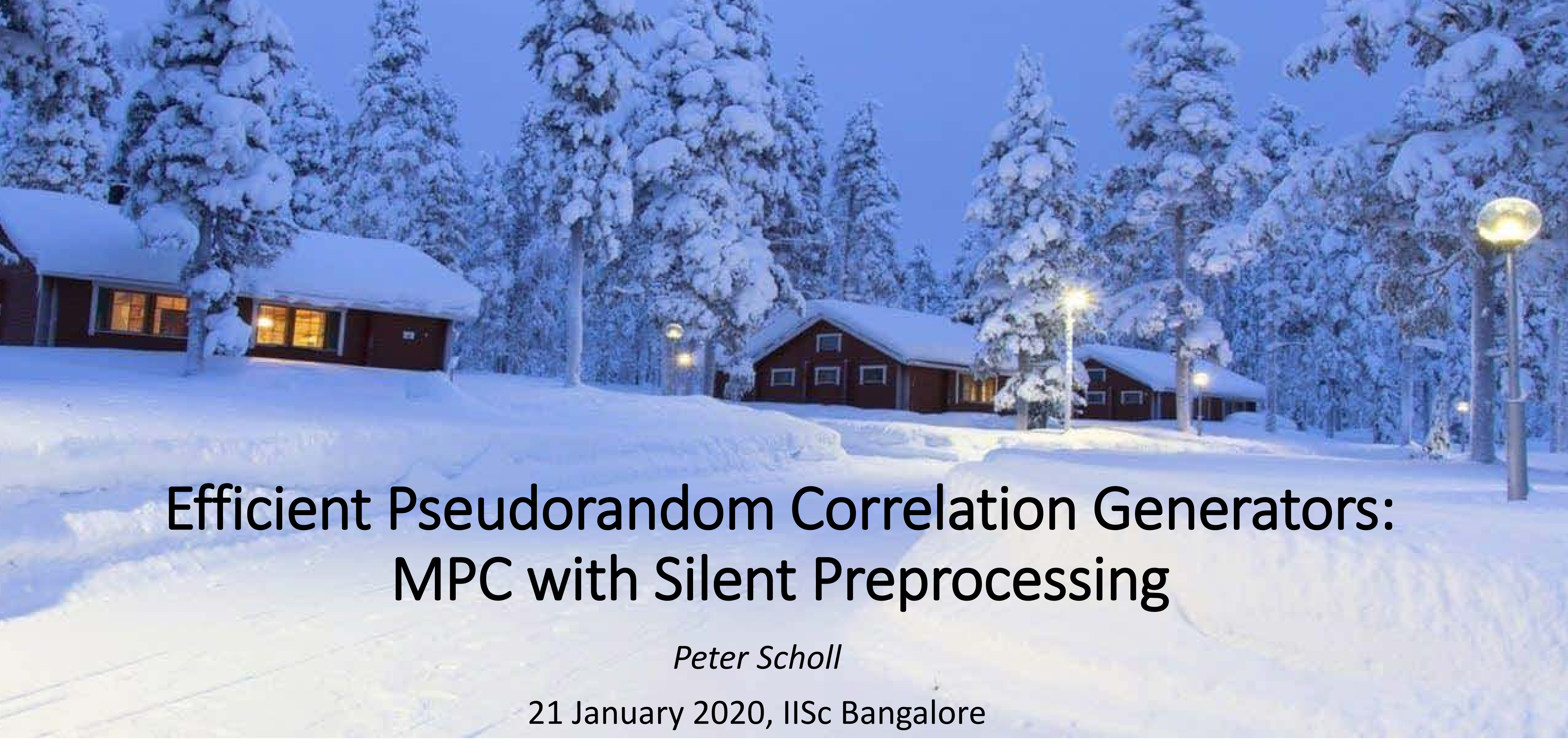


Efficient Pseudorandom Correlation Generators: MPC with Silent Preprocessing

Peter Scholl

21 January 2020, IISc Bangalore

Joint work with:

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

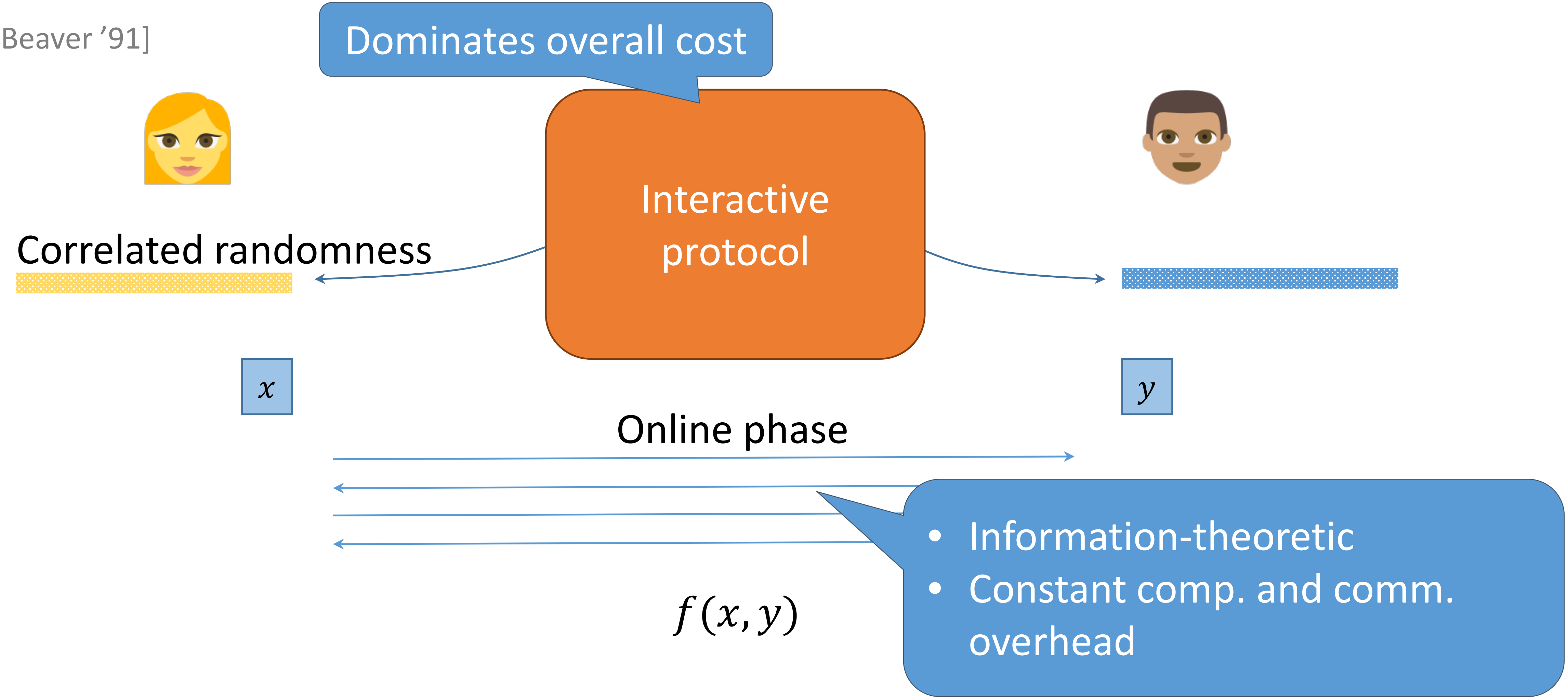
AARHUS UNIVERSITY

Outline

- Pseudorandom correlation generators (PCGs)
  - ➢Motivation: MPC in the preprocessing model
- Why LPN is a perfect match for HSS/PCGs
- PCG for OT from LPN:
  - ➢Two-round "silent" OT extension
  - ➢Practical
- PCG for OLE from LPN
  - ➢Concretely efficient under variant of ring-LPN

