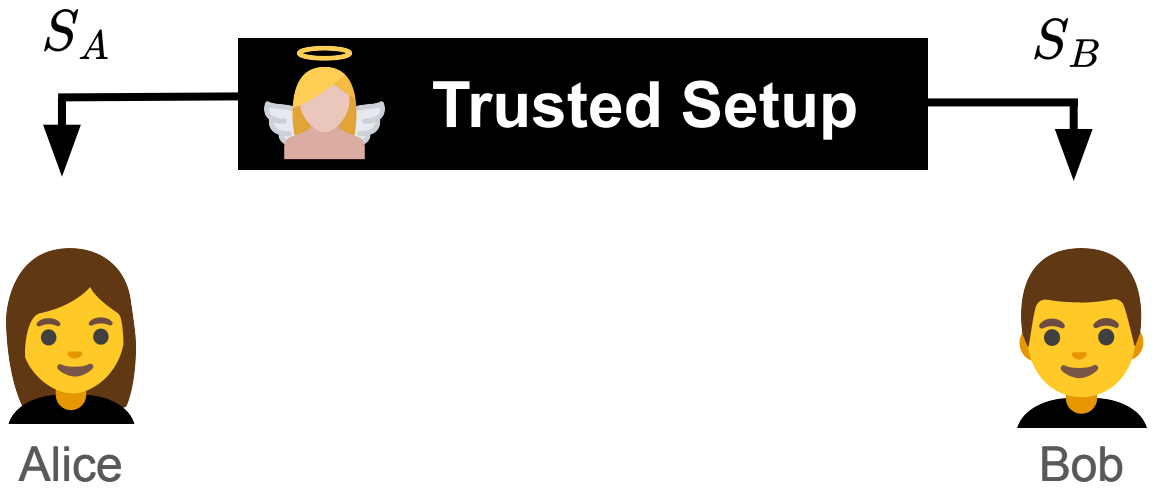


Alice

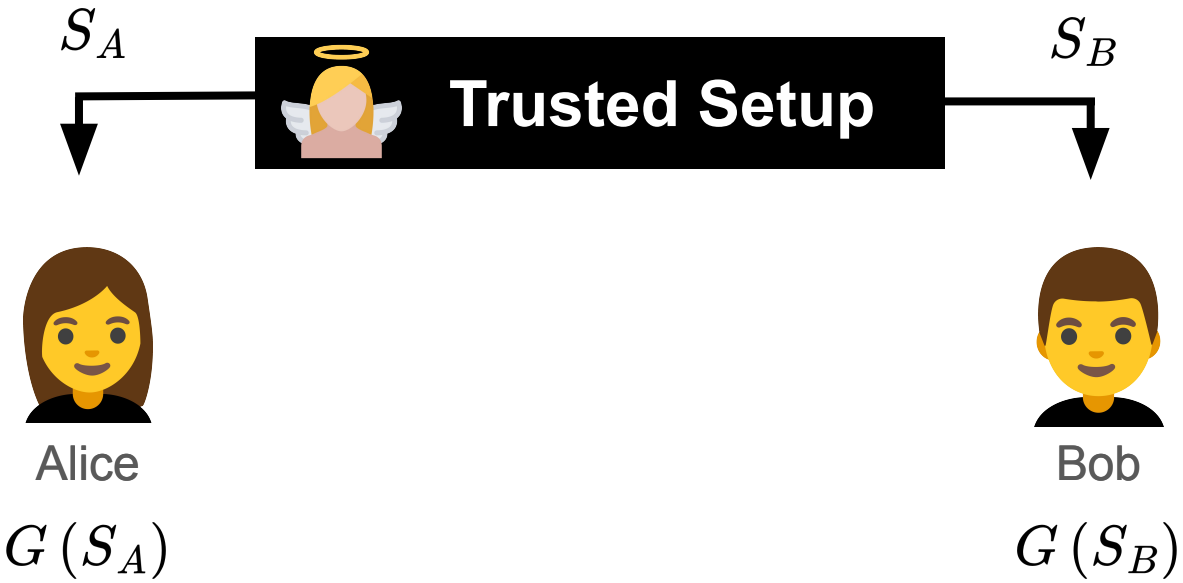


Bob

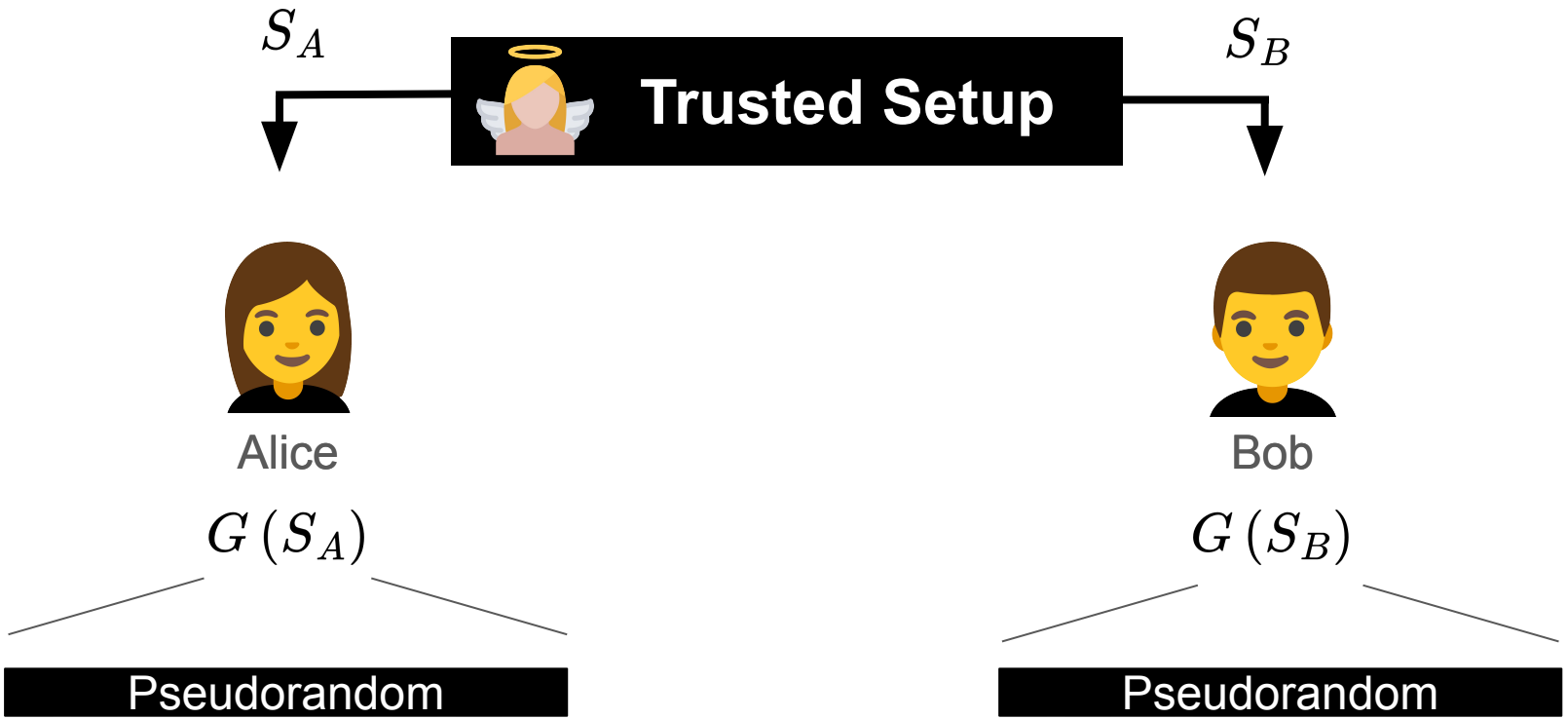
Pseudorandom Correlations[BCGI'18]



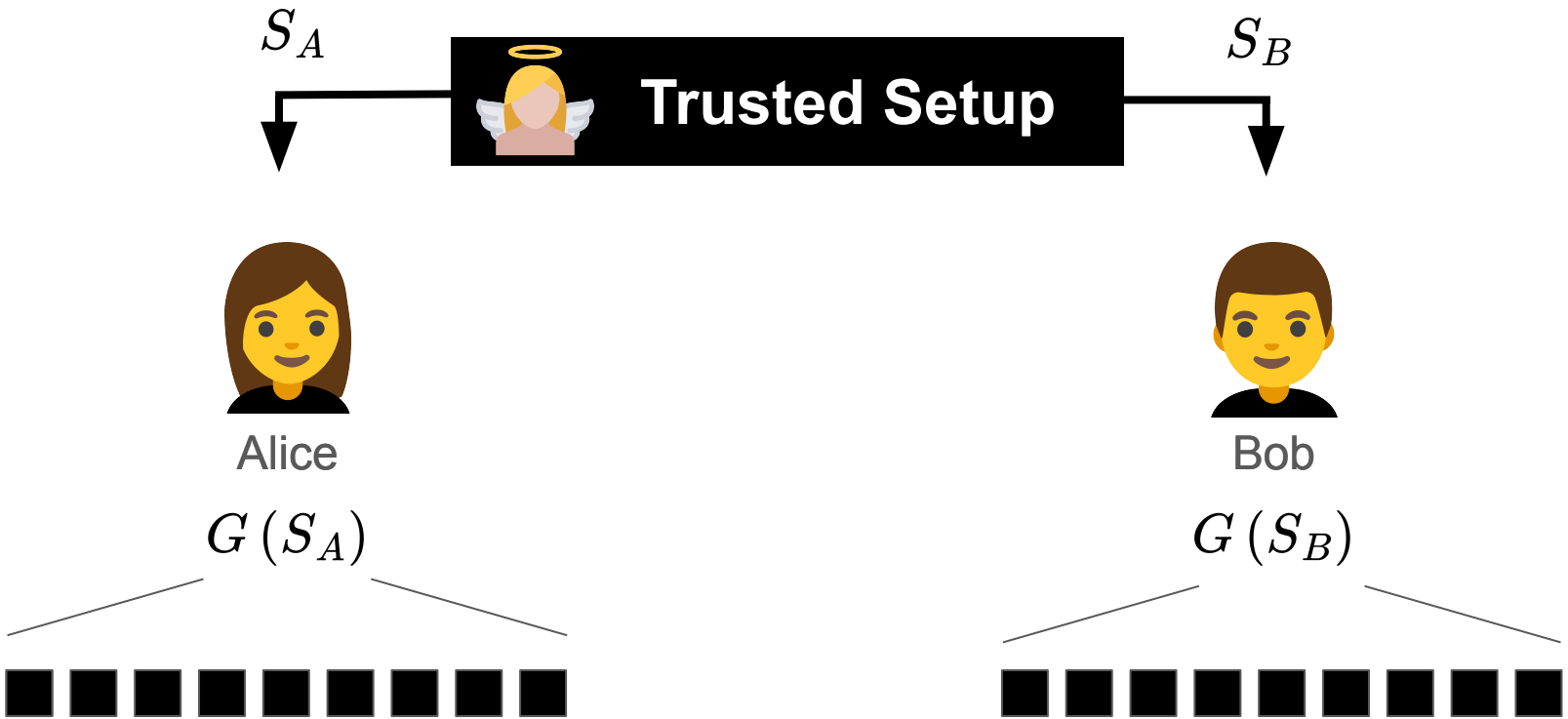
Pseudorandom Correlations[BCGI'18]



