PCG Part 3: Silent VOLE and OT Protocols from LPN

Peter Scholl

26 January 2022, Bar-Ilan University Winter School

Based on joint work with:

Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, Peter Rindal

AARHUS UNIVERSITY

This week's talks

VOLE 1: introduction, basic protocols & applications

VOLE 2: application to efficient zero knowledge

PCG 1-2

PCG 3: PCGs from LPN: the gory details

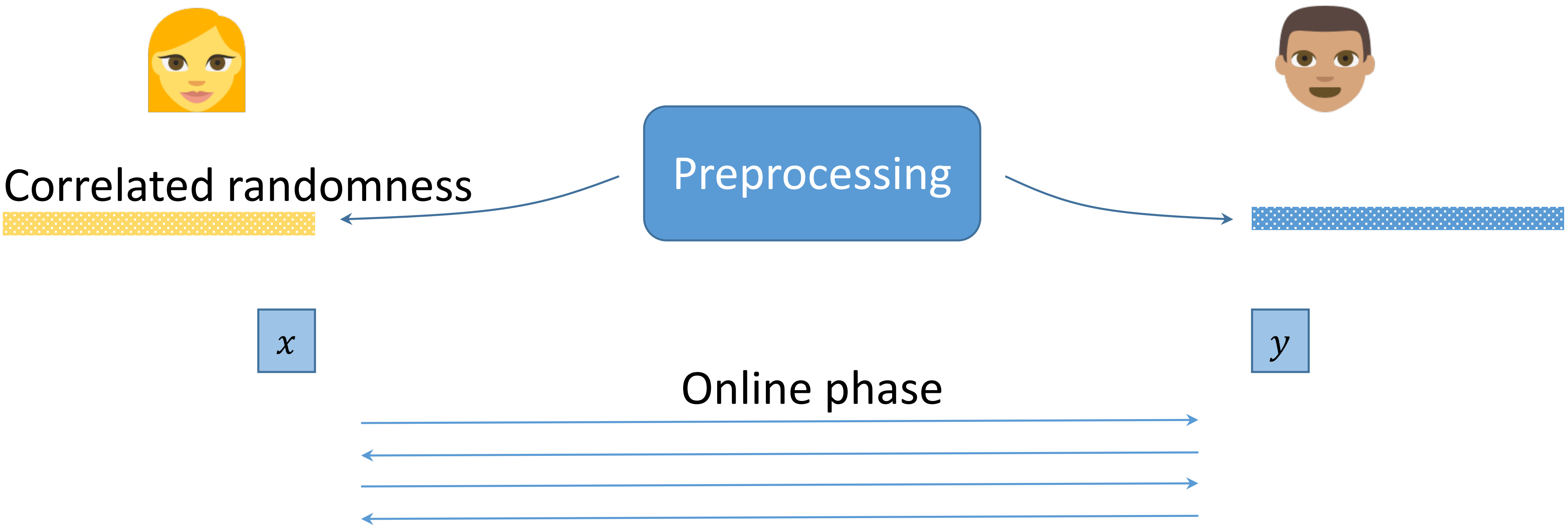
PCG 4: PCFs from number-theoretic assumptions

Outline

- Recap of OT extension (non-silent!)
- Blueprint for silent OT
  - ➢Instantiate with LPN
- PCG setup protocol for silent OT/VOLE
  - ➢Two-rounds, active security
- Conclusion & open problems

Secure Computation with Preprocessing

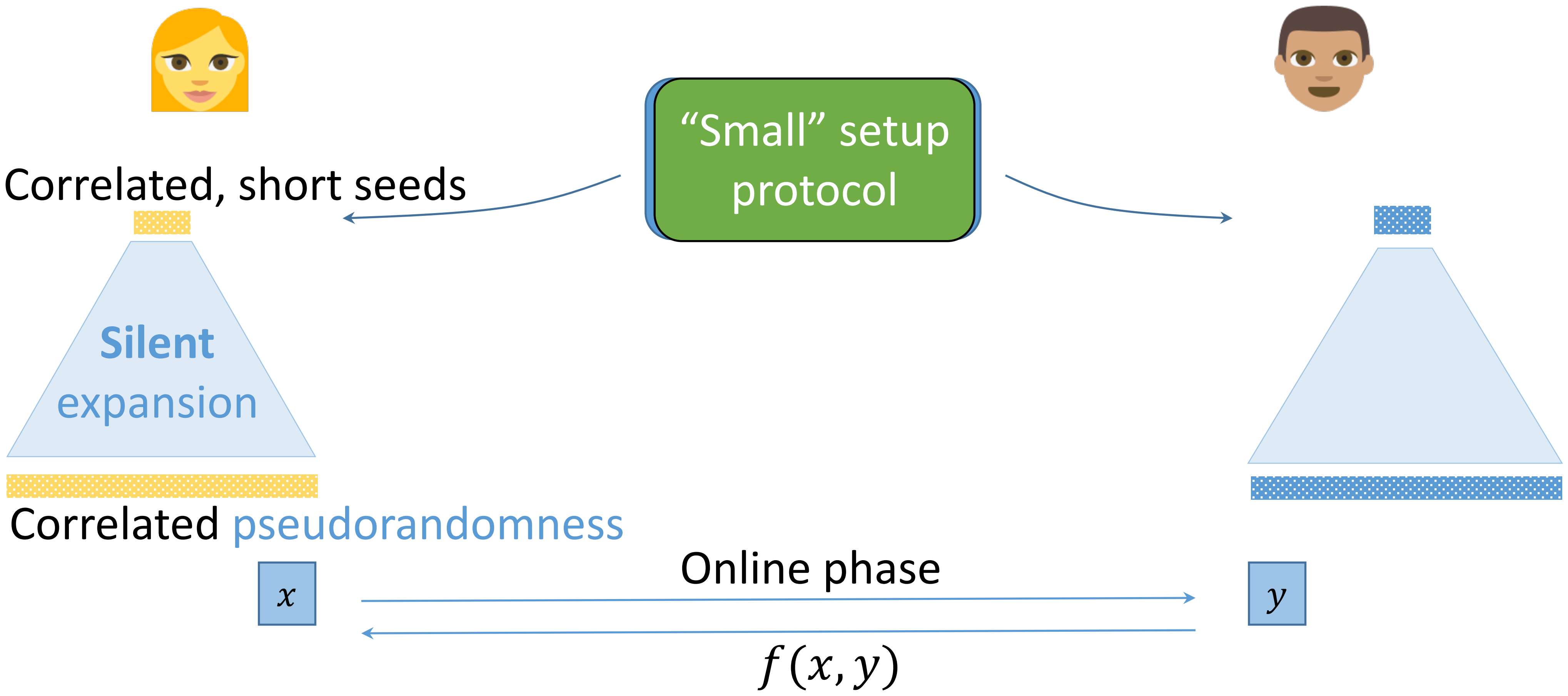
[Beaver '91]



$$f(x, y)$$

Secure Computation with Silent Preprocessing

[BCGI 18, BCGIKS 19]

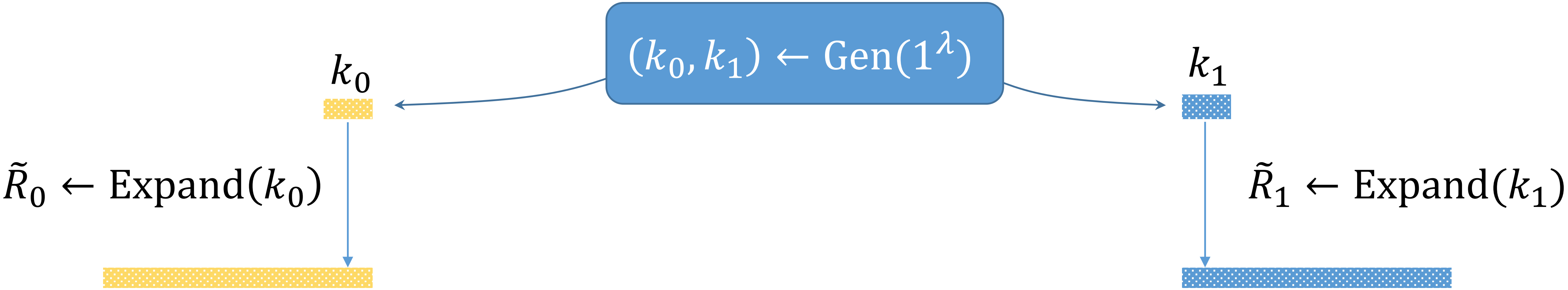


Pseudorandom Correlation Generators

[BCGI 18, BCGIKS 19]