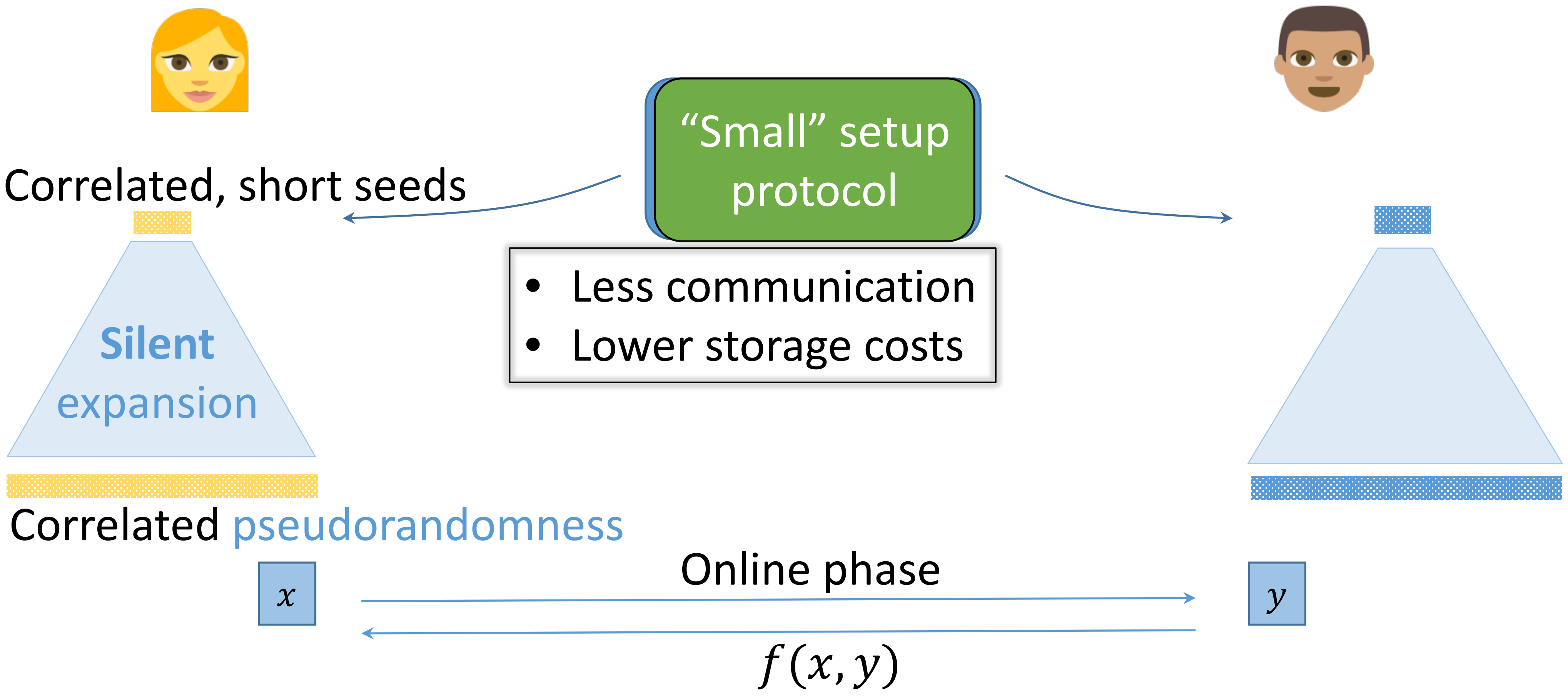
Secure Computation with Preprocessing

[Beaver ’91]



Secure Computation with Silent Preprocessing

[BCGI 18, BCGIKS 19]

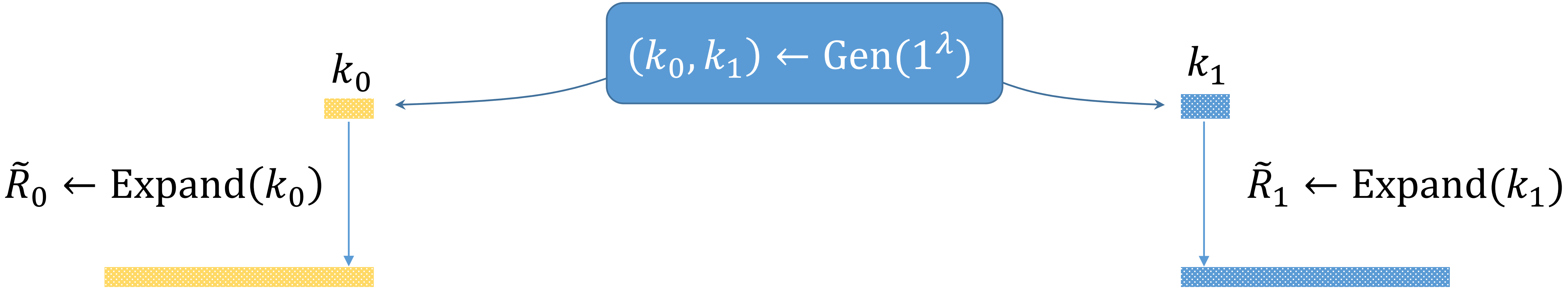


- Less communication
- Lower storage costs

Pseudorandom Correlation Generators

[BCGI 18, BCGIK**S** 19]

- Target correlation: (R_0, R_1)
  - E.g. random OT $((b, m_b), (m_0, m_1))$
- Algorithms Gen, Expand:



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

Landscape of PCGs

"Gentria"		
➢ LWE+	General additive correlations	[BCGIKS 19]
"Cryptomania"		
➢ DDH + low-degree PRG	Low-degree correlations (1/poly error)	[BCGIO 17]
➢ LWE + low-degree PRG	Low-degree correlations	[BCGIKS 19]
"Lapland"		
➢ LPN	vector-OLE	[BCGI 18]
	OT, constant degree correlations	[BCGIKS 19]
"Minicrypt"		
➢ OWF	Linear correlations	[GI 99, CDI 05]
	Truth tables	[BCGIKS 19]

Background: LPN and LWE (spot the difference!)