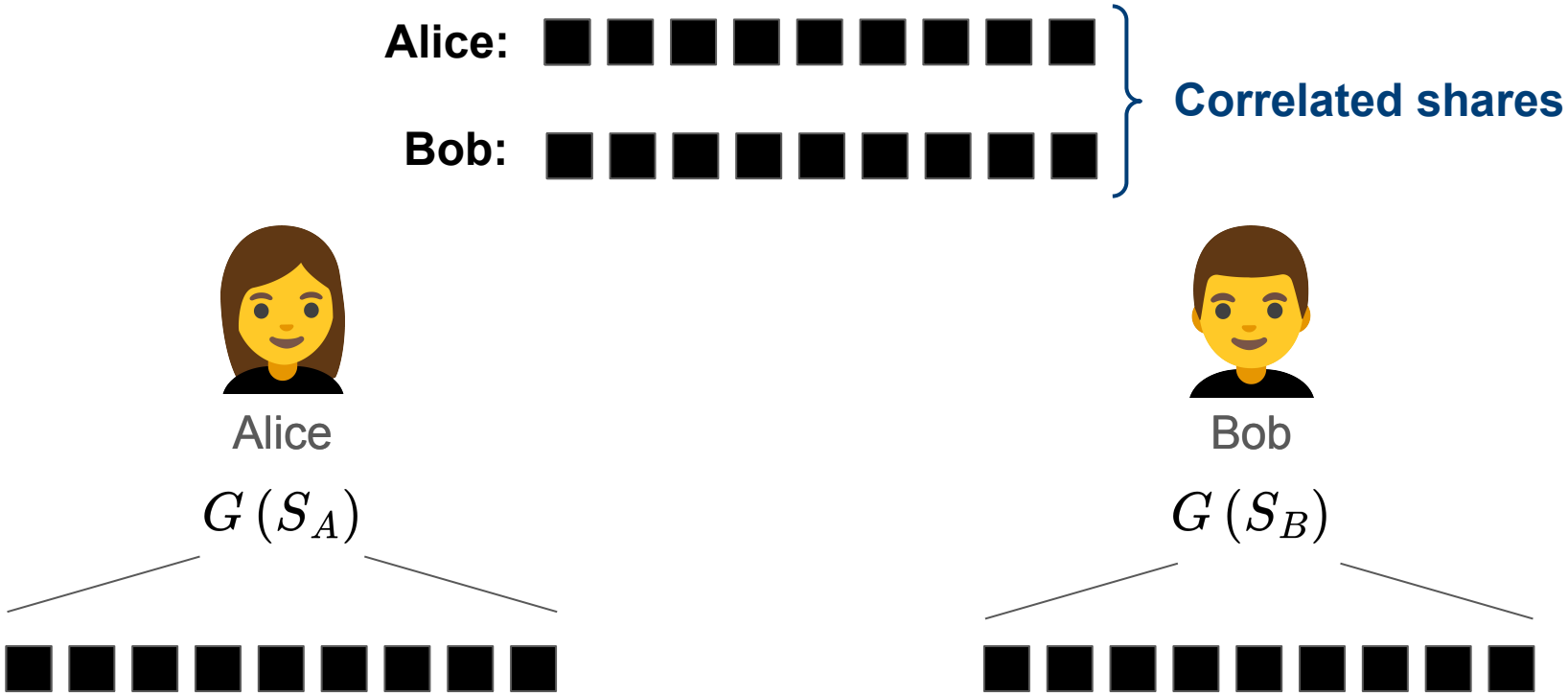
Pseudorandom Correlations[BCGI'18]



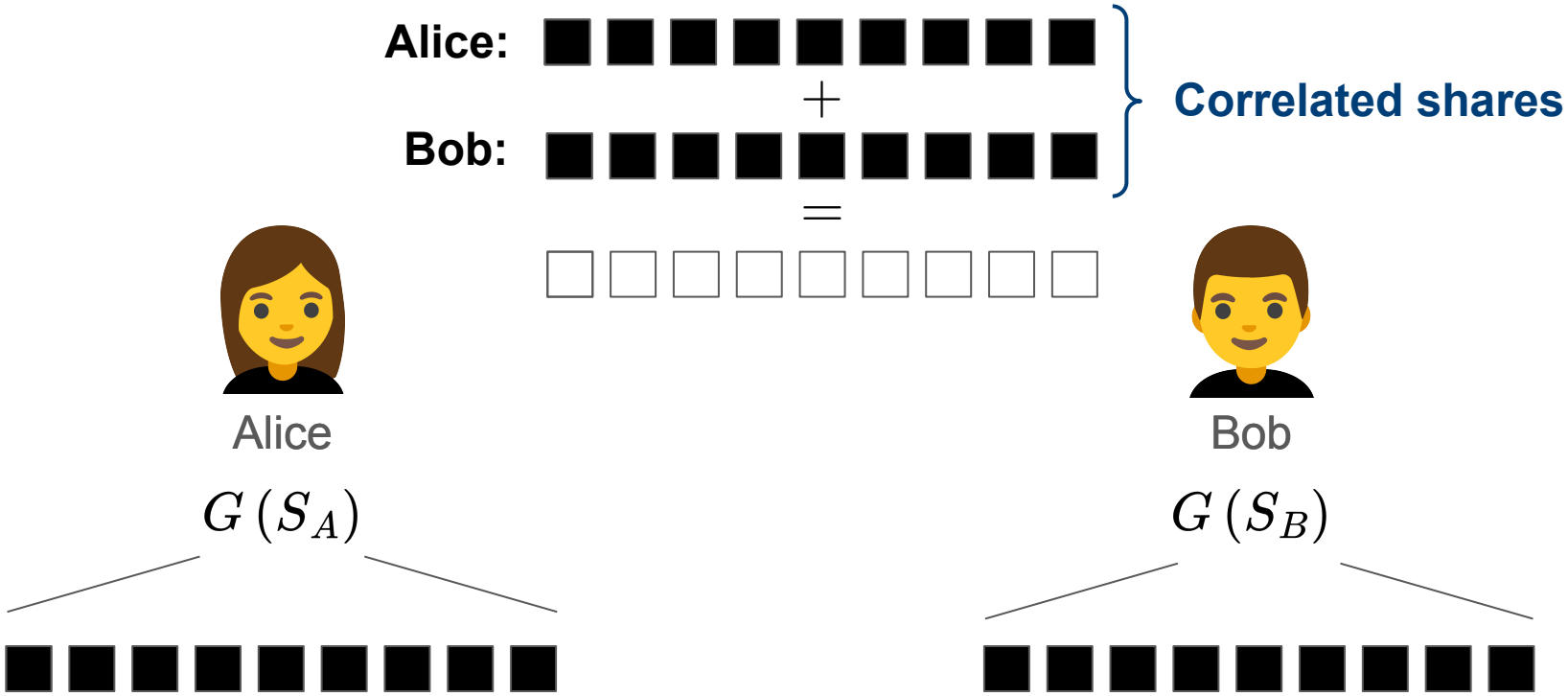
Pseudorandom Correlations[BCGI'18]