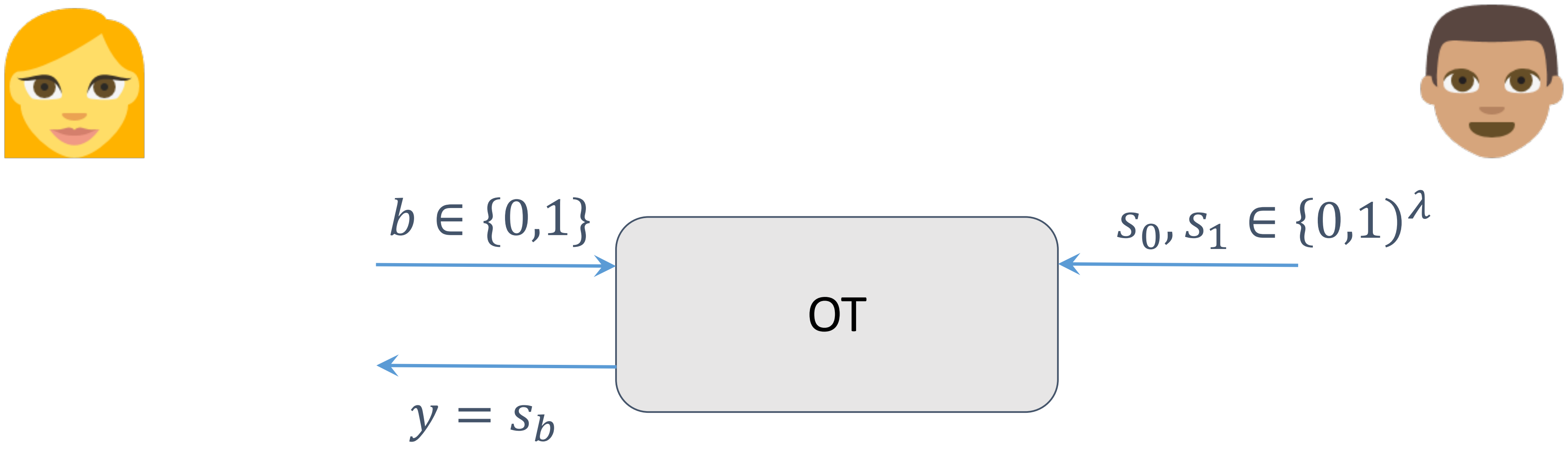
- Target correlation: (R_0, R_1)

- Algorithms Gen, Expand:



Security: $\left(k_0, \tilde{R}_1\right) \approx (k_0, [R_1 | R_0 = \mathrm{Expand}(k_0)])$

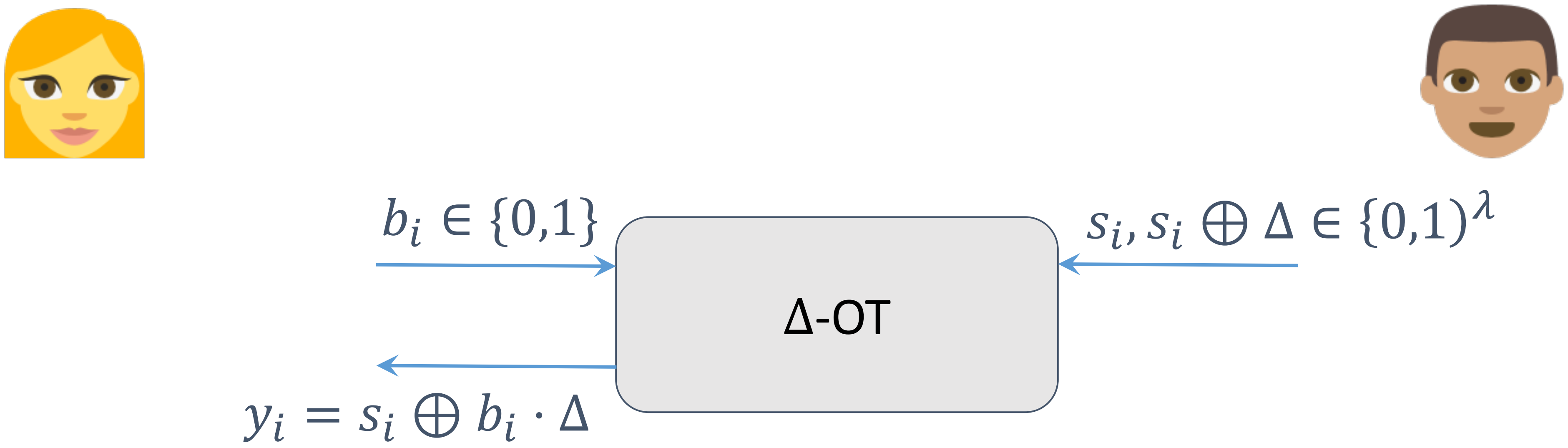
Oblivious Transfer



OT requires public-key cryptography

OT extension: costly PK operations only in setup phase

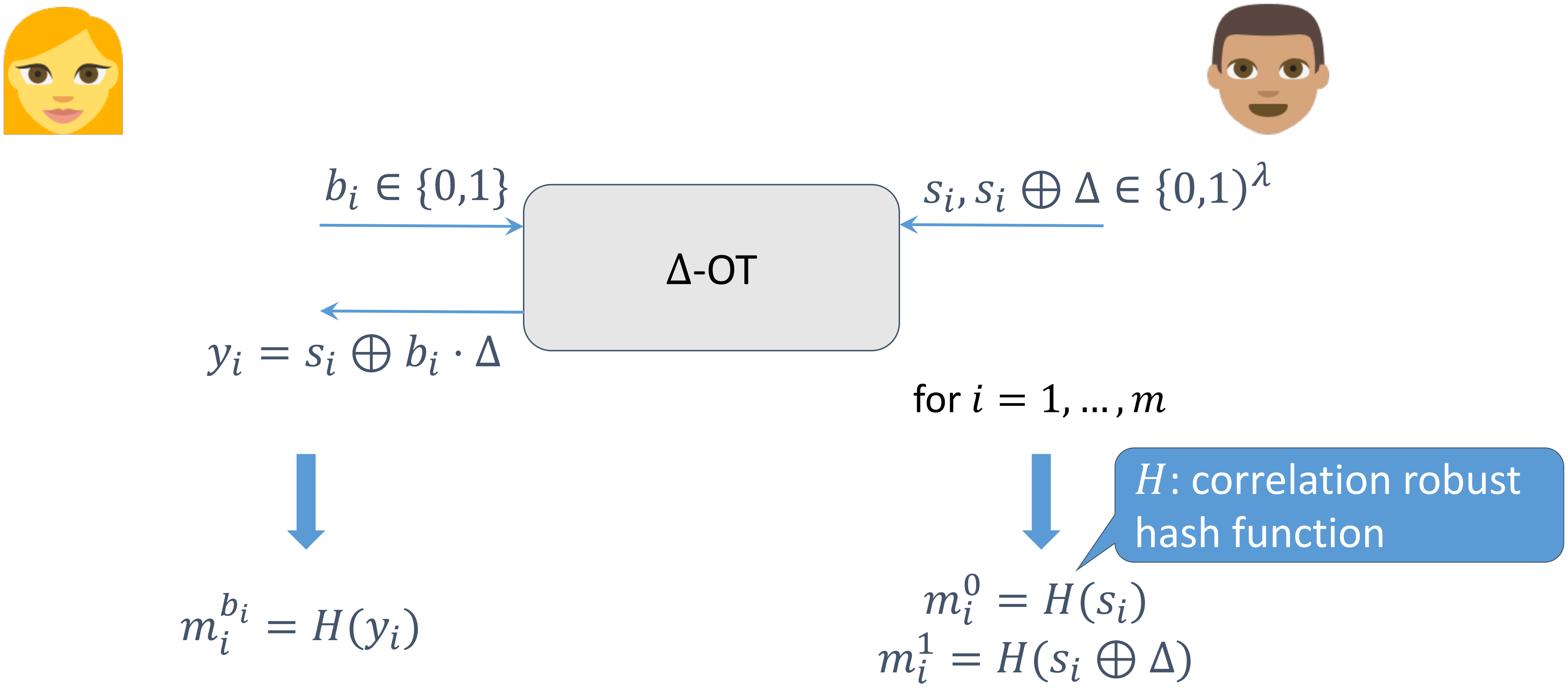
(Batch of) **Correlated** Oblivious Transfers



for $i = 1, \ldots, m$

(Equivalent to subfield VOLE, or information-theoretic MACs over $\mathbb{F}_2$)