Given $A \in Z_p^{m \times n}$:

$$A s + e \mod p \approx u$$

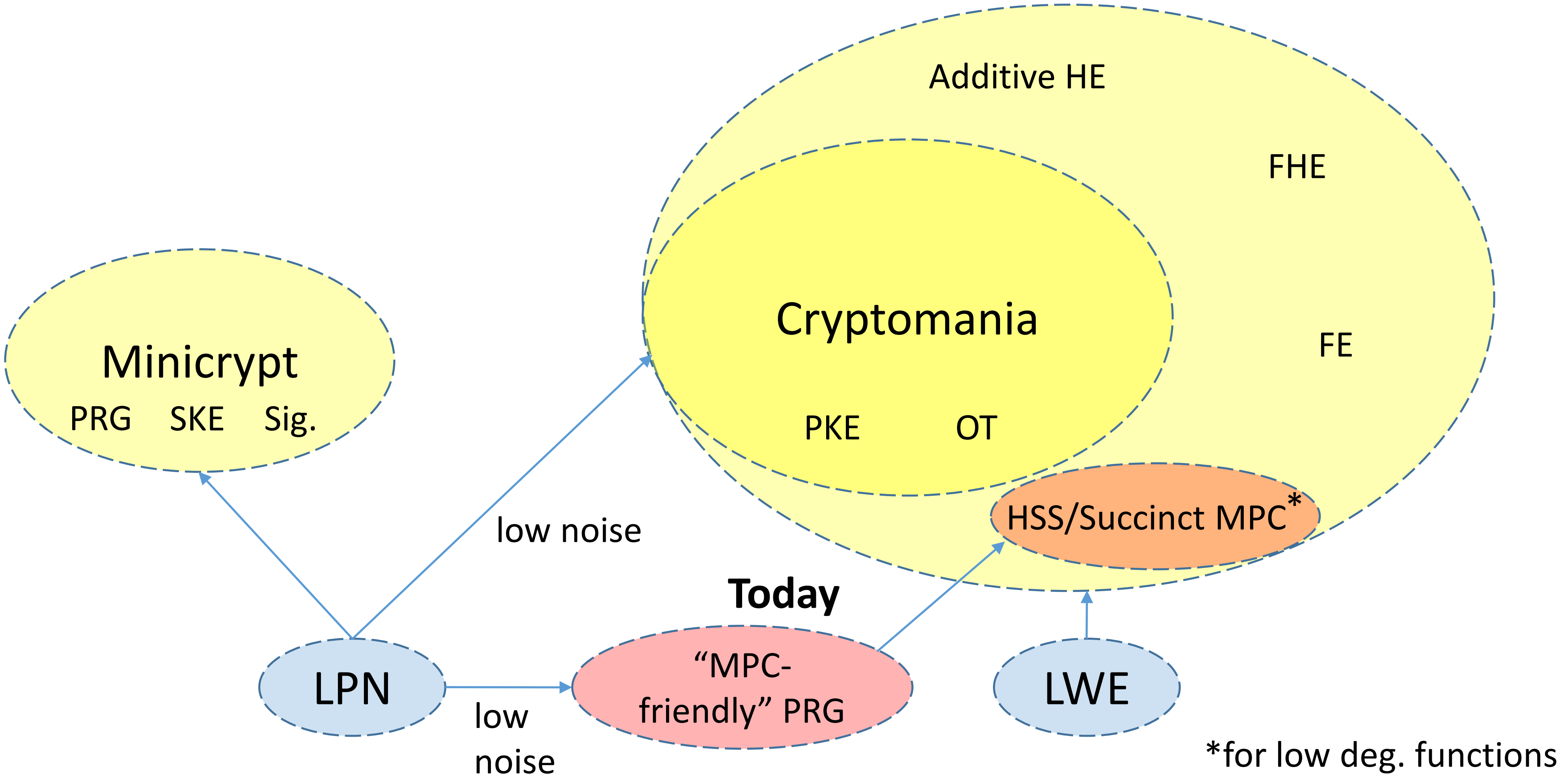
LWE

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

LPN

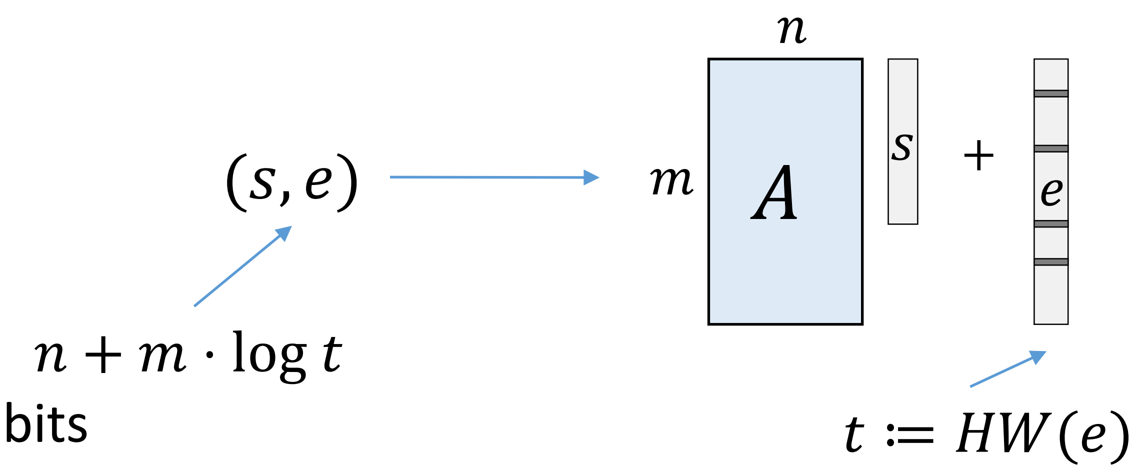
- ~~$p = 2$~~ $p \geq 2$ (arithmetic generalization/RLC)
- $s \leftarrow Z_p^n$
- $HW(e)$ is small

LWE and LPN: what are they good for?