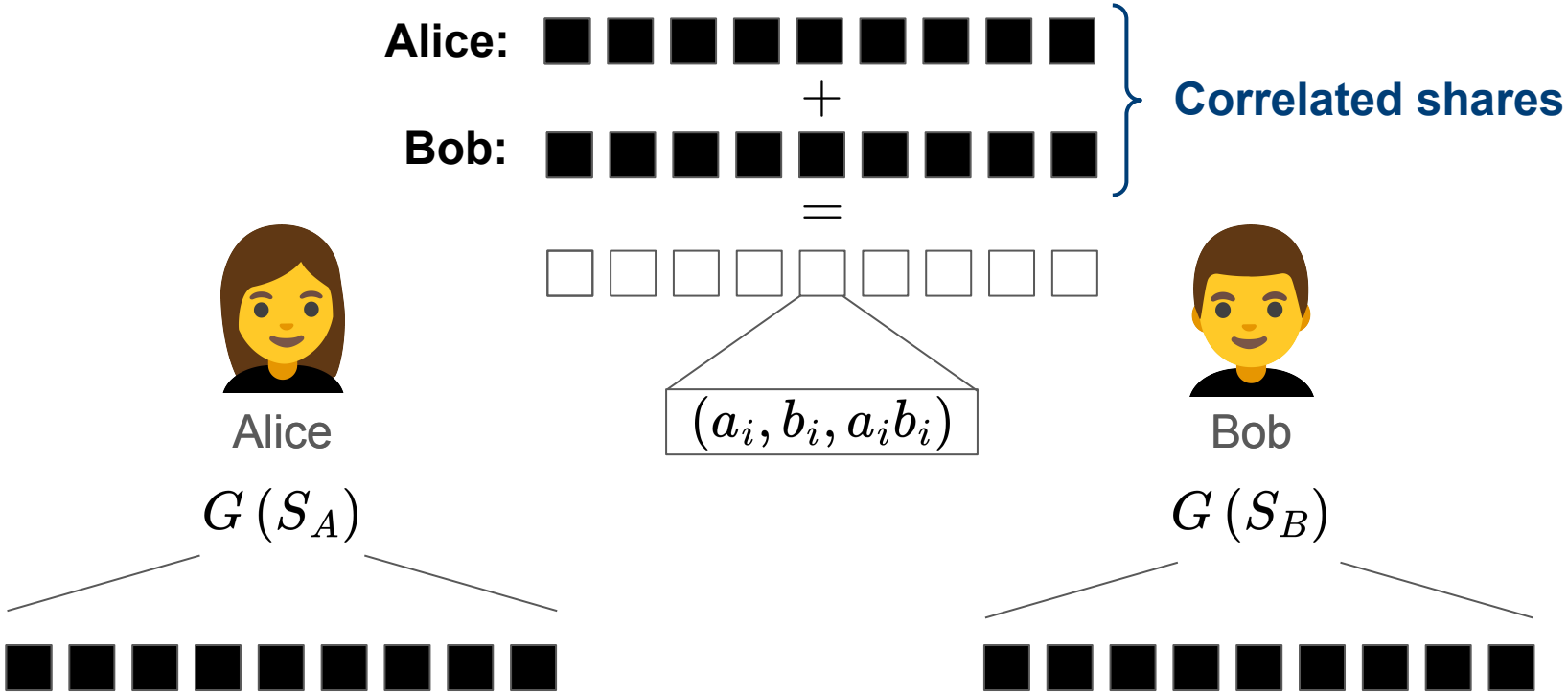
Pseudorandom Correlations[BCGI'18]



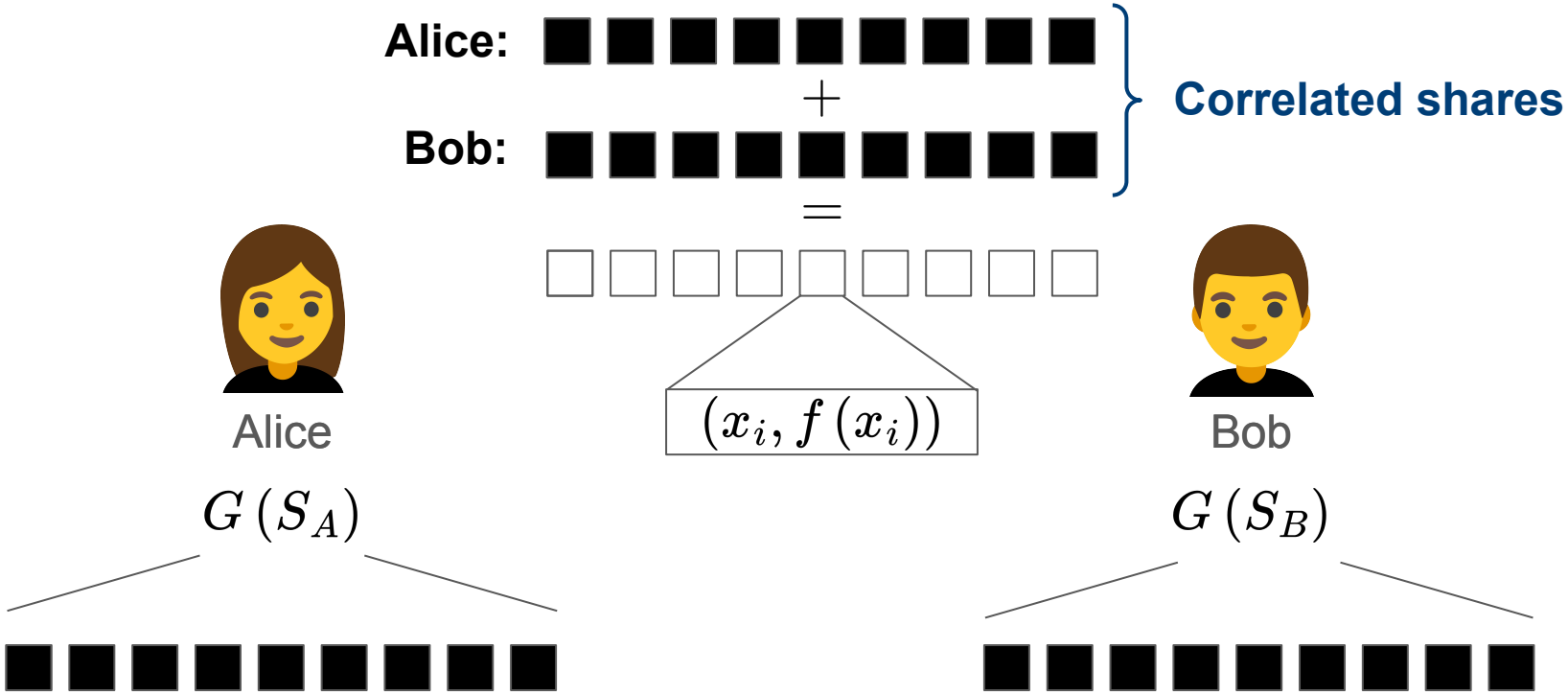
Pseudorandom Correlations[BCGI'18]



Pseudorandom Correlations[BCGI'18]



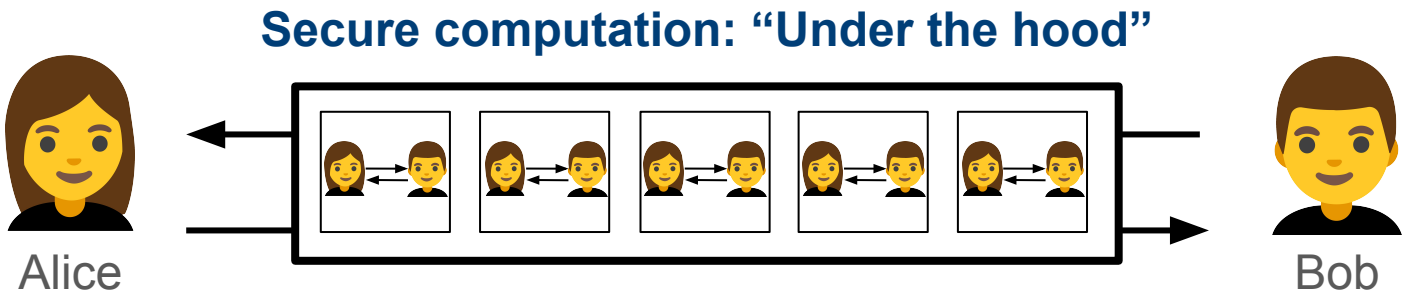
Pseudorandom Correlations[BCGI'18]