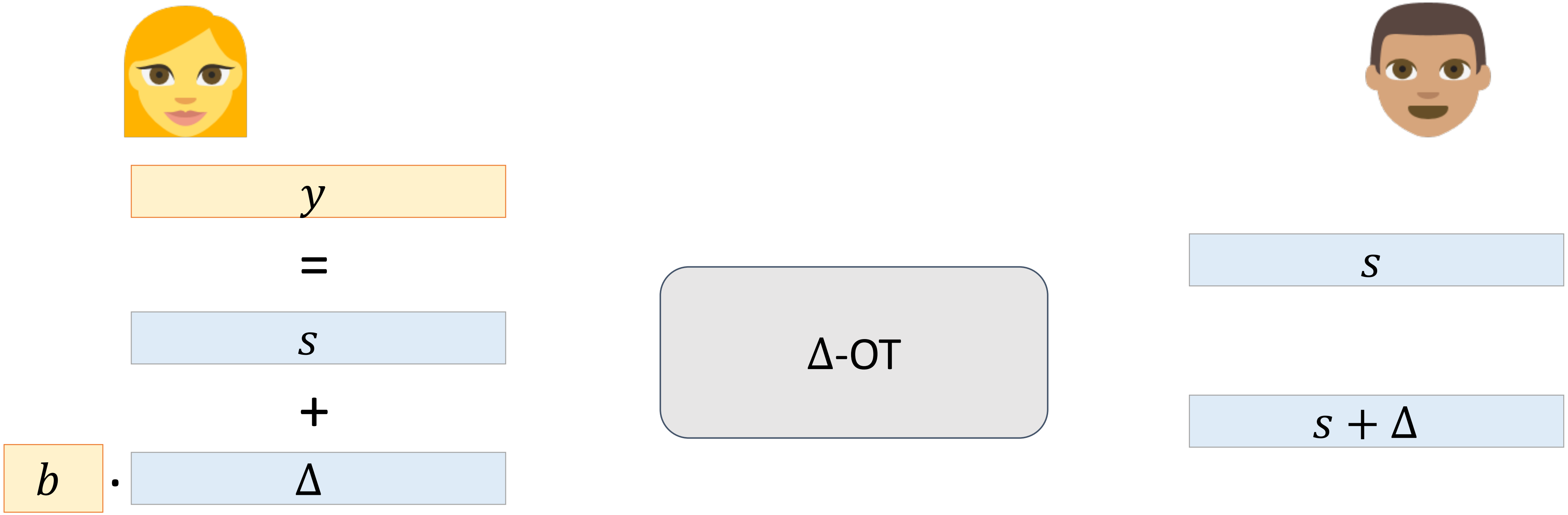
From correlated OT to random OT [IKNP 03]



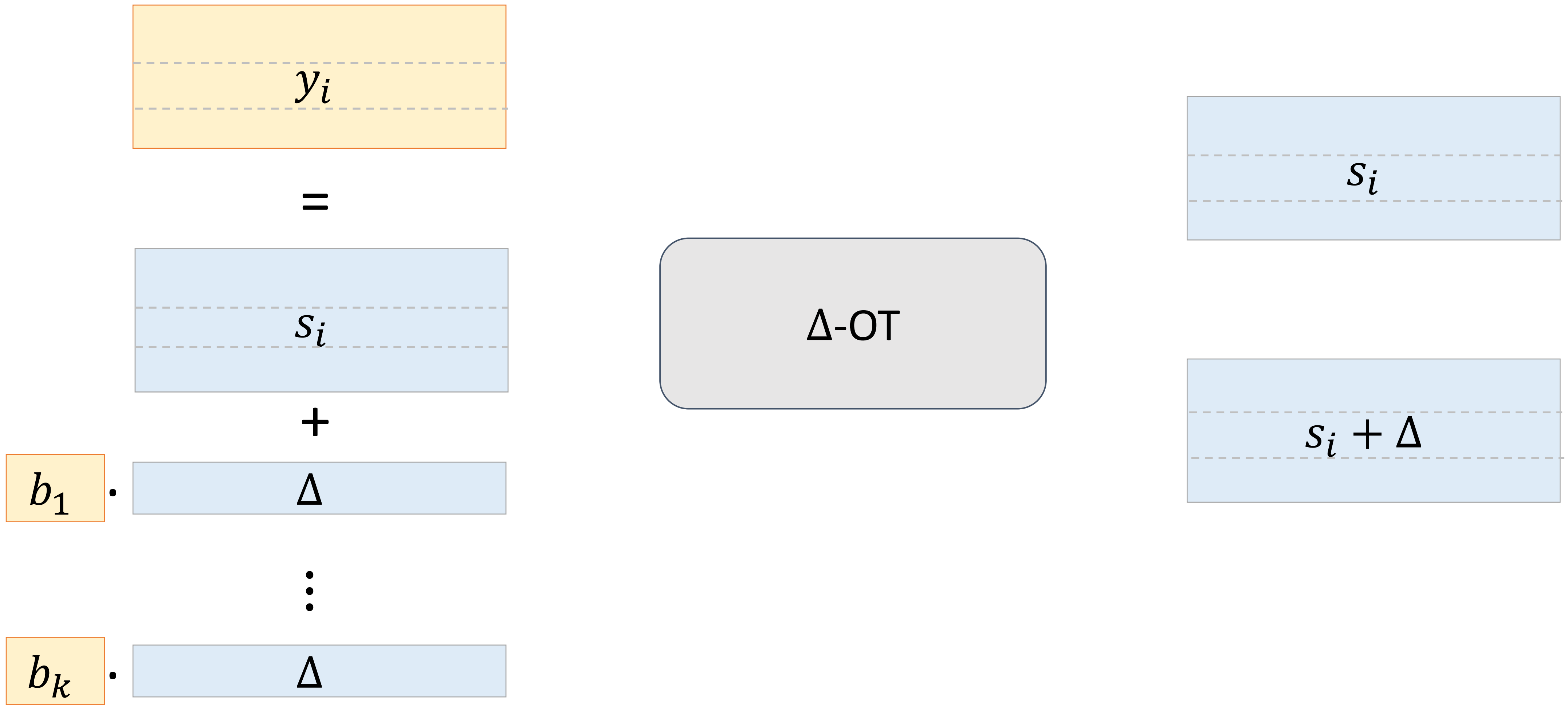
IKNP OT Extension: Correlate, Transpose & Hash

[IKNP 03]