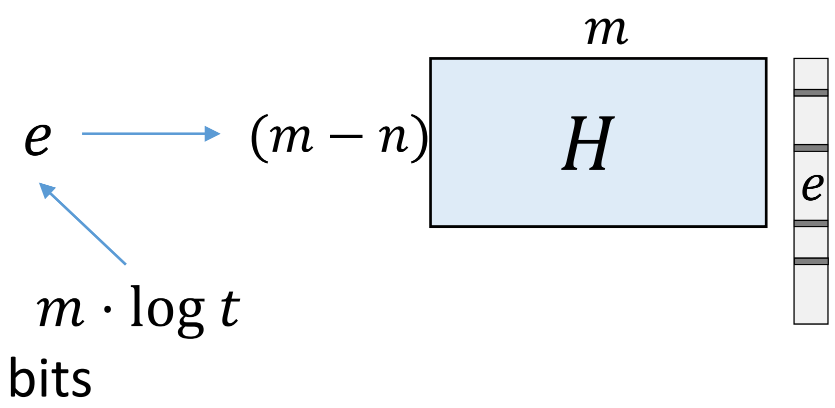


Simple PRGs from LPN

"Primal" construction



"Dual" construction



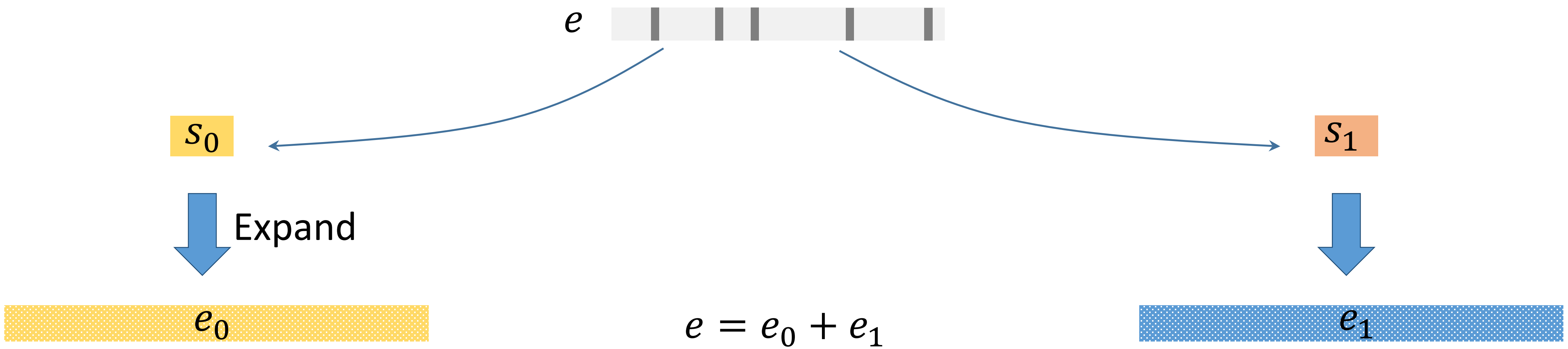
Security: both equiv. to LPN (if H is parity-check matrix of code A)

Limited to quadratic stretch

Arbitrary poly stretch
- best attack: $\exp(t)$

Blueprint: How to exploit sparse noise for PCGs

Step 1: Compress secret-shares of sparse vector with FSS



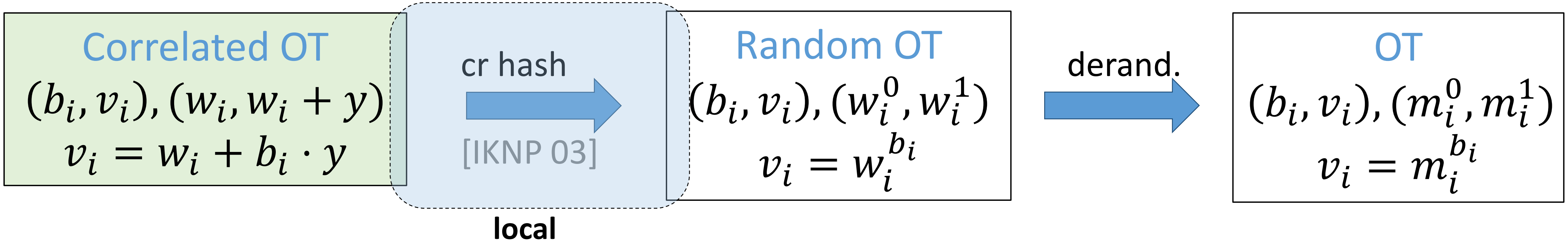
Step 2: Use e as seed for PRG $e \to H \cdot e$

I: PCG for oblivious transfer from LPN

Oblivious Transfer

- **Problem:** OT is expensive ("public-key")
- **OT extension:** many OTs from a few base OTs + symmetric crypto [IKNP 03]
- **Problem:** communication $O(n\lambda)$ for n OTs
- **Silent OT extension:** communication sublinear in n

Towards silent OT extension

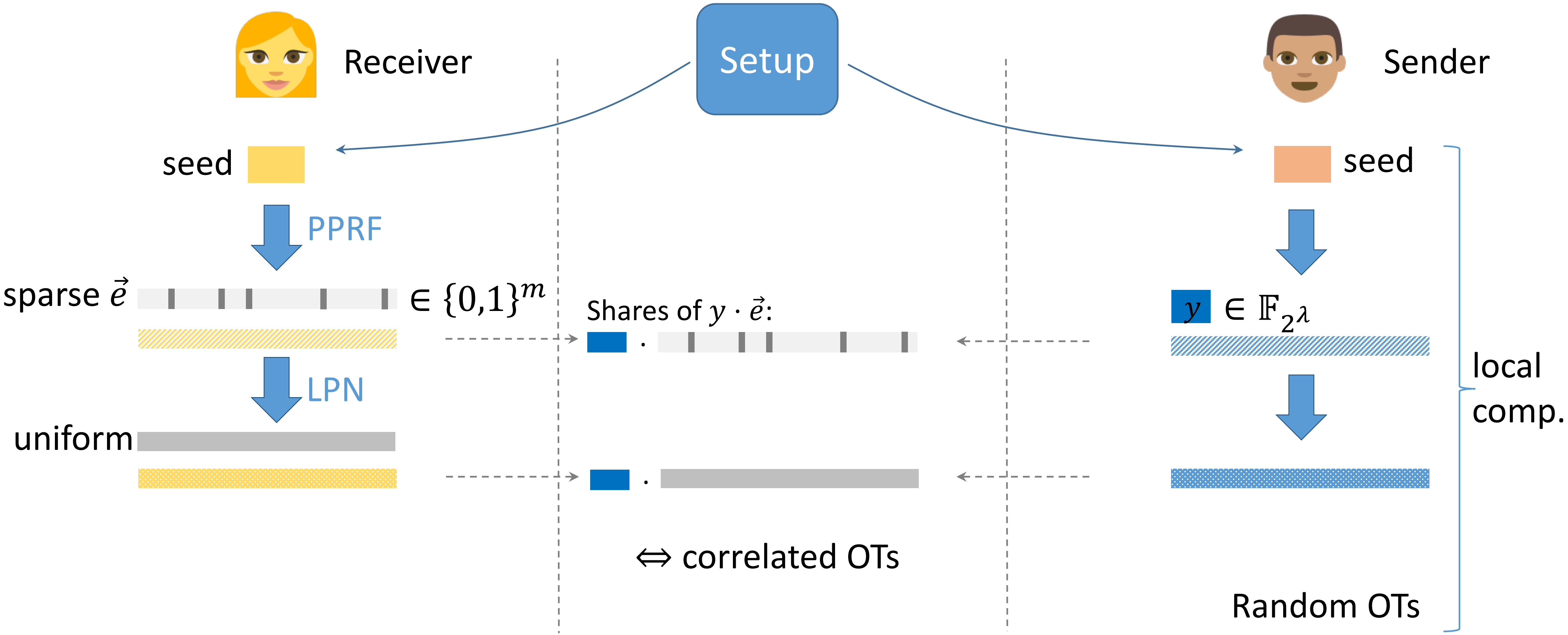


Goal: a PCG for correlated OT

i.e. compression of:

$$\vec{v} + \vec{w} = y \cdot \vec{b}$$

Silent OT Extension: Overview



Main tool: puncturable PRF

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

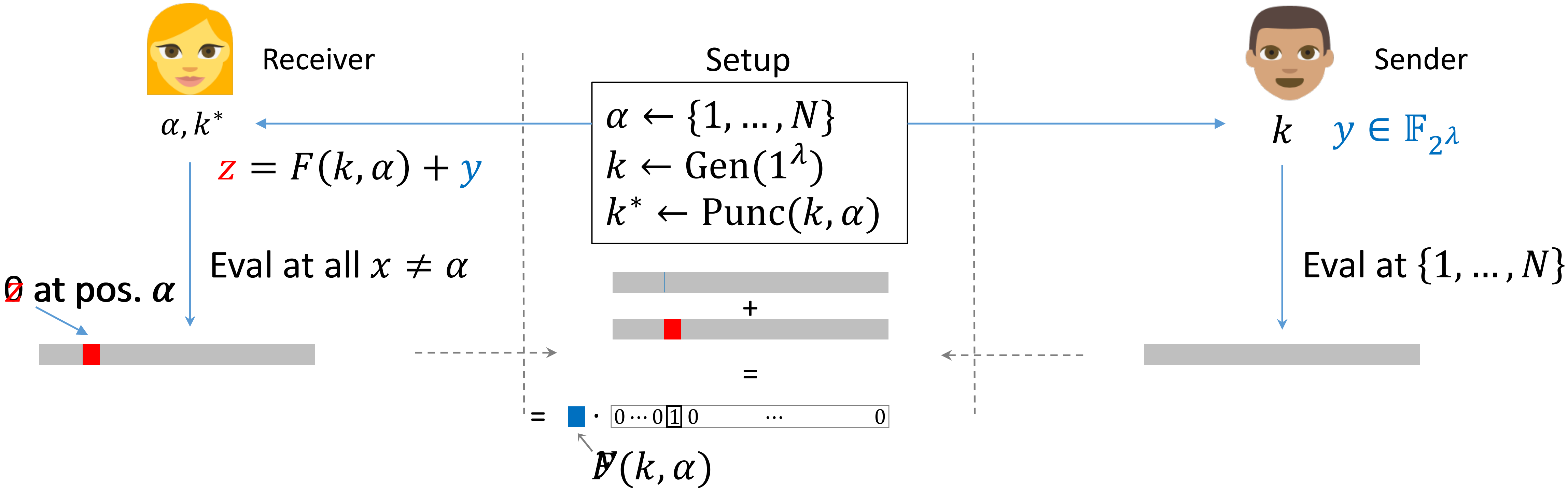
- $k \leftarrow \mathrm{Gen}(1^\lambda)$
  - ➢ Master key: allows evaluating $F(k, x)$ for all x
- $k^* \leftarrow \mathrm{Punc}(k, \alpha)$
  - ➢ Punctured key: can evaluate at all points except for $x = \alpha$

- 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]

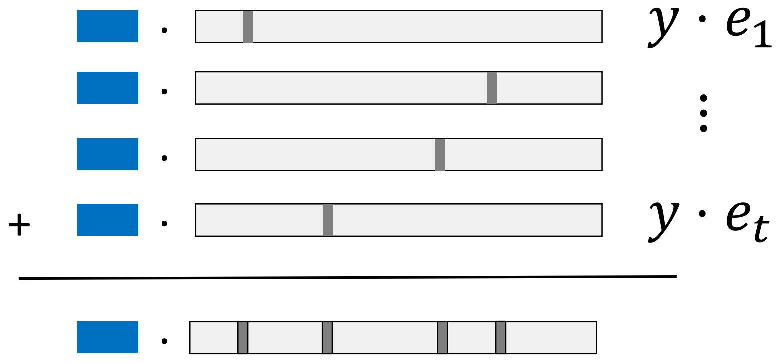
Key observation: puncturable PRF compresses sparse vectors



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

From weight-1 vectors to weight-t vectors

Approach 1: addition

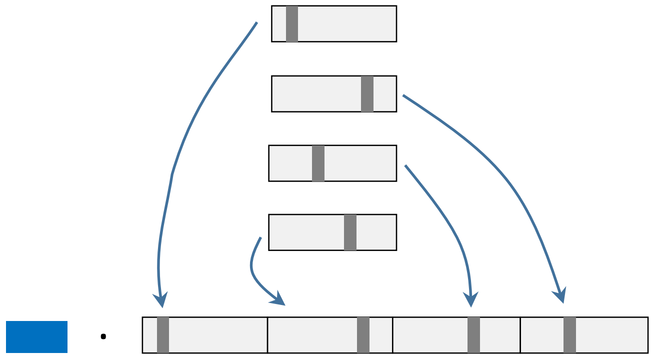


Weight e.g. $t = 4$

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

$O(N)$ (batch codes [BCGI18, SGRR 19])