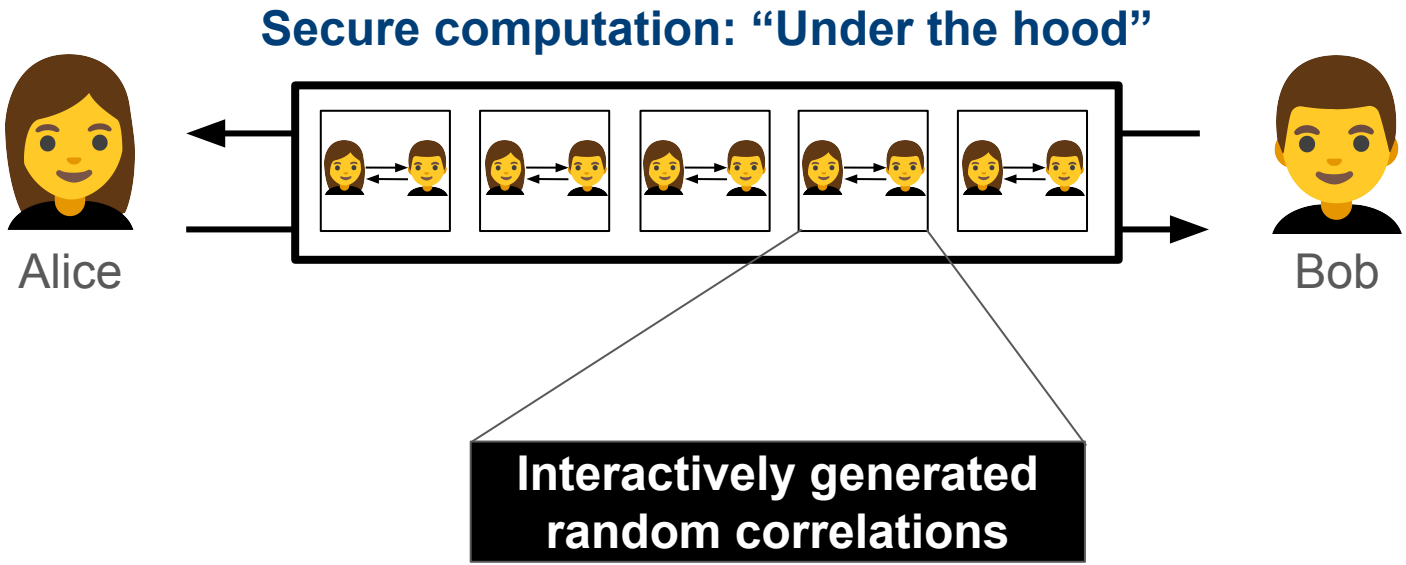
Application: secure computation



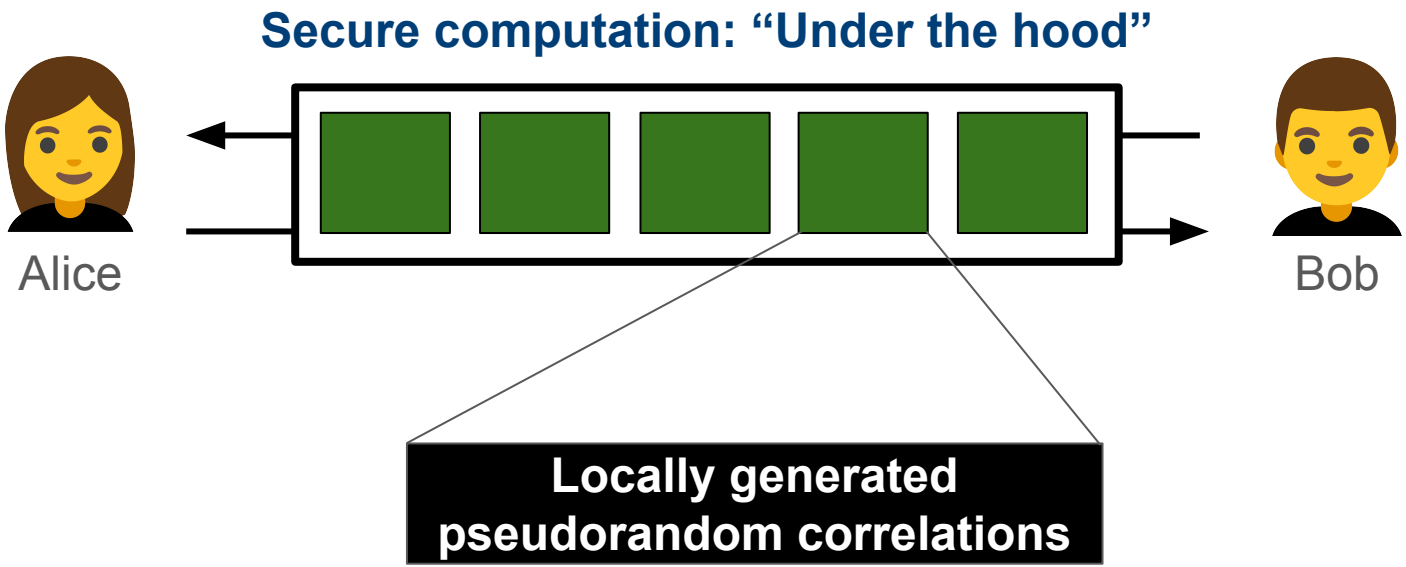
Application: secure computation



Application: secure computation



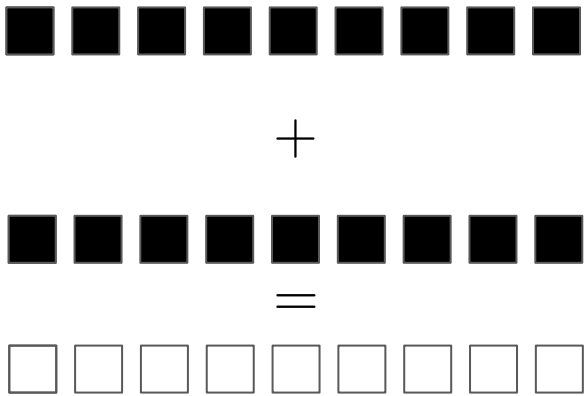
Application: secure computation



Existing Pseudorandom Correlations

[BCGIO'17][BCGI'18][BCGIKS'19] and followups.

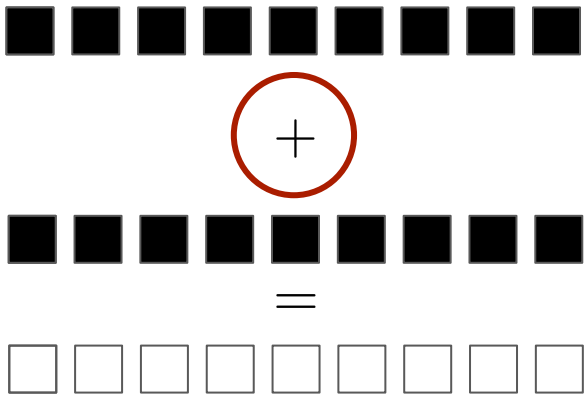
Existing constructions focus on **additive** 2-party correlations



Existing Pseudorandom Correlations

[BCGIO'17][BCGI'18][BCGIKS'19] and followups.

Existing constructions focus on **additive** 2-party correlations

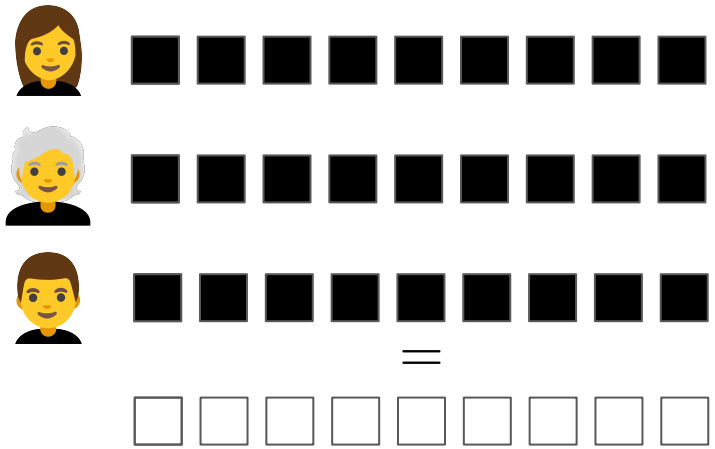


What about non-additive correlations?

For example: Shamir Secret Shares of a target correlation?

What about non-additive correlations?

For example: Shamir Secret Shares of a target correlation?



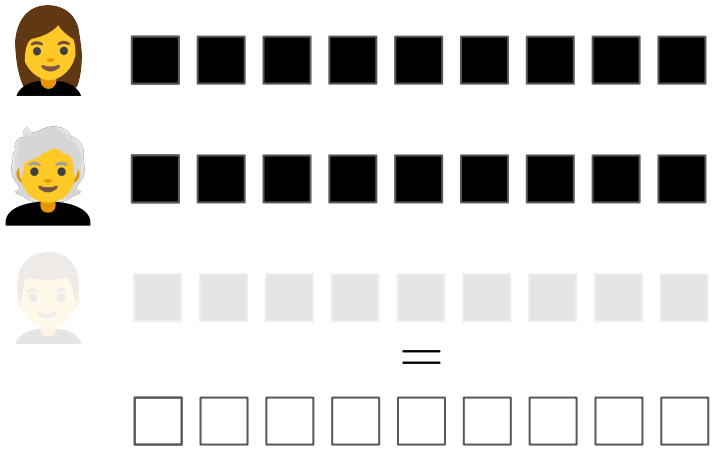
What about non-additive correlations?

For example: Shamir Secret Shares of a target correlation?



What about non-additive correlations?

For example: Shamir Secret Shares of a target correlation?

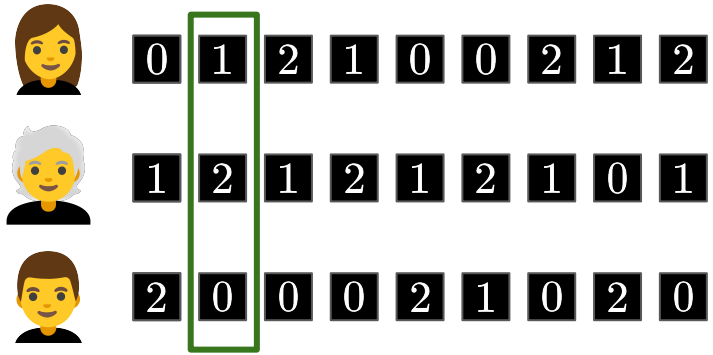


What about non-additive correlations?

For example: Pseudorandom Permutations? (each party gets $\pi(i)$ for a random π)

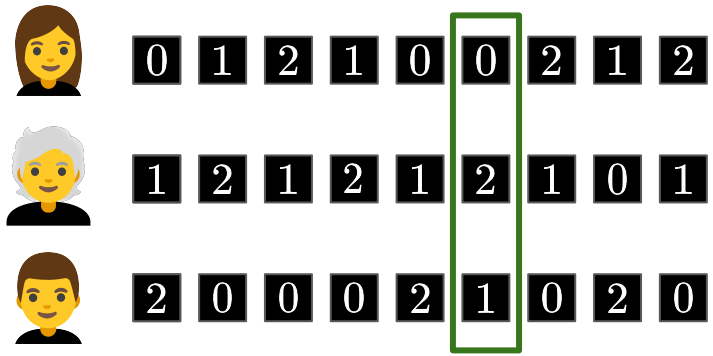
What about non-additive correlations?

For example: Pseudorandom Permutations? (each party gets $\pi(i)$ for a random π)



What about non-additive correlations?

For example: Pseudorandom Permutations? (each party gets $\pi(i)$ for a random π)



Why non-additive correlations?

Shamir Secret Shares of a correlation:

Unlocks silent preprocessing for *threshold* multi-party computation.

Why non-additive correlations?

Shamir Secret Shares of a correlation:

Unlocks silent preprocessing for *threshold* multi-party computation.

Pseudorandom permutations:

Enables more bandwidth-efficient anonymous broadcast protocols.