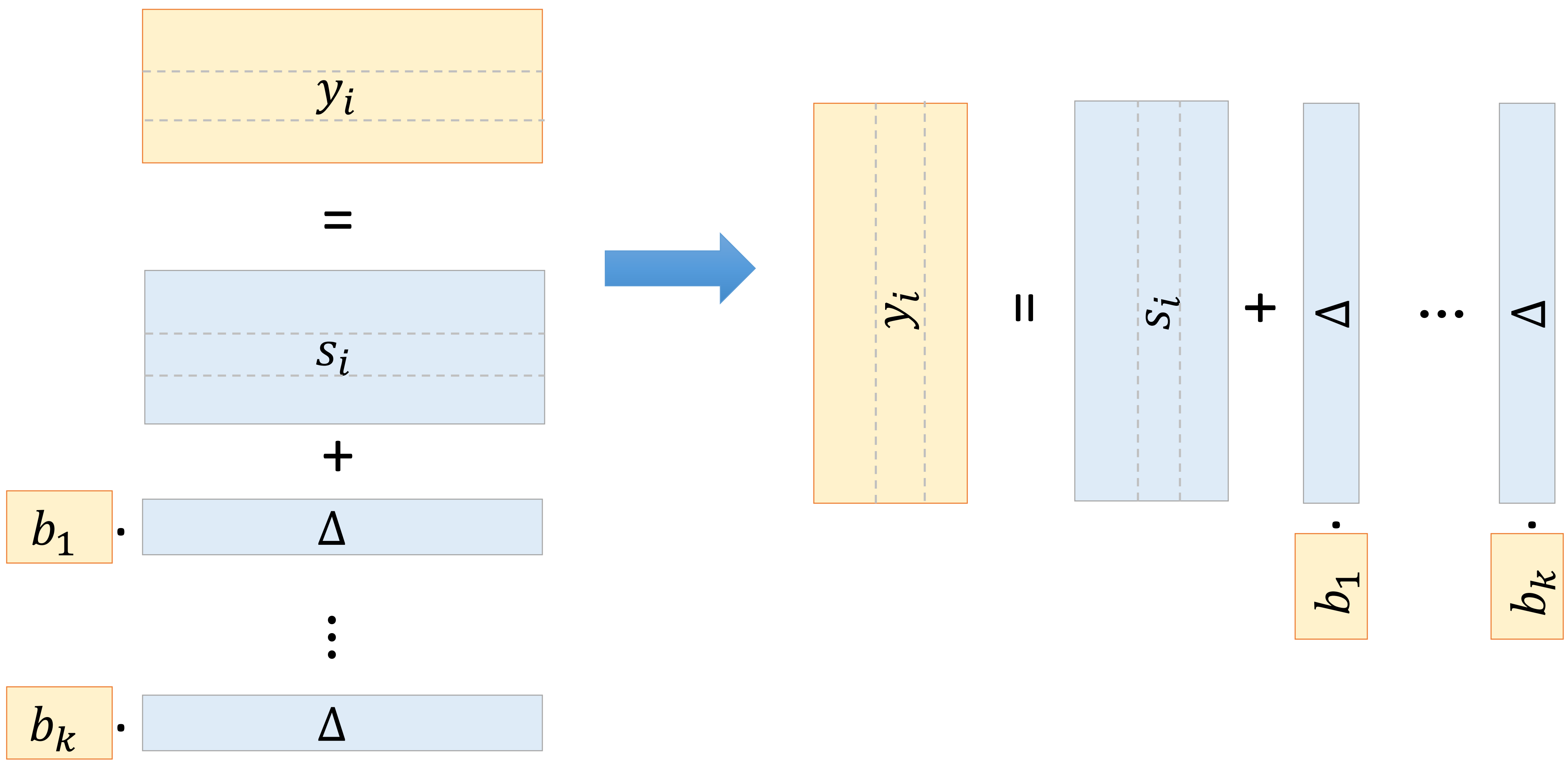
IKNP: correlate



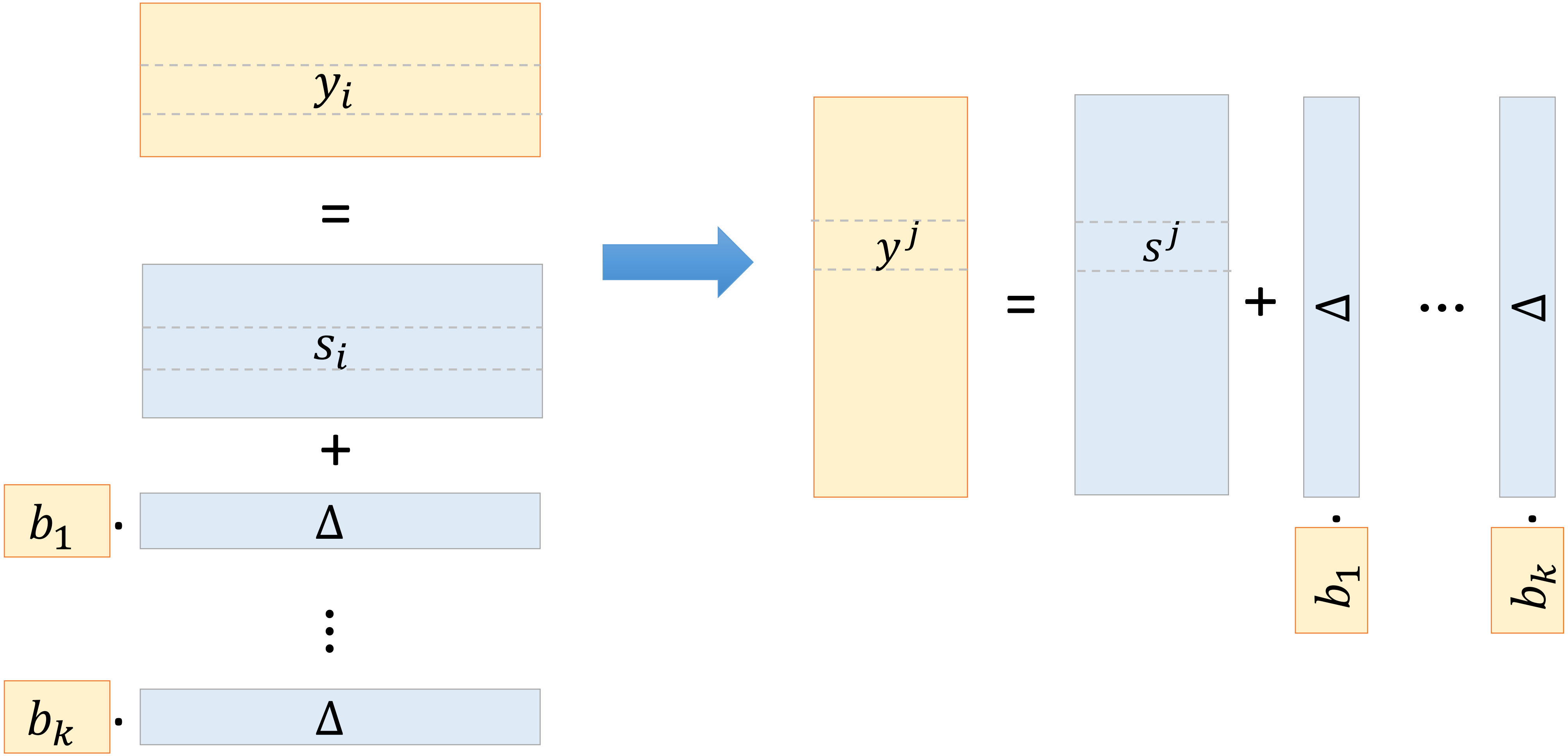
IKNP: correlate



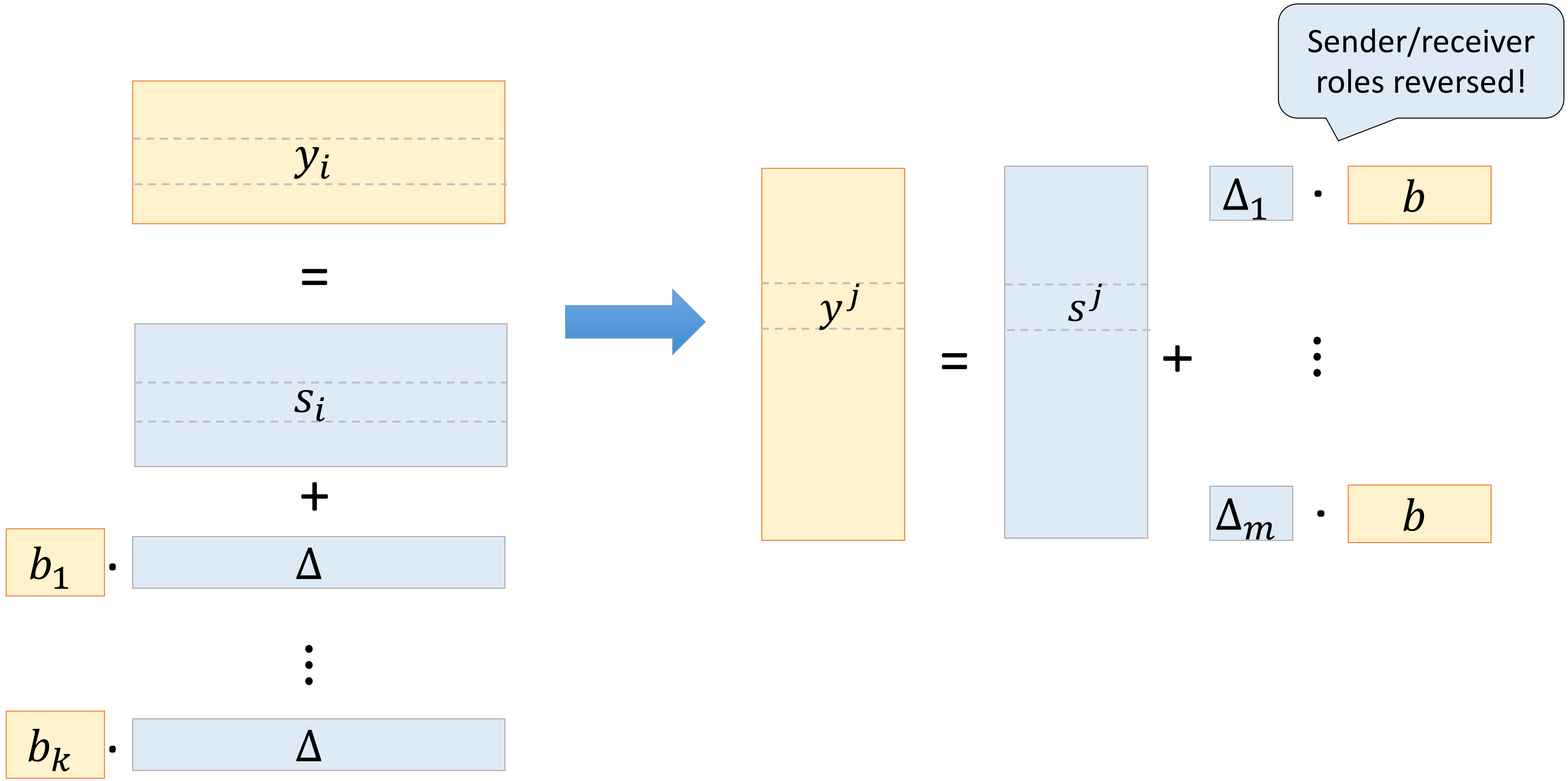
IKNP: correlate, transpose



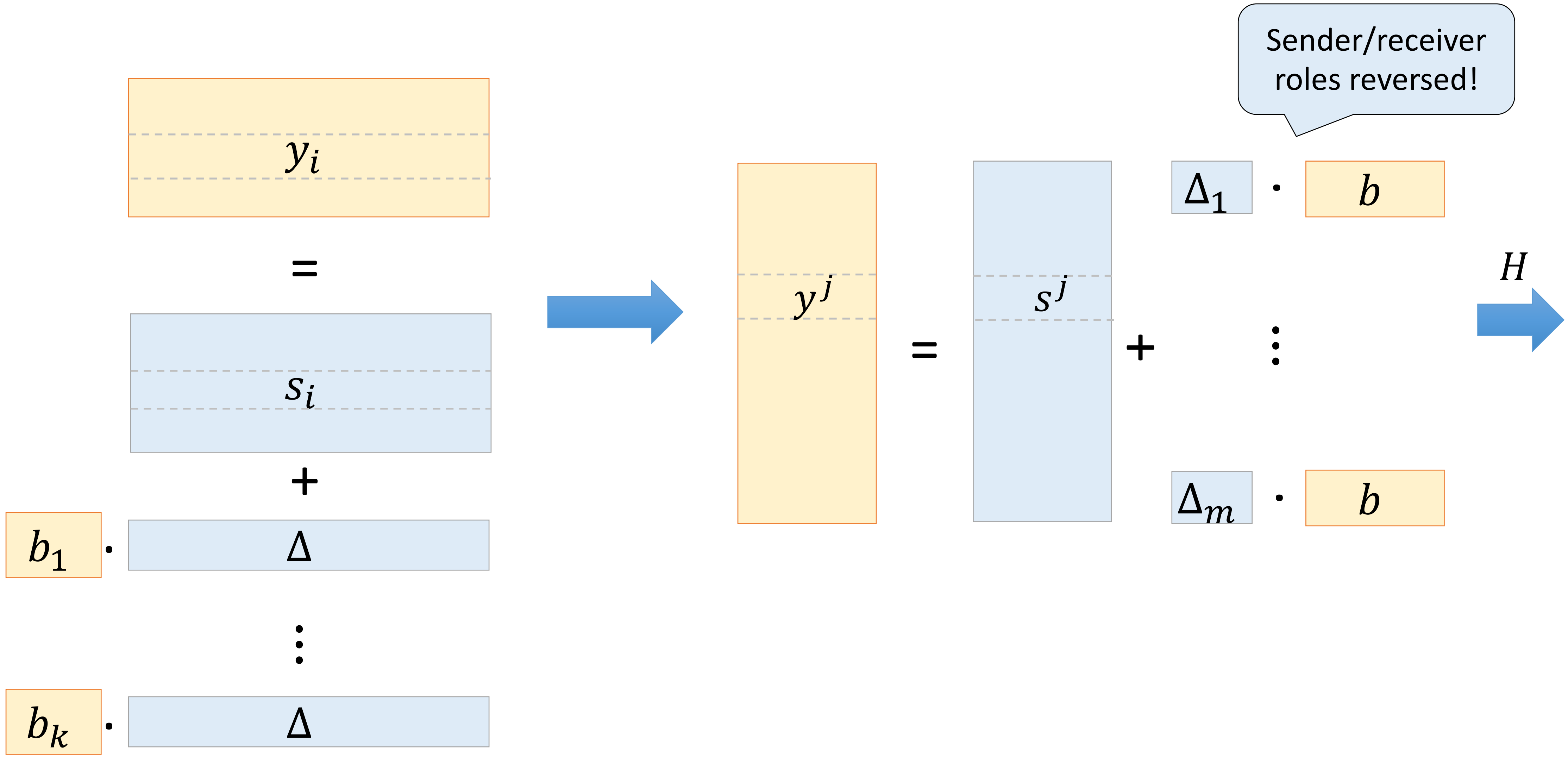
IKNP: correlate, transpose



IKNP: correlate, transpose



IKNP: correlate, transpose and hash



Bottleneck:
- Long correlated OTs
- Cost: 128 bits per OT

IKNP OT Extension: Correlate, Transpose & Hash

[IKNP 03]

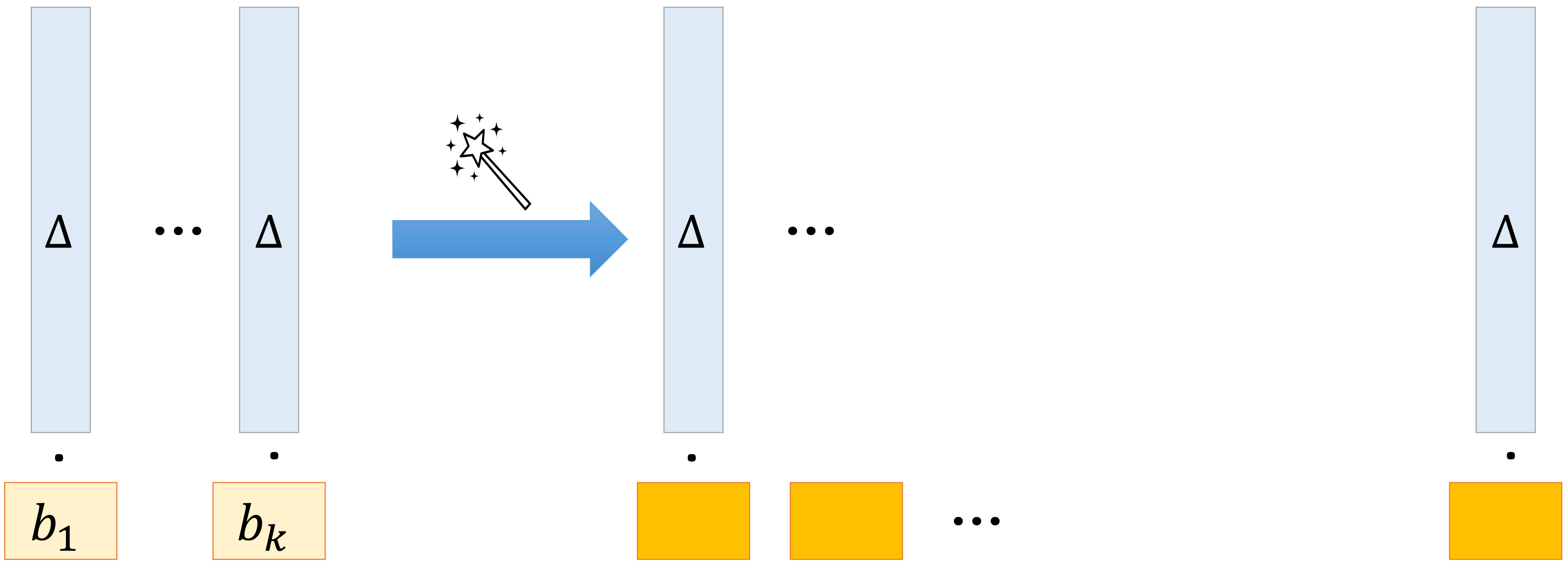