Approach 2: concatenation

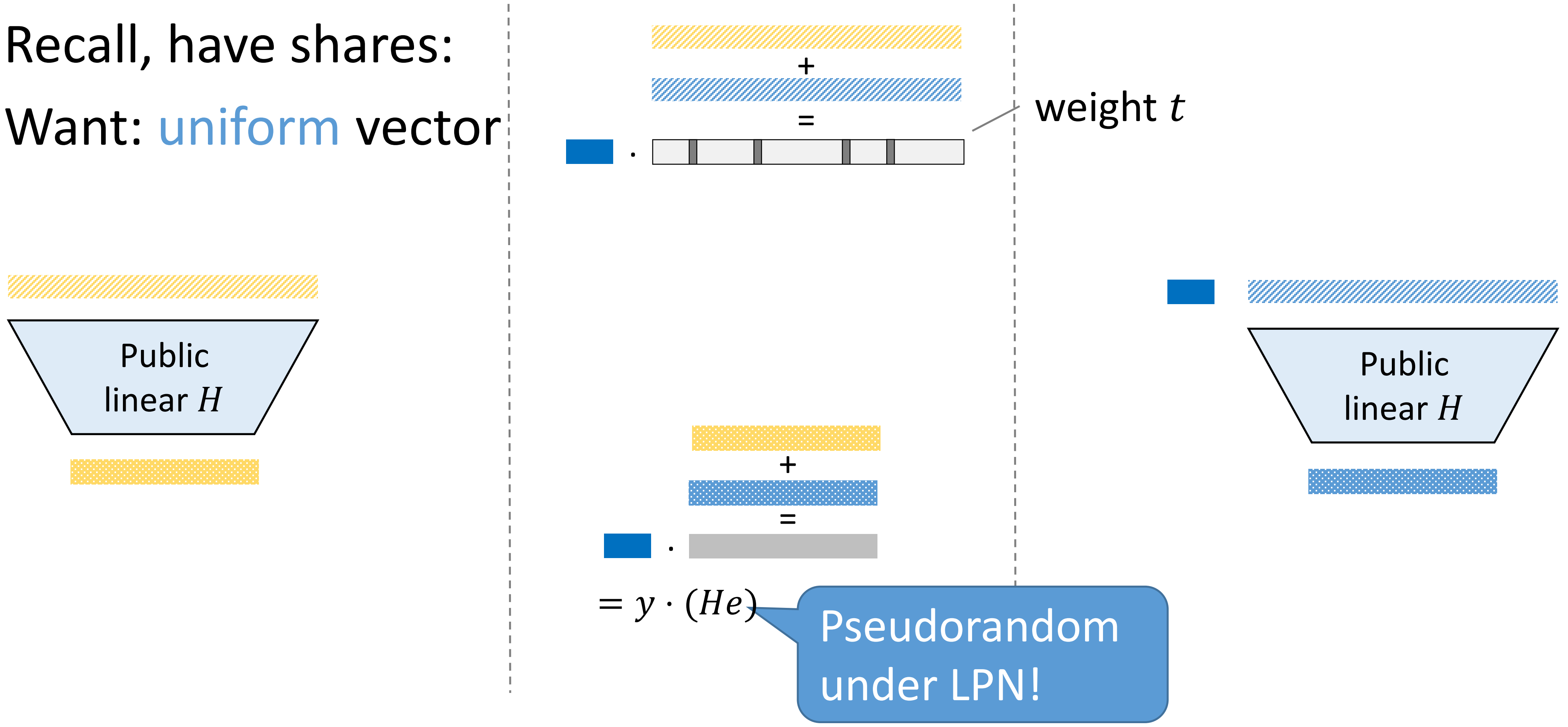


$$O\left(t \cdot \frac{N}{t}\right) = O(N)$$

Note: regular error pattern

From sparse products to correlated OT

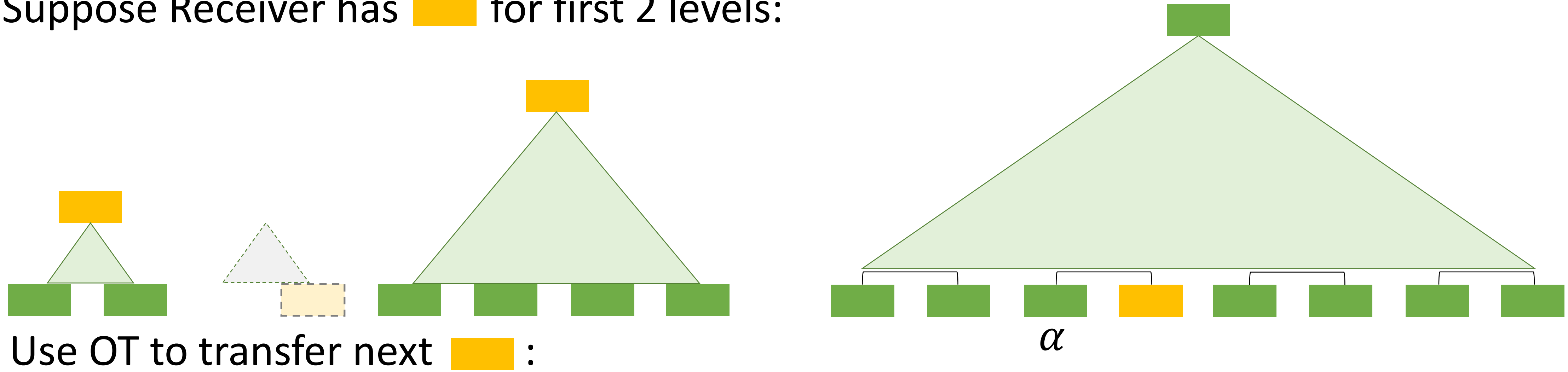
- Recall, have shares:
- Want: uniform vector



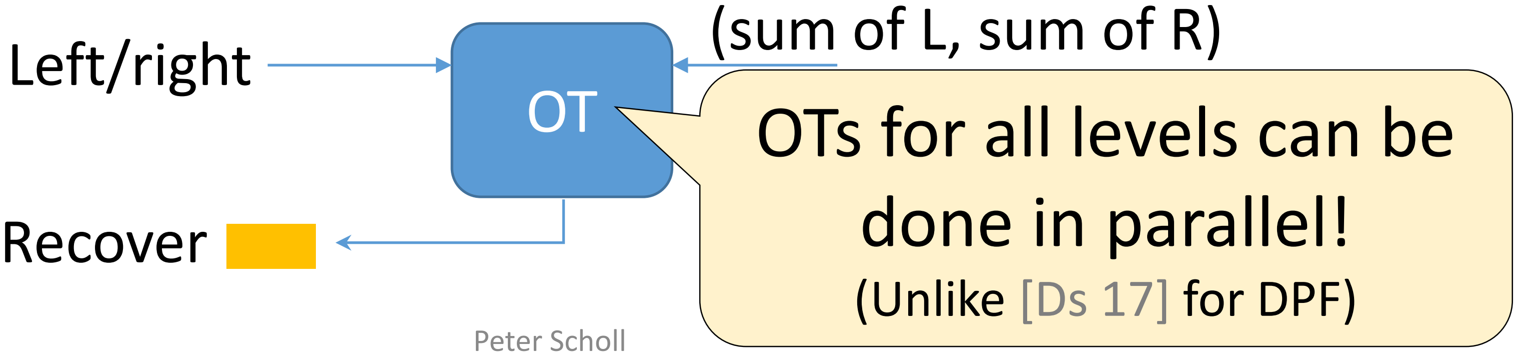
Setup protocol: inside the puncturable PRF

Based on [Doerner-shelat ‘17]

Suppose Receiver has ■ for first 2 levels:



Use OT to transfer next ■ :



Recap: silent OT extension

- Setup protocol: 2 rounds from any 2-round OT
  - ➢Cost: $O(\lambda \log N)$ base Ots

- Silent expansion (N OTs):
  - ➢$O(N \log N)$ PRF evaluations
  - ➢1 multiplication $H \cdot x$