> ***"There is no good non-interactive means of enabling all players to select distinct message position."* — Golle & Juels, EUROCRYPT 2004**

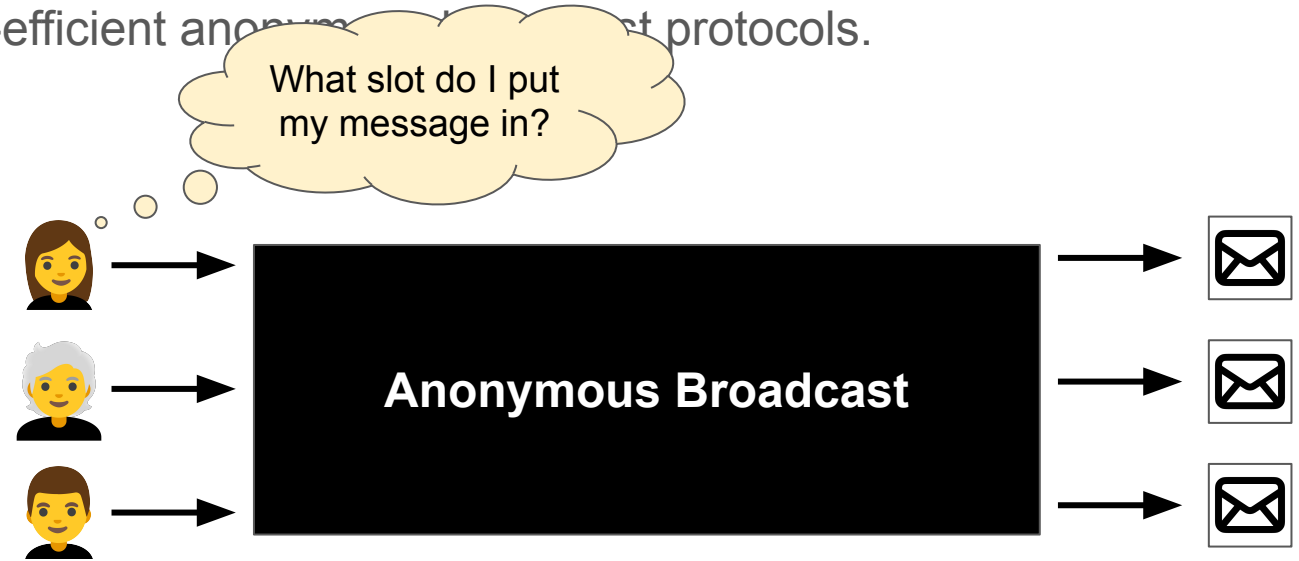
Why non-additive correlations?

Local sampling of pseudorandom permutations enables more bandwidth-efficient anonymous broadcast protocols.



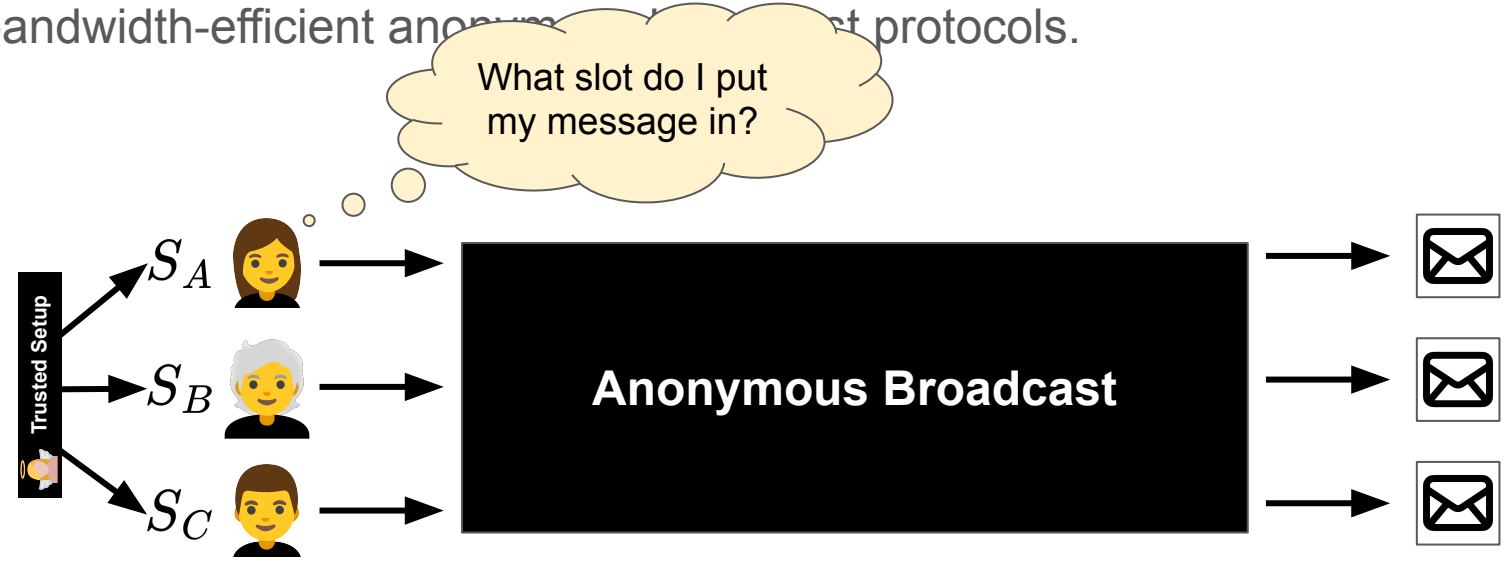
Why non-additive correlations?

Local sampling of pseudorandom permutations enables more bandwidth-efficient anonym[illegible]t protocols.



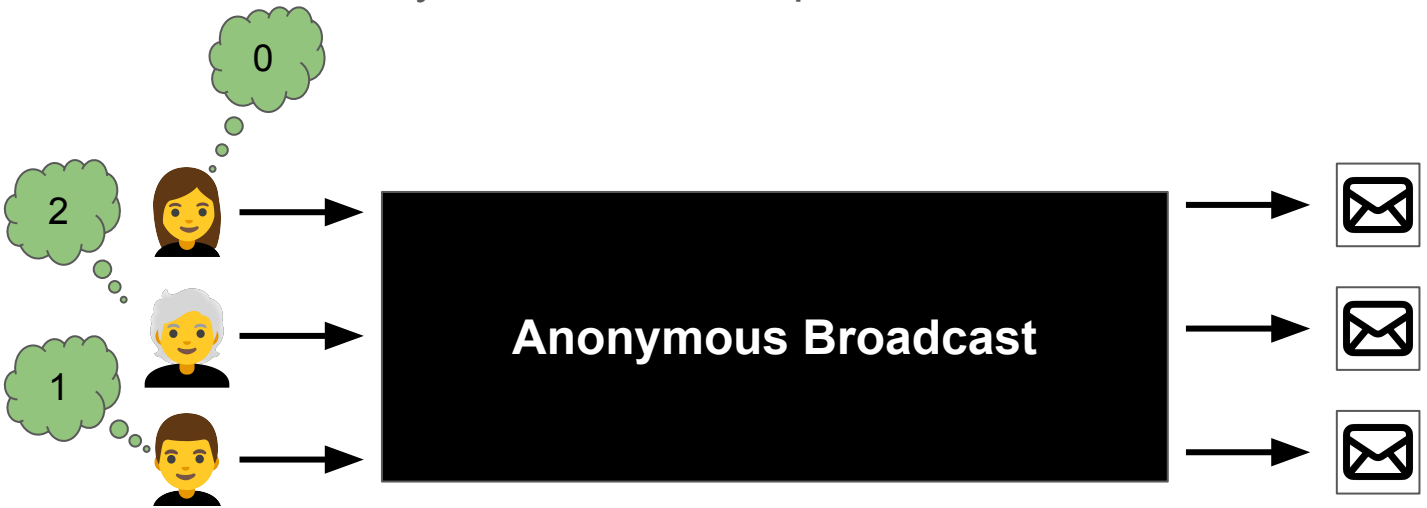
Why non-additive correlations?

Local sampling of pseudorandom permutations enables more bandwidth-efficient anonym[illegible]t protocols.



Why non-additive correlations?

Local sampling of pseudorandom permutations enables more bandwidth-efficient anonymous broadcast protocols.



Building correlation generators for *non-additive* correlations

Tool: Homomorphic Secret Sharing [BGI'16]

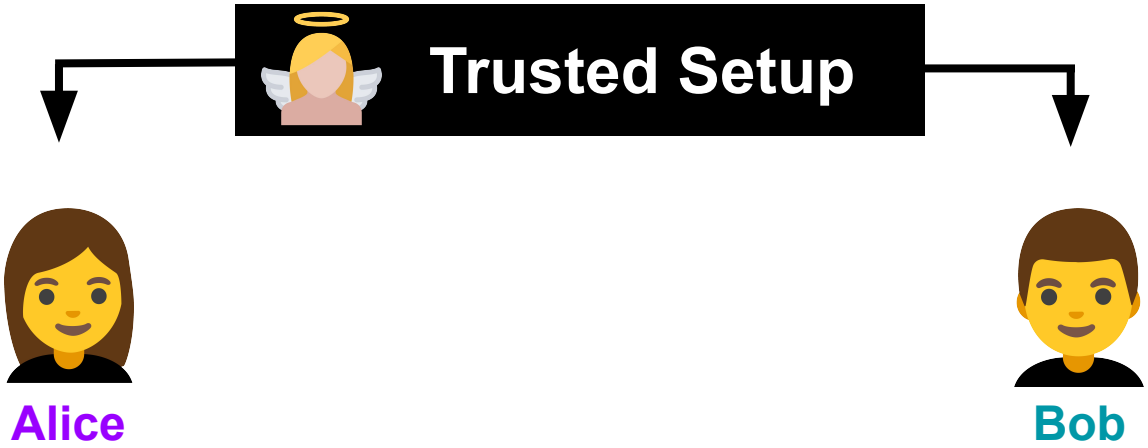


Alice



Bob

Tool: Homomorphic Secret Sharing [BGI'16]



Tool: Homomorphic Secret Sharing [BGI'16]