Much “smaller” correlation

- Roles stay the same

Silent OT Extension: Correlate, Expand & Hash

[BCGIK**S** 19]

Silent OT Extension: Correlate, Expand & Hash



Silent expansion via homomorphic PRGs?

- Suppose we have a PRG where

$$G(s + t) \;\ldots\; s) + G(t)$$

- Receiver can expand $\vec{b} \rightarrow G(\vec{b})$
  - ➢ Parties expand s_i, y_i the same way
  - ➢ Preserves OT relation

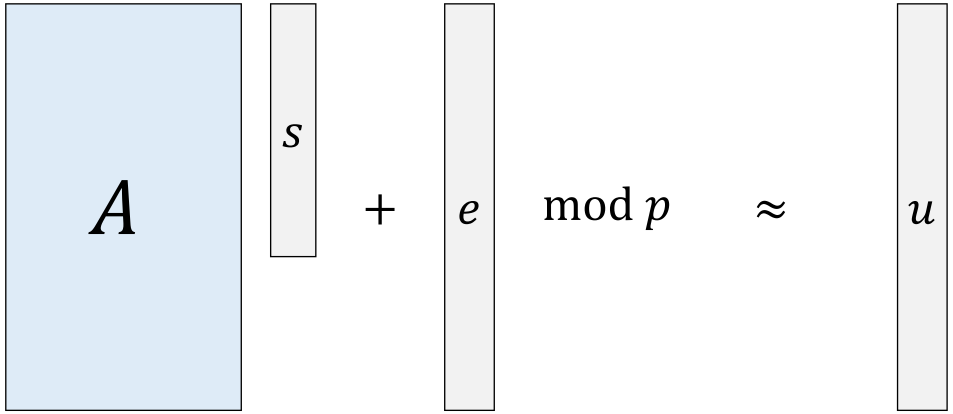
- G is totally insecure!