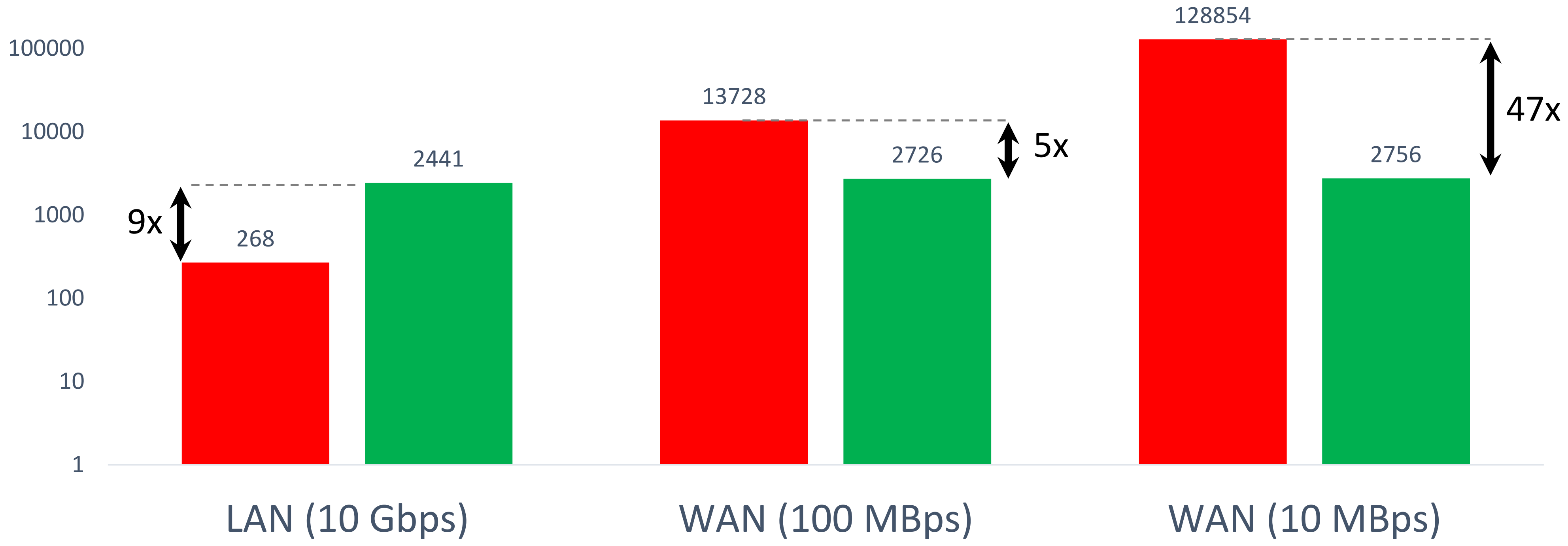
- Implies two-round OT extension on chosen inputs
  - ➢Can convert from random → chosen in parallel with setup
  - ➢First concretely efficient two-round OT extension (bypass [GMMM 18] impossibility via LPN)

Extras: active security, implementation

- **Active security:**
  - Lightweight PPRF consistency checks for malicious sender
    - Allows selective failure attacks – sender can guess 1 bit of LPN error
    - Assume problem is hard with 1-bit leakage
  - 10-20% overhead on top of semi-honest

- **Implementation:**
  - Main challenge: fast mult. by H
  - Quasi-cyclic H: polynomial mult. mod $X^n - 1$
  - Security based on quasi-cyclic syndrome decoding / ring-LPN

Runtimes (ms) for *n*=10 million random OTs



IKNP vs silent OT

Total comm: 160 MB vs 127 kB

II: PCG for OLE correlations from LPN and ring-LPN

Degree-2 correlation: Oblivious Linear Evaluation (OLE)

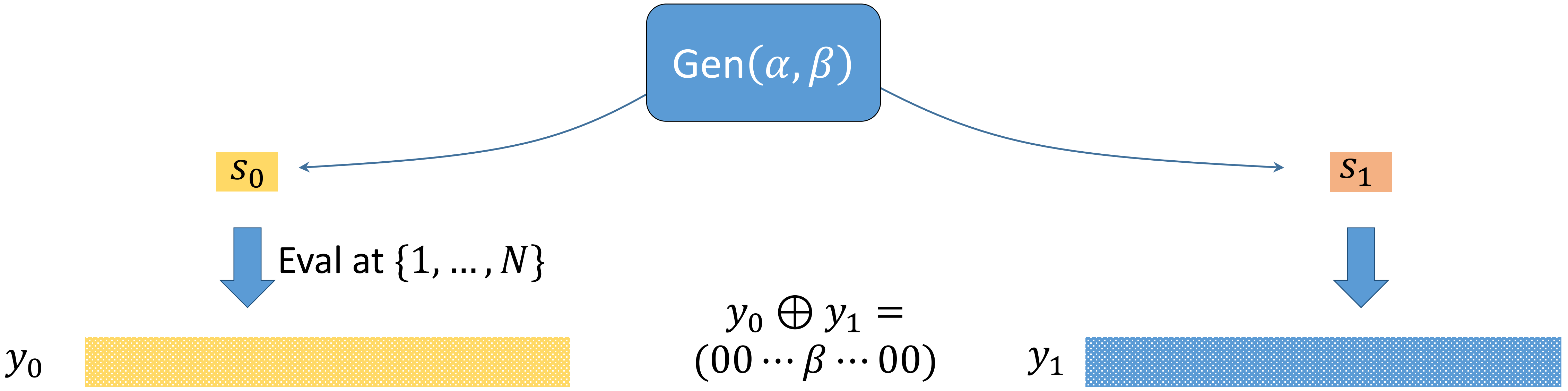
Related: multiplication triples

- Obtained from 2 random OLEs (two parties)

Main tool: FSS for point functions

- Point function $f_{\alpha,\beta}: \{1, \ldots, N\} \to \{0,1\}^{\lambda}$

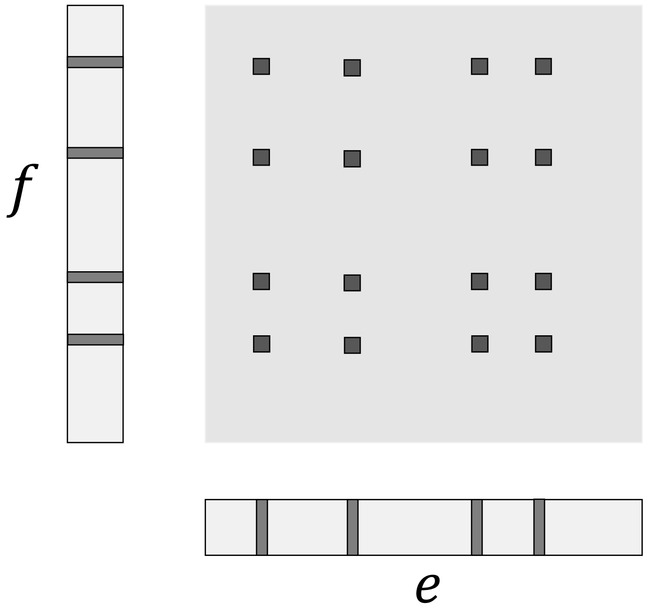
$$f_{\alpha,\beta}(x) = \begin{cases} \beta & \text{if } x = \alpha \\ 0 & \text{o.w.} \end{cases}$$