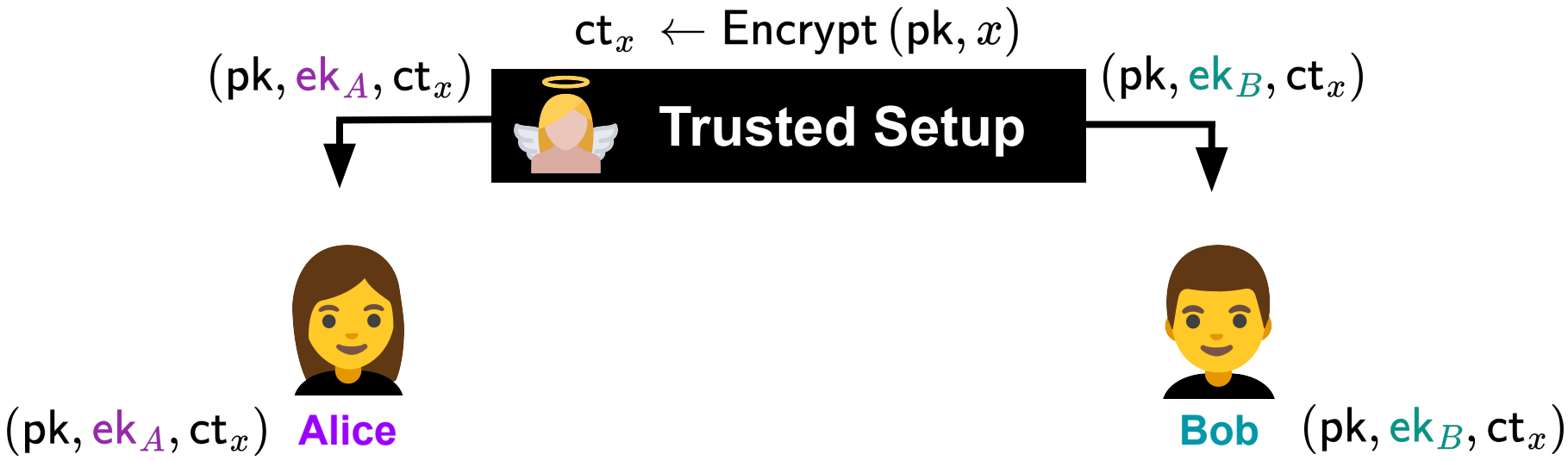
Tool: Homomorphic Secret Sharing [BGI'16]



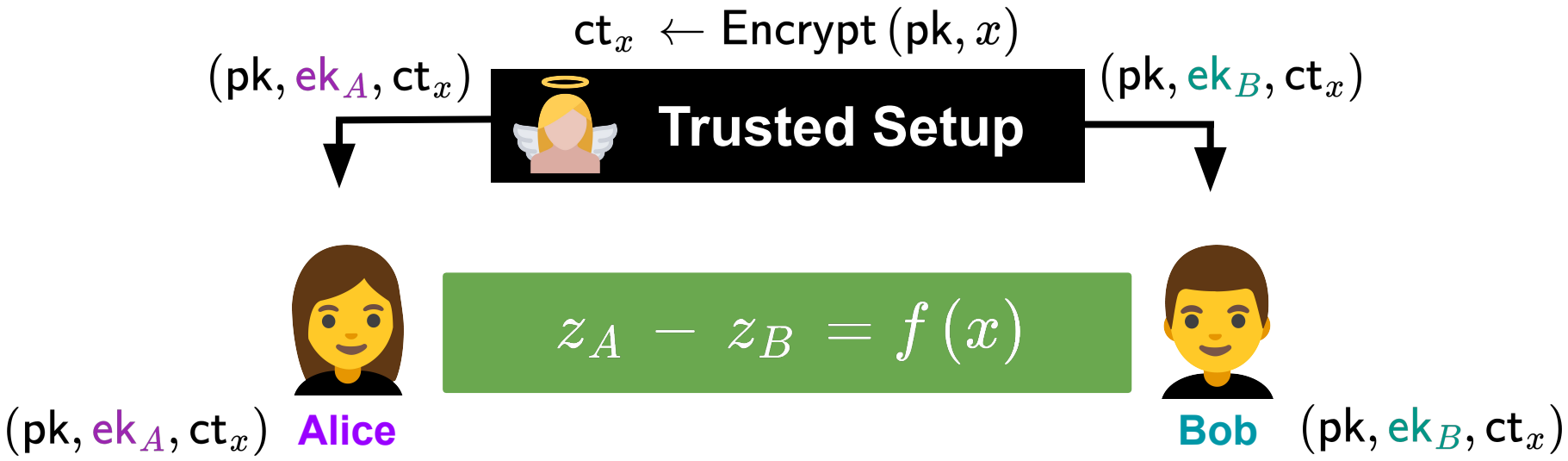
Tool: Homomorphic Secret Sharing [BGI'16]



Tool: Homomorphic Secret Sharing [BGI'16]



Tool: Homomorphic Secret Sharing [BGI'16]

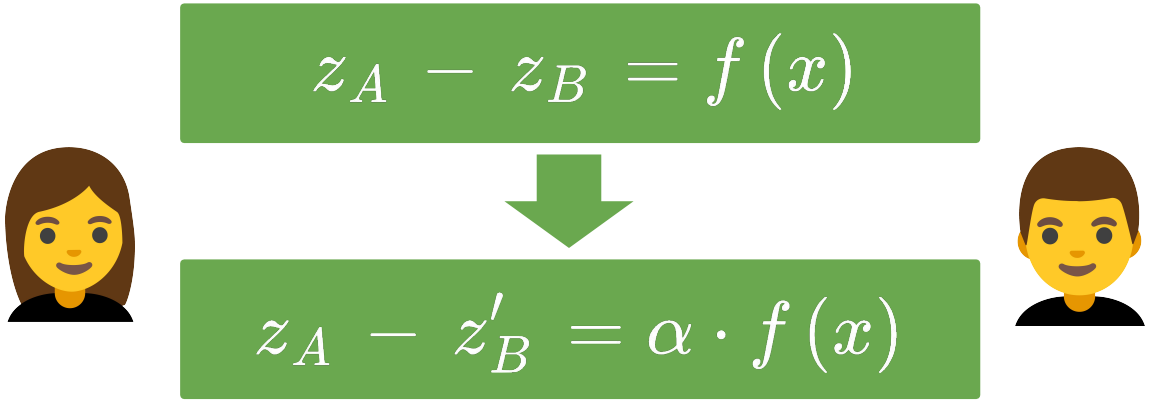


$z_A \leftarrow \mathsf{Eval}(\mathsf{pk}, f, \mathsf{ek}_A, \mathsf{ct}_x)$

$z_B \leftarrow \mathsf{Eval}(\mathsf{pk}, f, \mathsf{ek}_B, \mathsf{ct}_x)$

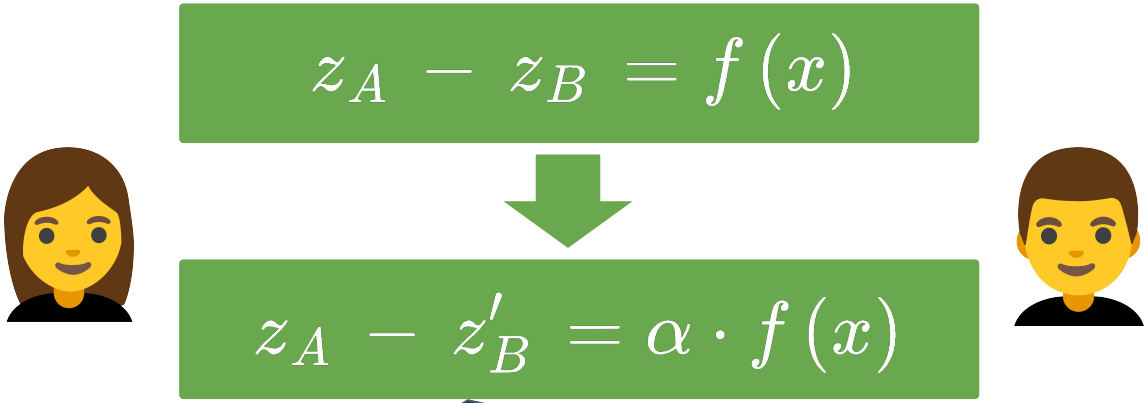
Tool: Homomorphic Secret Sharing [BGI'16]

With Simulatable Memory Shares[CMPR'23]



Tool: Homomorphic Secret Sharing [BGI'16]

With Simulatable Memory Shares[CMPR'23]



Bob's memory share can be programmed to produce a scaled output for public α!

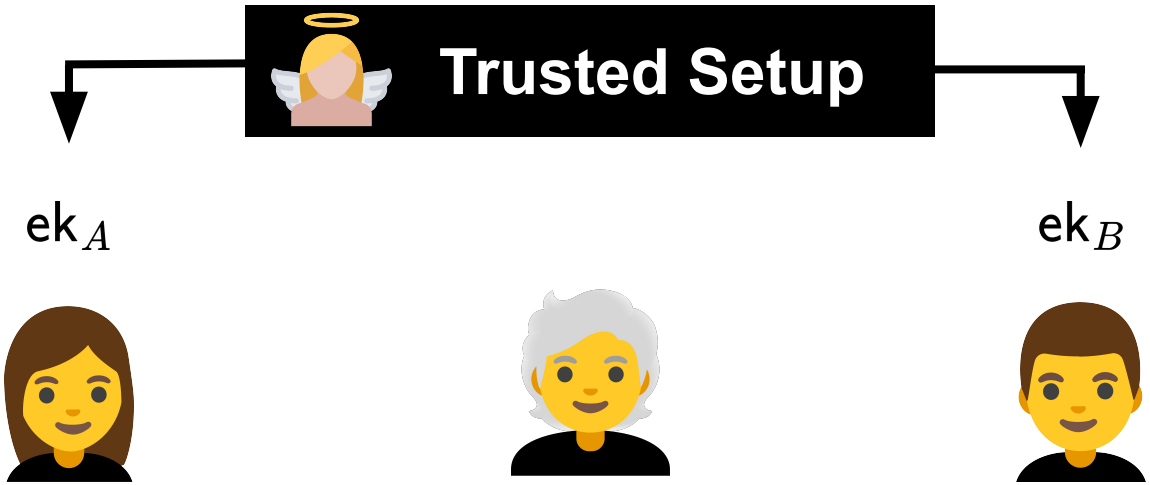
Constructing HSS for Degree-1 Shamir Shares