- Lattice-based PRGs are almost-homomorphic
  - ➢ Good enough for weaker form of silent OT [**S** 18]

Silent expansion via learning parity with noise

[BCGI 18]

Given $A \in \mathbb{Z}_p^{m \times n}$:



LWE

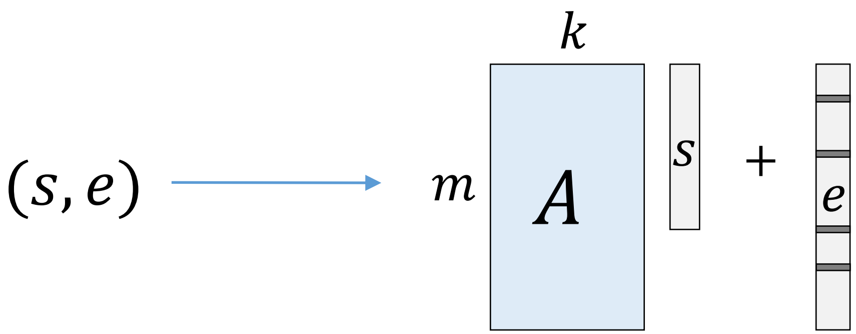
- $p > 2$
- $s \leftarrow Z_p^n$
- $||e||_\infty$ is small

LPN

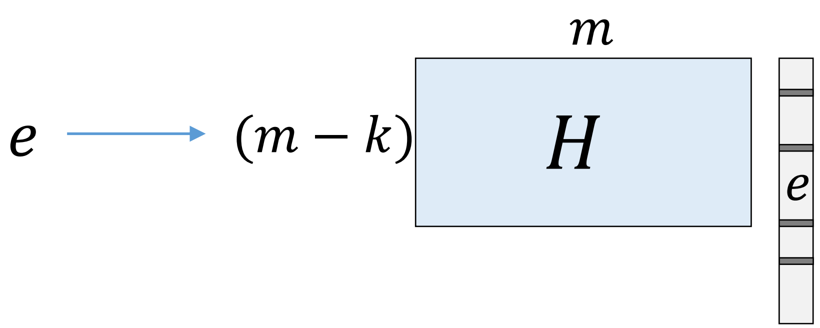
- $p \geq 2$ (arithmetic generalization)
- $s \leftarrow \mathbb{Z}_p^n$
- $HW(e)$ is small

"Linear-ish" PRGs from LPN

"Primal" construction



"Dual" construction



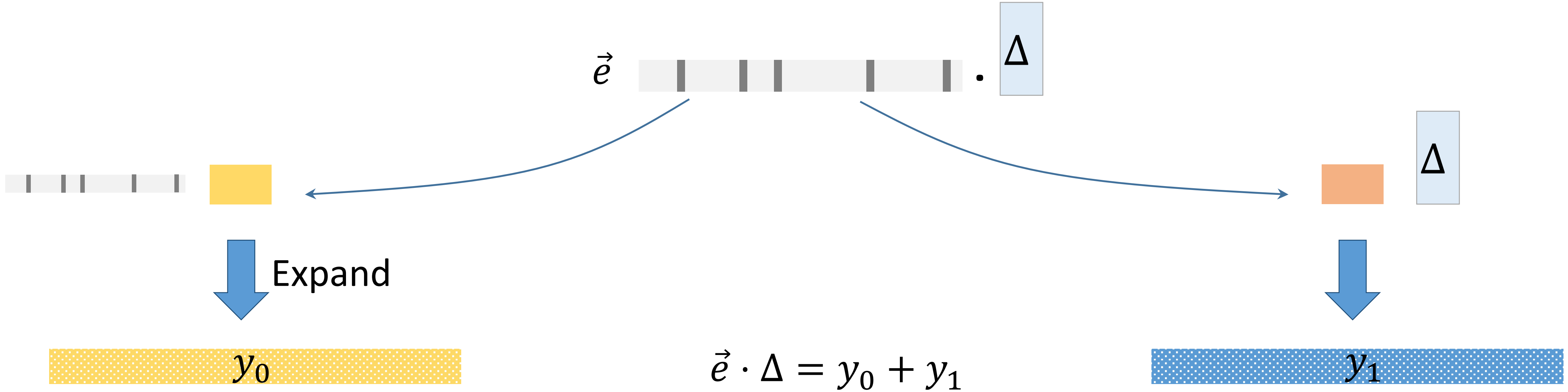
Evaluation is linear in (s, e)!

Limited to quadratic stretch

Arbitrary poly stretch
(increase m, fix $HW(e)$)
$\Rightarrow$ best attack: $\exp(HW(e))$

Secret-sharing sparse vectors: core of PCGs from LPN

Goal: compress secret-shares of sparse vector



$$\vec{e} \cdot \Delta = y_0 + y_1$$

Main tool: puncturable PRF

FSS is overkill!

- PRF $F : \{0,1\}^\lambda \times \{1, \ldots, N\} \to \{0,1\}^\lambda$

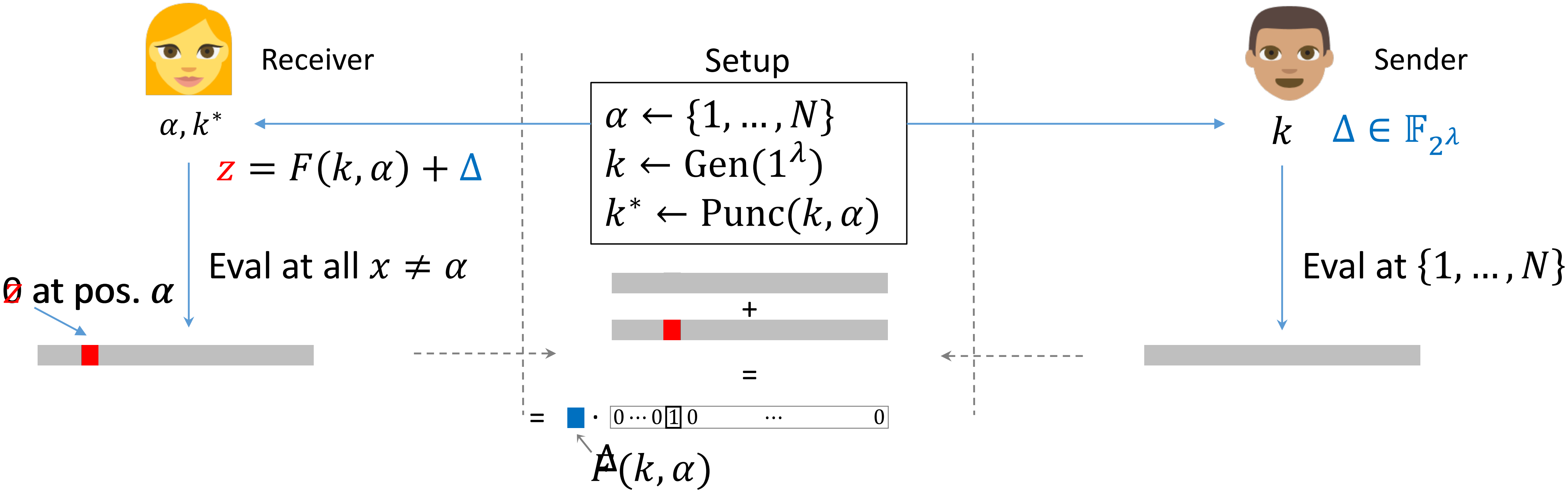
- $k \leftarrow \mathrm{Gen}(1^\lambda)$
  - Master key: allows evaluating $F(k, x)$ for all x
- $k^* \leftarrow \mathrm{Punc}(k, \alpha)$
  - Punctured key: can evaluate at all points except for $x = \alpha$

- Security: $F(k, \alpha)$ is pseudorandom, given k^*

Simple tree-based construction from a PRG: $|k| = \lambda, \quad |k^*| = \lambda \cdot \log N$

[BW13], [BGI 13], [KPTZ 13]

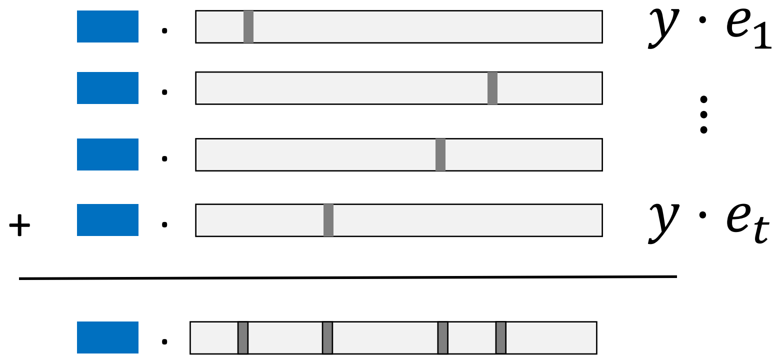
Sharing sparse vectors from puncturable PRF



- Shares compressed from $\lambda \cdot N$ to $\approx \lambda \cdot \log N$ bits
- Can tweak to multiply by arbitrary $\Delta \in \mathbb{F}_{2^\lambda}$

From weight-1 vectors to weight-t vectors

Approach 1: addition

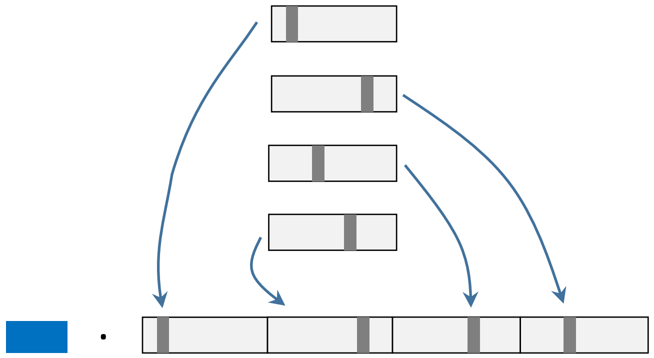


Weight e.g. $t = 4$

Expansion cost: $O(t \cdot N)$ (naïve)

$O(N)$ (cuckoo hashing [SGRR 19])

Approach 2: concatenation



$$O\left(t \cdot \frac{N}{t}\right) = O(N)$$

Note: regular error pattern

The missing pieces: plugging in LPN

- Use PPRF to share $\vec{e} \cdot \Delta$
- Primal: also share $\vec{s} \cdot \Delta$ via OT

- How to instantiate LPN matrix?