PCG for tensor product from LPN and FSS

[BCGIKS '19]

- Pick e, f with $HW\ t$
- Tensor product $e \otimes f$ is sparse
- Distribute shares of e, f and $e \otimes f$
  - ➢ With FSS for $O(t^2)$ points

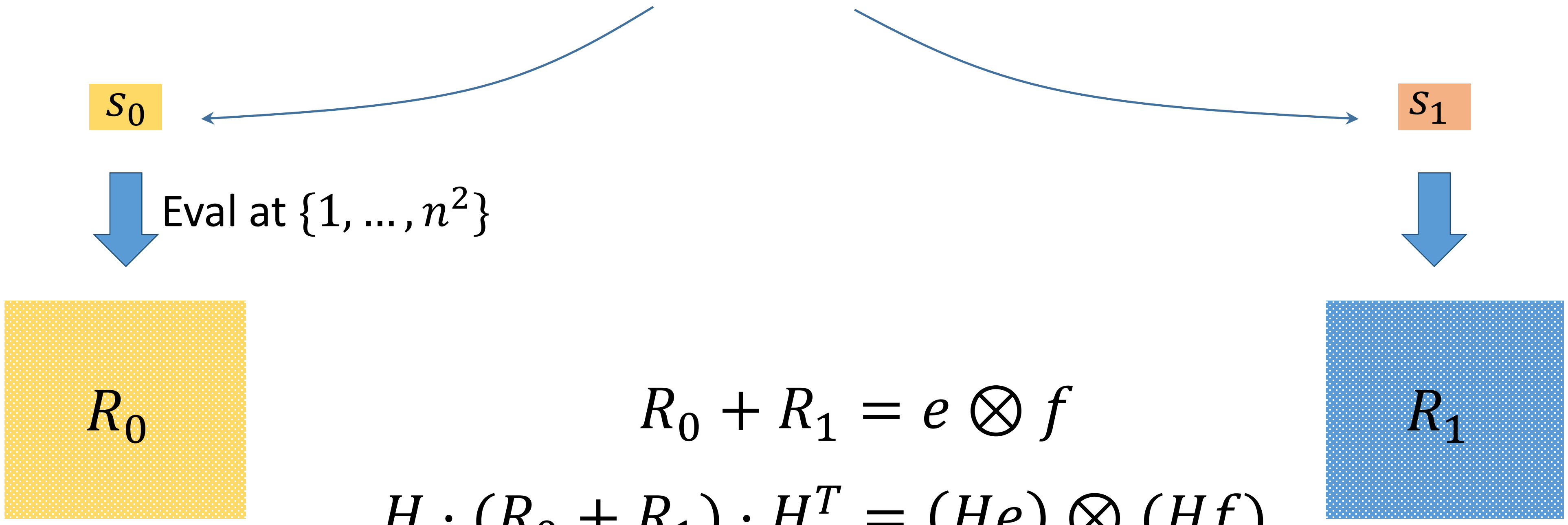


PCG for tensor product from LPN and FSS

[BCGIKS '19]

t-sparse e, f

t^2- point FSS for $e \otimes f$



$$R_0 + R_1 = e \otimes f$$

$$H \cdot (R_0 + R_1) \cdot H^T = (He) \otimes (Hf)$$

Applications of PCG for tensor product

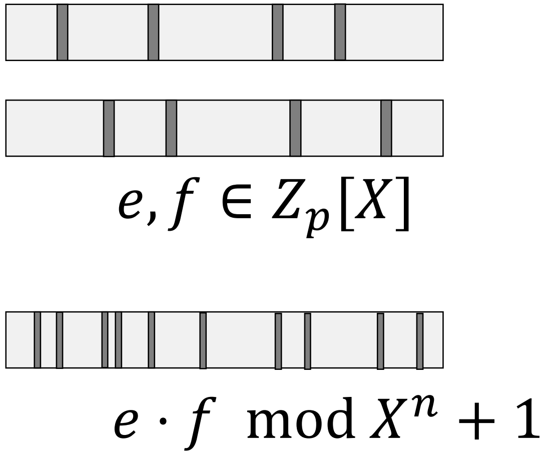
- Deg-2 correlations:
  - n OLEs or Beaver triples with $o(n)$ communication
  - Computation: $\Omega(n^2)$
  - Extends to deg-d (cost: $\Omega(n^d)$)

- PCG for deg-d $\Rightarrow$ homomorphic secret-sharing for deg-d functions
  - Let (Gen, Expand) be PCG for $\mathrm{R} = [r, r \otimes r, \dots, r \otimes^d r]$
  - **Share**(x): apply Gen and make $x' = x + r$ public
  - **Eval$_p$**: write $p(x)$ as $p'(r)$, where p' is determined by x', and linear in R

Efficient PCG for OLE from ring-LPN

[ongoing work]

- Idea:
  - ➢ Replace tensor product with polynomial multiplication
  - ➢ Similar to [BV11] for FHE



- Take sparse polys e, e', f, f'
- Distribute shares of $(e, e') \otimes (f, f')$
- Output

$$(he + e') \cdot (hf + f') \mod (X^n + 1)$$

Linear in $(e, e') \otimes (f, f')$

for public, random $h \in Z_p[X]$

Efficient PCG for OLE from ring-LPN
[ongoing work]

- **Cost**: for 1 OLE in $Z_p[X]/(X^n + 1)$
  - ➢ $O(t^2 + n \log n)$ computation

Gives n OLEs in Z_p if $X^n + 1$ splits into linear factors mod p

- **Security:**
  - ➢ Arithmetic ring-LPN

$$(h, h \cdot s + e) \quad \bmod (p, F(X))$$

  - ➢ Does not appear significantly weaker

Conclusion

- PCG for OT from LPN
  - Random OT (and correlated OT): practical, almost zero communication
  - (previously: λ bits per OT)
  - Two-round OT extension
- PCG for OLE
  - From LPN (expensive)
  - Efficient from fully splitting ring-LPN
- Open problems:
  - Optimize OT: better codes
  - Security of arithmetic ring-LPN
  - Efficient PCGs for more correlations:
    - Truth tables (active security), random bits ($\mathbb{Z}_p$), garbled circuits…

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

Code: https://github.com/osu-crypto/libOTe