From HSS with Simulatable Memory Shares

Main Idea: manipulate and reframe the values of the share held by each party

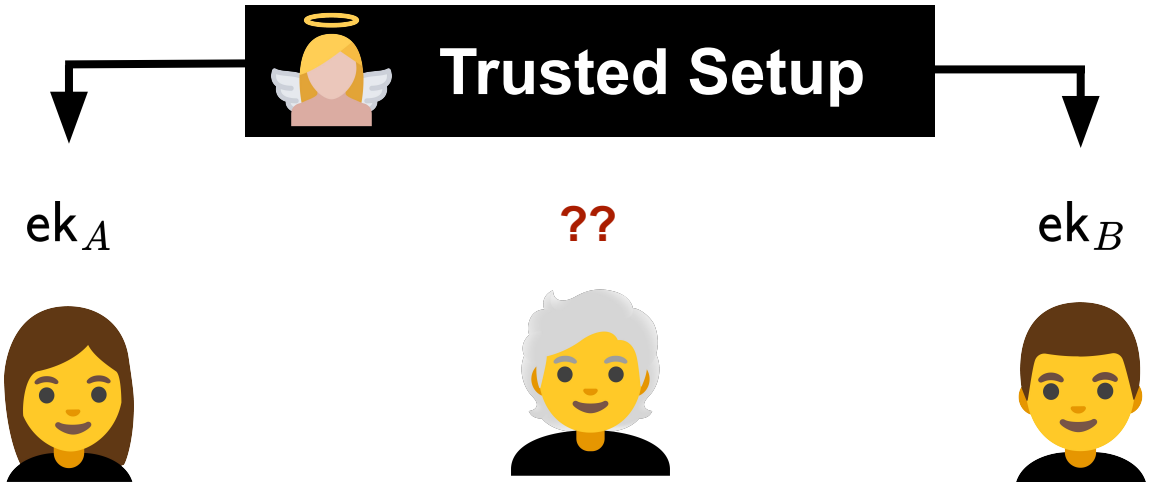
Constructing HSS for Degree-1 Shamir Shares

From HSS with Simulatable Memory Shares



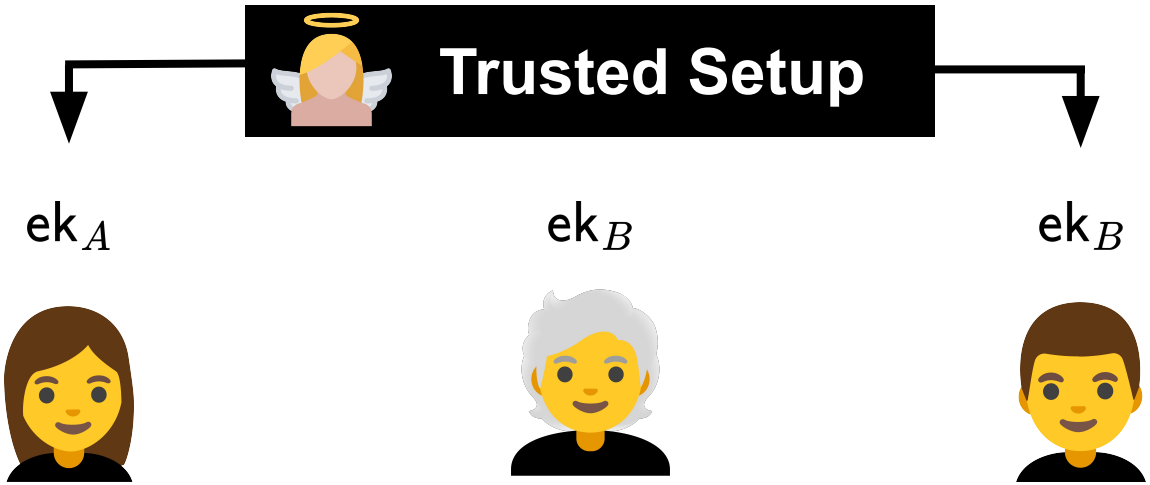
Constructing HSS for Degree-1 Shamir Shares

From HSS with Simulatable Memory Shares



Constructing HSS for Degree-1 Shamir Shares

From HSS with Simulatable Memory Shares



Focusing on *Degree-1* Shamir Shares

Constructing HSS for Degree-1 Shamir Shares

From HSS with Simulatable Memory Shares

ek_A 

ek_B 

ek_B 

Constructing HSS for Degree-1 Shamir Shares

From HSS with Simulatable Memory Shares

ek_A 

$z_A \leftarrow \mathsf{Eval}(\mathsf{pk}, f, \mathsf{ek}_A, \mathsf{ct}_x)$

ek_B 

$z_B \leftarrow \mathsf{Eval}(\mathsf{pk}, f, \mathsf{ek}_B, \mathsf{ct}_x)$

ek_B 

$z_B \leftarrow \mathsf{Eval}(\mathsf{pk}, f, \mathsf{ek}_B, \mathsf{ct}_x)$

Same evaluation!

Constructing HSS for Degree-1 Shamir Shares

From HSS with Simulatable Memory Shares

ek_A 

$M_A \leftarrow \mathsf{Sim}_0\left(1^\lambda\right)$ Sampled independently of any α

ek_B 

$M_B^{(1)} \leftarrow \mathsf{Sim}_1\left(M_A, \alpha_1\right)$

ek_B 

$M_B^{(2)} \leftarrow \mathsf{Sim}_1\left(M_A, \alpha_2\right)$

Same evaluation key, but different memory shares!

Constructing HSS for Degree-1 Shamir Shares

From HSS with **Simulatable Memory Shares**

ek_A 

$$z_A \leftarrow \mathsf{Eval}\,(\mathsf{pk},\ f,\ \mathsf{ek}_A, \mathsf{ct}_x)$$

ek_B 

~~$z_B \leftarrow \mathsf{Eval}\,(\mathsf{pk},\ f,\ \mathsf{ek}_B, \mathsf{ct}_x)$~~

$$z_B^{(1)} \leftarrow \mathsf{Eval}'\,(\mathsf{pk},\ f,\ \mathsf{ek}_B, \mathsf{ct}_x, \alpha_1)$$

ek_B 

~~$z_B \leftarrow \mathsf{Eval}\,(\mathsf{pk},\ f,\ \mathsf{ek}_B, \mathsf{ct}_x)$~~

$$z_B^{(2)} \leftarrow \mathsf{Eval}'\,(\mathsf{pk},\ f,\ \mathsf{ek}_B, \mathsf{ct}_x, \alpha_2)$$

Use simulatable memory shares with constants α_i

$$\alpha_i = \frac{i}{i+1}$$

Constructing HSS for Degree-1 Shamir Shares

From HSS with Simulatable Memory Shares



ek_A

$z_A \leftarrow \mathsf{Eval}(\mathsf{pk}, f, \mathsf{ek}_A, \mathsf{ct}_x)$

z_A



ek_B

~~$z_B \leftarrow \mathsf{Eval}(\mathsf{pk}, f, \mathsf{ek}_B, \mathsf{ct}_x)$~~

$z_B^{(1)} \leftarrow \mathsf{Eval}'(\mathsf{pk}, f, \mathsf{ek}_B, \mathsf{ct}_x, \alpha_1)$

Scale each party's share locally



ek_B

~~$z_B \leftarrow \mathsf{Eval}(\mathsf{pk}, f, \mathsf{ek}_B, \mathsf{ct}_x)$~~

$z_B^{(2)} \leftarrow \mathsf{Eval}'(\mathsf{pk}, f, \mathsf{ek}_B, \mathsf{ct}_x, \alpha_2)$

$s_i \leftarrow (i+1) z_B^{(i)}$

Claim: z_A and $s_i \leftarrow (i+1)z_B^{(i)}$ are evaluations of the polynomial $q(\ell) = f(x) + \ell(z_A - f(x))$ on $\{1, 2, \ldots, n\}$

Claim: z_A and $s_i \leftarrow (i+1)z_B^{(i)}$ are evaluations of the polynomial $q(\ell) = f(x) + \ell(z_A - f(x))$ on $\{1, 2, \ldots, n\}$

$$q(1) = \cancel{f(x)} + z_A - \cancel{f(x)} = z_A$$ ✓

Claim: z_A and $s_i \leftarrow (i+1)z_B^{(i)}$ are evaluations of the polynomial $q(\ell) = f(x) + \ell(z_A - f(x))$ on $\{1, 2, \ldots, n\}$

$$q(1) = \cancel{f(x)} + z_A - \cancel{f(x)} = z_A \quad \checkmark$$

$$q(2) = 2z_A - f(x) = \cancel{2z_A} - \cancel{2}\left(\cancel{z_A} - z_B^{(1)}\right) = 2z_B^{(1)} \quad \checkmark$$

Claim: z_A and $s_i \leftarrow (i+1)z_B^{(i)}$ are evaluations of the polynomial $q(\ell) = f(x) + \ell(z_A - f(x))$ on $\{1, 2, \ldots, n\}$

$$q(1) = \cancel{f(x)} + z_A - \cancel{f(x)} = z_A \quad \checkmark$$

$$q(2) = 2z_A - f(x) = \cancel{2z_A} - \cancel{2}\left(\cancel{z_A} - z_B^{(1)}\right) = 2z_B^{(1)} \quad \checkmark$$

$$q(3) = 3z_A - 2f(x) = \cancel{3z_A} - \cancel{3}\left(\cancel{z_A} - z_B^{(2)}\right) = 3z_B^{(2)} \quad \checkmark$$