Matrix	Type	Complexity	Security
Sparse	Primal	$O(m)$	Back to [Ale 03]

The missing piece: plugging in LPN

- Use PPRF to share $\vec{e} \cdot \Delta$
- Primal: also share $\vec{s} \cdot \Delta$ via OT

- How to instantiate LPN matrix?

Matrix	Type	Complexity	Security
Sparse	Primal	$O(m)$	Back to [Ale 03]
Quasi-cyclic	Dual	$\tilde{O}(m)$	Same as NIST PQC

The missing piece: plugging in LPN

- Use PPRF to share $\vec{e} \cdot \Delta$
- Primal: also share $\vec{s} \cdot \Delta$ via OT

- How to instantiate LPN matrix?

Matrix	Type	Complexity	Security
Sparse	Primal	$O(m)$	Back to [Ale 03]
Quasi-cyclic	Dual	$\tilde{O}(m)$	Same as NIST PQC
Structured LDPC	Dual	$O(m)$	[CRR 21]

The missing piece: plugging in LPN

- Use PPRF to share $\vec{e} \cdot \Delta$
- Primal: also share $\vec{s} \cdot \Delta$ via OT

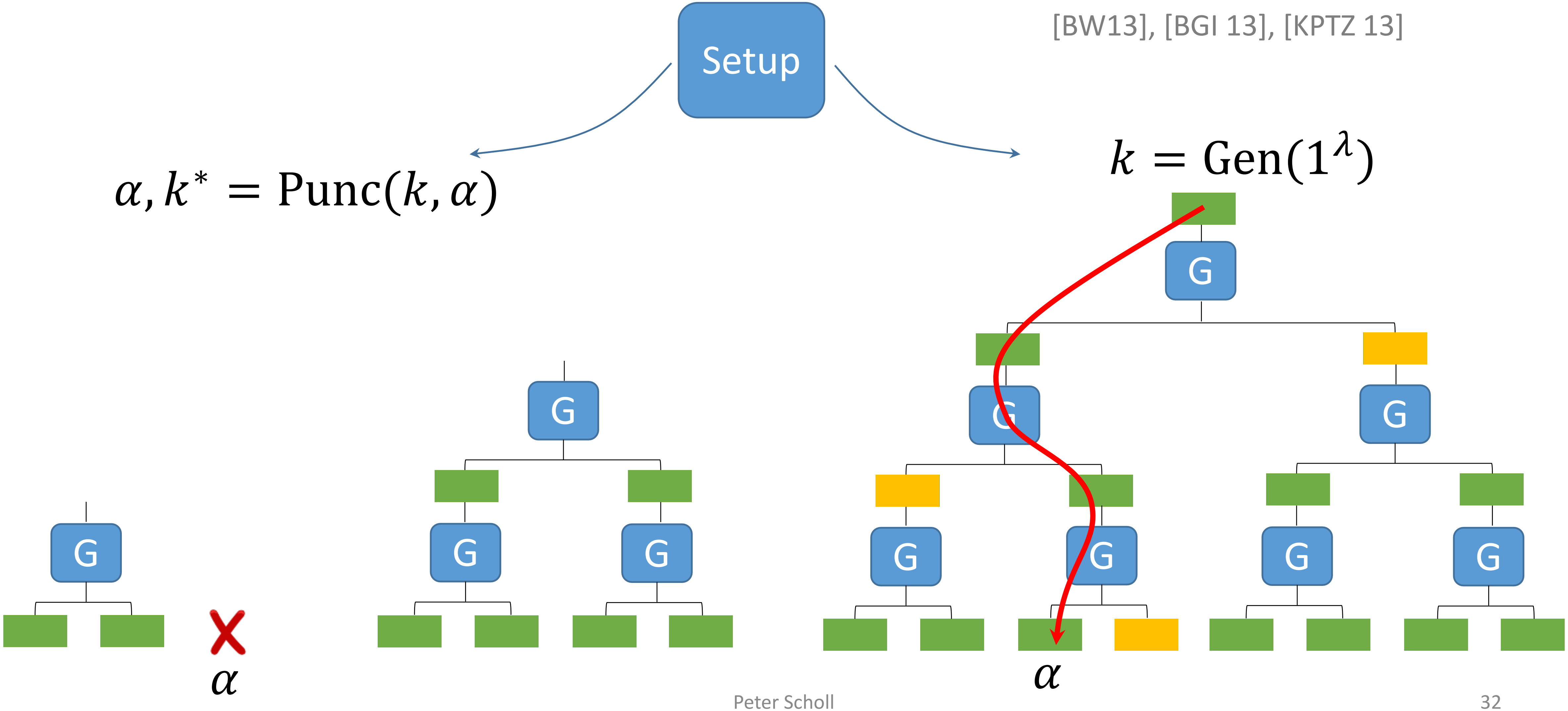
- How to instantiate LPN matrix?

Matrix	Type	Complexity	Security
Sparse	Primal	$O(m)$	Back to [Ale 03]
Quasi-cyclic	Dual	$\tilde{O}(m)$	Same as NIST PQC
Structured LDPC	Dual	$O(m)$	[CRR 21]
Cyclotomic ring-LPN (only for OLE)	Primal/dual	$\tilde{O}(m)$	[BCGIKS 20]

PCG setup protocol: some details

Setup protocol: inside the puncturable PRF

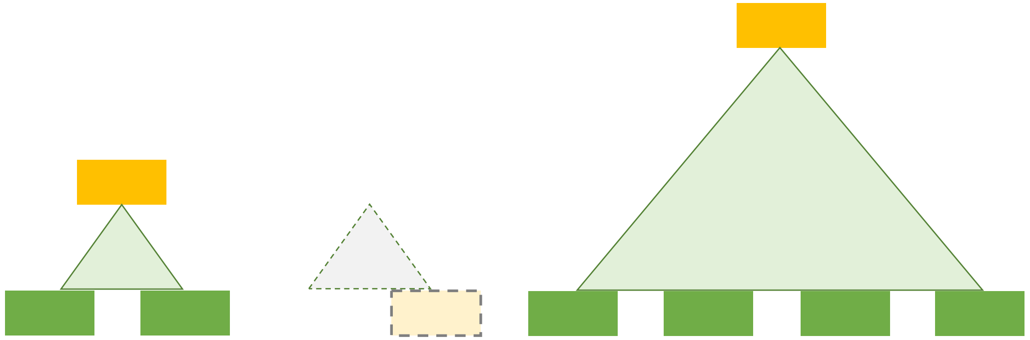
[BW13], [BGI 13], [KPTZ 13]