Claim: z_A and $s_i \leftarrow (i+1) z_B^{(i)}$ are evaluations of the polynomial $q(\ell) = f(x) + \ell(z_A - f(x))$ on $\{1, 2, \ldots, n\}$

$$q(1) = \cancel{f(x)} + z_A - \cancel{f(x)} = z_A \quad \checkmark$$

$$q(2) = 2z_A - f(x) = \cancel{2z_A} - \cancel{2}\left(\cancel{z_A} - z_B^{(1)}\right) = 2z_B^{(1)} \quad \checkmark$$

$$q(3) = 3z_A - 2f(x) = \cancel{3z_A} - \cancel{3}\left(\cancel{z_A} - z_B^{(2)}\right) = 3z_B^{(2)} \quad \checkmark$$

$$q(0) = f(x)$$

Claim: z_A and $s_i \leftarrow (i+1)z_B^{(i)}$ are evaluations of the polynomial $q(\ell) = f(x) + \ell(z_A - f(x))$ on $\{1, 2, \ldots, n\}$

$$q(1) = \cancel{f(x)} + z_A - \cancel{f(x)} = z_A \checkmark$$

$$q(2) = 2z_A - f(x) = \cancel{2z_A} - \cancel{2}\left(\cancel{z_A} - z_B^{(1)}\right) = 2z_B^{(1)} \checkmark$$

$$q(3) = 3z_A - 2f(x) = \cancel{3z_A} - \cancel{3}\left(\cancel{z_A} - z_B^{(2)}\right) = 3z_B^{(2)} \checkmark$$

$$q(0) = f(x)$$

So all parties have Degree-1 Shamir Secret Shares of $f(x)$!

Compressing Degree-1 Shamir Shares

From HSS with Simulatable Memory Shares

Correctness: Analysis on the previous slide shows that each party gets an evaluation of the polynomial $q(\ell)$ where $q(0) = f(x)$. Correctness then follows from HSS correctness by letting f be a target correlation function.

Security: When the HSS output shares are *pseudorandom*, the resulting sharing is indistinguishable from a random Degree-1 Shamir Sharing of $f(x)$.

Beyond Degree 1:
Threshold-Shared Correlations

Degree-t Shamir Shares of Constant-Degree Correlations

When $n - t = O(1)$, from LPN + MQ

Main idea: split each value into pieces over t-subsets (replicated sharing), which the parties can convert to degree-t Shamir shares locally.

Each cross term of $a \cdot b$ only involves the $O(1)$ parties that hold one of its factors, so an existing constant-party DPF[CKY'25] can distribute it.

See paper for details!

Building Pseudorandom Permutations

Constructing Pseudorandom Permutations

From Compressed Degree-1 Shamir Secret Shares

Insight: A permutation of $\{0, 1, 2\}$ can be generated from a degree-1 Shamir sharing of a bit over $\mathbb{F}_3$, where the bit is encoded as the coefficient of the sharing polynomial rather than as its constant term.

Constructing Pseudorandom Permutations

From Compressed Degree-1 Shamir Secret Shares

Insight: A permutation of $\{0, 1, 2\}$ can be generated from a degree-1 Shamir sharing of a bit over $\mathbb{F}_3$, where the bit is encoded as the coefficient of the sharing polynomial rather than as its constant term.

Define: $p(y) = z_A + \mu y$ where $z_A \in \mathbb{F}_3$ and $\mu \in \{0, 1\}$

Constructing Pseudorandom Permutations

From Compressed Degree-1 Shamir Secret Shares

Insight: A permutation of $\{0,1,2\}$ can be generated from a degree-1 Shamir sharing of a bit over $\mathbb{F}_3$, where the bit is encoded as the coefficient of the sharing polynomial rather than as its constant term.

Define: $p(y) = z_A + \mu y$ where $z_A \in \mathbb{F}_3$ and $\mu \in \{0,1\}$



$p(0) = z_A$ The secret is the slope μ (not p(0)), so Alice can safely hold p(0).



$p(-1) = z_A - \mu$



$p(1) = z_A + \mu$

$z_B^{(\alpha)} = z_A - \alpha\mu = p(-\alpha)$

μ is a PRF bit computed under HSS.

Constructing Pseudorandom Permutations

From Compressed Degree-1 Shamir Secret Shares

Insight: A permutation of $\{0, 1, 2\}$ can be generated from a degree-1 Shamir sharing of a bit over $\mathbb{F}_3$, where the bit is encoded as the coefficient of the sharing polynomial rather than as its constant term.

Define: $p(y) = z_A + \mu y$ where $z_A \in \mathbb{F}_3$ and $\mu \in \{0, 1\}$



$p(0) = z_A$



$p(-1) = z_A - \mu$ → $z_A - \mu + 2$



$p(1) = z_A + \mu$ → $z_A + \mu + 1$

Fixed public shifts ⇒ always distinct

Constructing Pseudorandom Permutations

From Compressed Degree-1 Shamir Secret Shares

Claim: the three shares are always pairwise distinct, i.e. a permutation of {0, 1, 2}.

$x \in \mathbb{F}_3 \quad \mu \in \{0, 1\}$

 x

 $x - \mu + 2$

 $x + \mu + 1$

Constructing Pseudorandom Permutations

From Compressed Degree-1 Shamir Secret Shares

Claim: the three shares are always pairwise distinct, i.e. a permutation of {0, 1, 2}.

$x \in \mathbb{F}_3 \quad \mu \in \{0, 1\}$



x



$x - \mu + 2$



$x + \mu + 1$

$\mu = 0 : \quad (x, x + 2, x + 1)$

x = 0	x = 1	x = 2
0	1	2
2	0	1
1	2	0

Constructing Pseudorandom Permutations

From Compressed Degree-1 Shamir Secret Shares

Claim: the three shares are always pairwise distinct, i.e. a permutation of {0, 1, 2}.

$x \in \mathbb{F}_3 \quad \mu \in \{0, 1\}$



x



$x - \mu + 2$



$x + \mu + 1$

$\mu = 0 : \quad (x, x+2, x+1)$

x = 0	x = 1	x = 2
0	1	2
2	0	1
1	2	0

$\mu = 1 : \quad (x, x+1, x+2)$

x = 0	x = 1	x = 2
0	1	2
1	2	0
2	0	1

Constructing Pseudorandom Permutations

From Compressed Degree-1 Shamir Secret Shares

Claim: the three shares are always pairwise distinct, i.e. a permutation of {0, 1, 2}.

$x \in \mathbb{F}_3 \quad \mu \in \{0, 1\}$



x



$x - \mu + 2$



$x + \mu + 1$

$\mu = 0 : \quad (x, x+2, x+1)$

x = 0	x = 1	x = 2
0	1	2
2	0	1
1	2	0

$\mu = 1 : \quad (x, x+1, x+2)$

x = 0	x = 1	x = 2
0	1	2
1	2	0
2	0	1

Every column is a permutation of {0, 1, 2} ✓

Full results in the paper:

	Assumptions	Correlation	# Parties
Prior Work [ASY'22]	iO + MKFHE	Any reverse-samplable correlation	n
Prior Work [DHRW'16] + [CDI'05]	LWE	Linear-output correlations (incl. Shamir)	n
Prior Work [ABGIKM'24] + [BCGIKS'19]	LPN	Degree-1 Shamir Shares of const. degree poly	n
This work (PCF)	HSS + PRF in NC1	Degree-1 Shamir Shares of any NC1 correlation	n
This work (PCG)	LPN + MQ	Degree-t Shamir Shares of const. degree poly ($n - t = O(1)$)	n
This work (PCG)	LPN + MQ	Linear threshold shares of const. degree poly ($t = O(1)$)	n
This work (PCF)	HSS + PRF in NC1	Permutations	3

Thank you

Email: sacha@tinfoil.sh
ePrint: ia.cr/2026/255

On Compressing Linearly Shared Correlations

Geoffroy Couteau[1], Alexander Koch[2], Nikolas Melissaris[1], Peter Scholl[3], Sacha Servan-Schreiber[4], and Xiaxi Ye[5]

[1] CNRS, IRIF & Université Paris Cité
[2] Secorvo Security Consulting GmbH
[3] Aarhus University
[4] Tinfoil
[5] Tsinghua University

References

[ABGIKM'24]: Agarwal, Boyle, Gilboa, Ishai, Kelkar, and Ma. Compressing unit-vector correlations via sparse pseudorandom generators. CRYPTO 2024.

[ASY'22]: Abram, Scholl, and Yakoubov. Distributed (correlation) samplers: How to remove a trusted dealer in one round. EUROCRYPT 2022.

[BCGI'18]: Boyle, Couteau, Gilboa, and Ishai. Compressing vector OLE. ACM CCS 2018.

[BCGIKS'19]: Boyle, Couteau, Gilboa, Ishai, Kohl, and Scholl. Efficient pseudorandom correlation generators: Silent OT extension and more. CRYPTO 2019.

[BCGIO'17]: Boyle, Couteau, Gilboa, Ishai, and Orrù. Homomorphic secret sharing: Optimizations and applications. ACM CCS 2017.

[BGI'16]: Boyle, Gilboa, and Ishai. Breaking the circuit size barrier for secure computation under DDH. CRYPTO 2016.

[CDI'05]: Cramer, Damgård, and Ishai. Share conversion, pseudorandom secret-sharing and applications to secure computation. TCC 2005.

[CKY'25]: Couteau, Kumar, and Ye. Multiparty homomorphic secret sharing and more from LPN and MQ. TCC 2025.

[CMPR'23]: Couteau, Meyer, Passelègue, and Riahinia. Constrained pseudorandom functions from homomorphic secret sharing. EUROCRYPT 2023.

[DHRW'16]: Dodis, Halevi, Rothblum, and Wichs. Spooky encryption and its applications. CRYPTO 2016.

[GJ'04]: Golle and Juels. Dining cryptographers revisited. EUROCRYPT 2004.