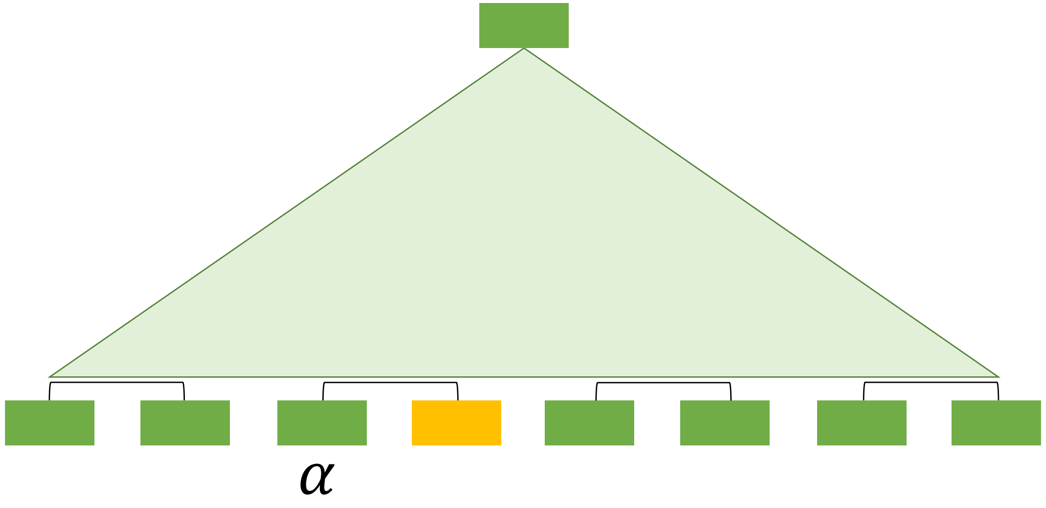


Setup protocol: inside the puncturable PRF

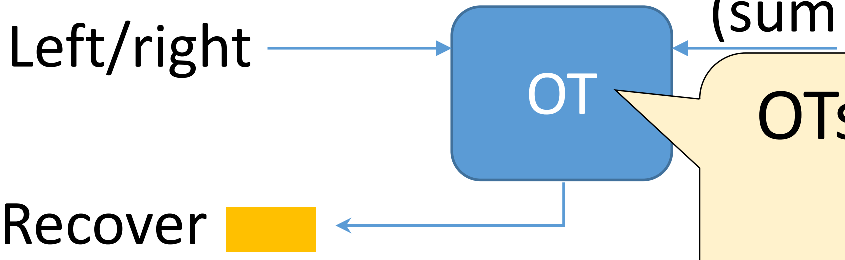
Based on [Doerner-shelat ‘17]

Suppose Receiver has [yellow box] for first 2 levels:





Use OT to transfer next [yellow box] :



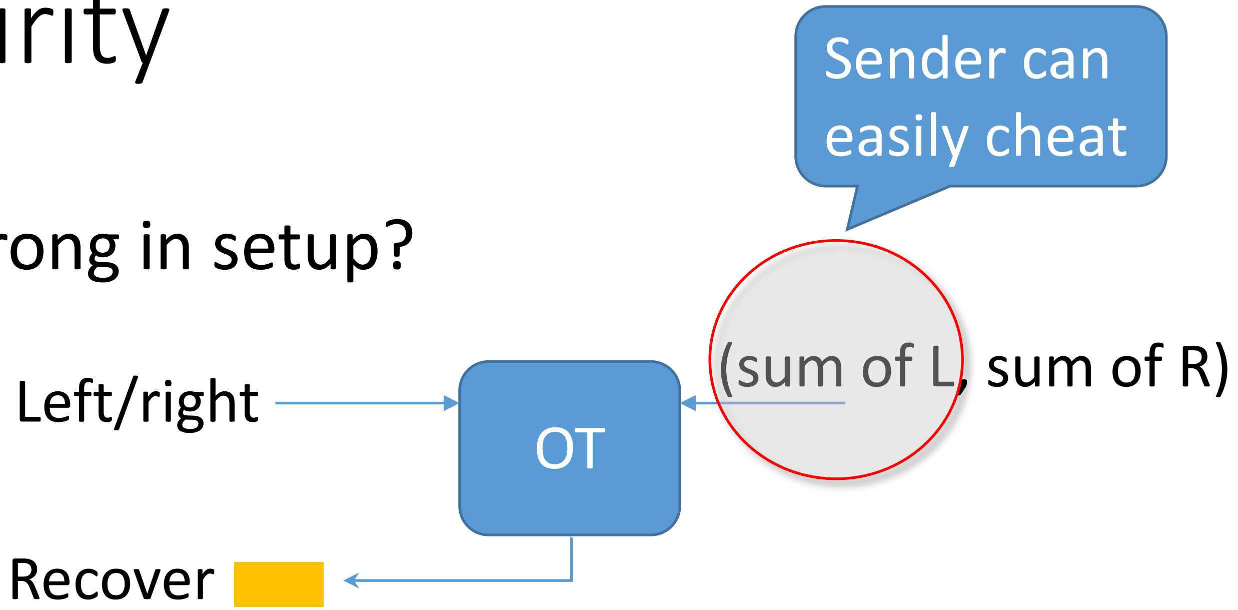
OTs for all levels can be done in parallel!
(Unlike [Ds 17] for DPF)

Setup Protocol for Silent OT/VOLE

- 2-round punctured PRF setup from any 2-round OT
  - $\log N$ parallel OTs
- 2-round Silent OT setup from any 2-round OT
  - Total cost: $\approx t \log N$ "seed" OTs for LPN noise weight t
  - (VOLE: also need seed VOLE)
- Two-round OT extension on chosen inputs
  - Can convert from random → chosen in parallel with setup
  - First concretely efficient two-round OT extension (previously only [Beaver '95])

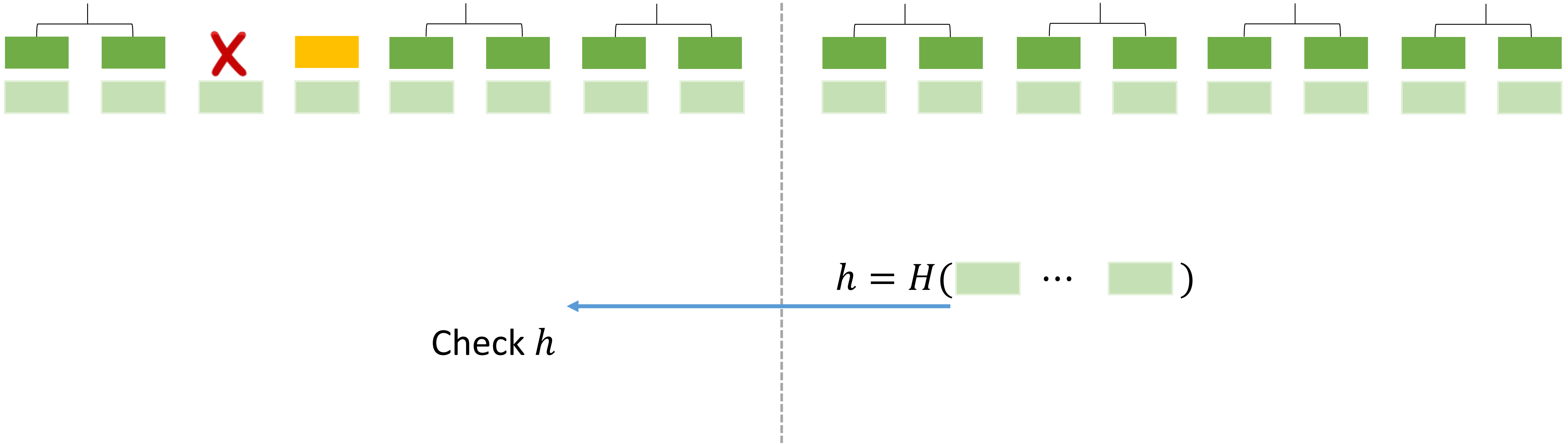
Active security

- What can go wrong in setup?



- Solution: consistency checks
  - ➢Still allows selective failure attacks – sender can guess 1 bit of LPN error
  - ➢Assume problem is hard with 1-bit leakage

Consistency check: hash the PPRF tree [BCGIKRS 19]



Collision-resistance ⇒ tree is consistent

Ensuring consistency among the trees

- What if sender uses different Δ's?
  - ➢Hash check doesn't catch this…
- Solution: another check!
  - ➢Random linear combination (like MAC check)
- Ferret/Wolverine [YWLZW 20, WYKW 21]:
  - ➢Linear combination instead of hash check
  - ➢Simpler, also ensures consistent Δ's

Performance for n=10 million random OTs (LAN)

128-bit security

Protocol	One-time setup (kB)	Comms	Time (ms)	Primal/dual
IKNP	-	160 MB	$\sim$400	-
[BCGIKRS 19]	-	**122 kB**	$\sim$5000	Dual (quasi-cyclic)
Ferret [WYKY 20]	1130 kB	550 kB	$\sim$500	Primal
Silver [CRR 21]	-	**122 kB**	$\sim$**300**	Dual (structured LDPC)

Conclusion

- Silent OT and VOLE:
  - Linear structure of LPN
  - Sharing sparse vectors via PPRF
- Two-round setup protocols
  - Actively secure
  - Give two-round OT extension
- Open problems:
  - More silent-friendly applications
  - Optimize multi-point PPRF: $\lambda \log N \rightarrow \lambda + \log N$?
  - Setup: can we do 1-round?
  - Security of LPN variants
    - Especially structured LDPC, VD-LPN, ring-LPN…

Thank you!

Efficient Pseudorandom Correlation Generators: Silent OT Extension and More

Boyle, Couteau, Gilboa, Ishai, Kohl, Scholl

https://ia.cr/2019/129

Two-Round OT Extension and Silent Non-Interactive Secure Computation

BCGIKS + Rindal

https://ia.cr/2019/1159