# Pseudorandom Correlation Generators


**Yuval Ishai**

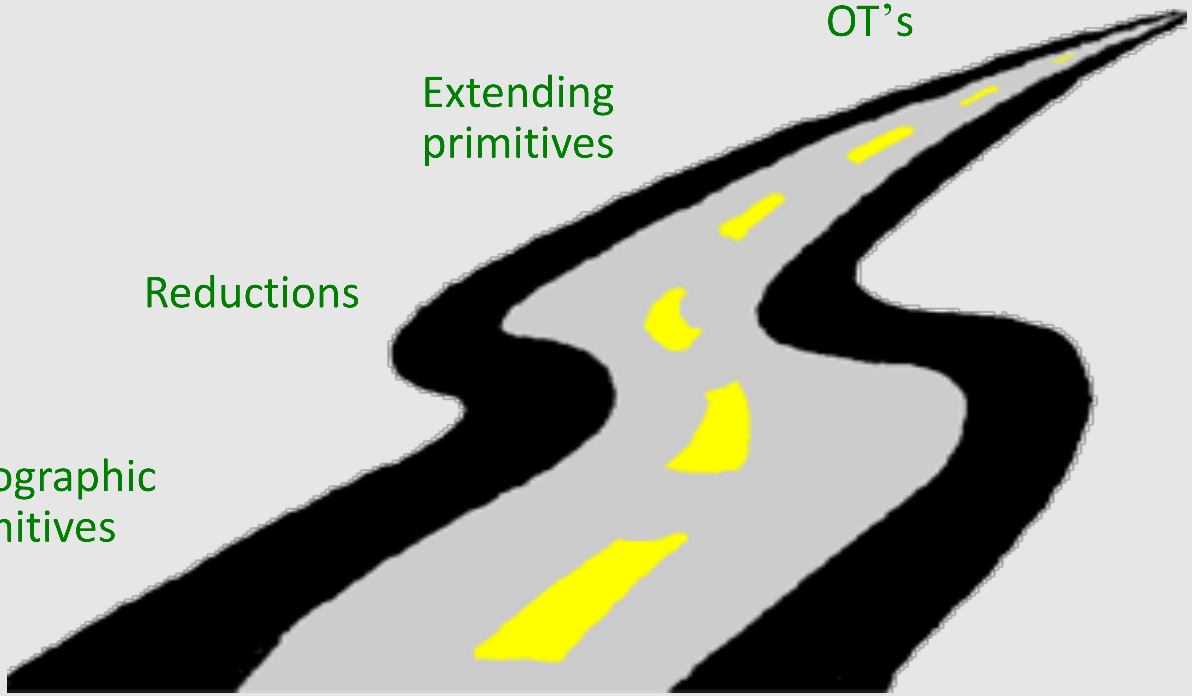
Technion


Mostly based on works with Elette Boyle, Geoffroy Couteau,
Niv Gilboa, Lisa Kohl, and Peter Scholl

# Road Map

IKNP, Crypto 2003
"Extending Oblivious Transfers Efficiently"

OT Factory



Extending OT's

Extending primitives

Reductions

Cryptographic primitives

# Road Map

Today's lectures

Part I

Motivation

Definitions

Constructions from PRG

Part II

Constructions From LPN

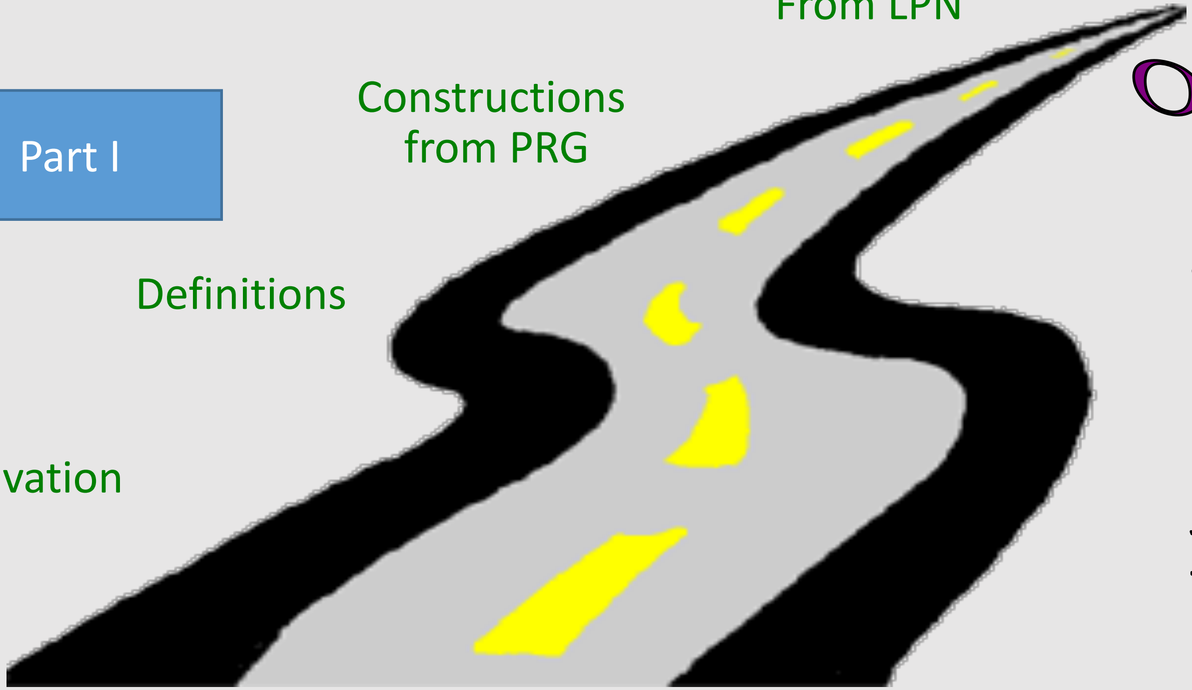
Silent OT Factory

+

VOLE

+

more...



Part III

Part IV

Peter tomorrow

# Background and Motivation

# Secure (2-Party) Computation

[Yao86,GMW87]


$f(x, y)$

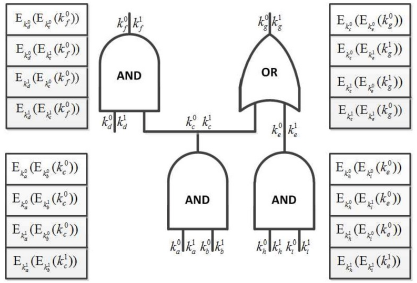
Learn $f(x, y)$ and **nothing else** about $x, y$

# Secure Computation Paradigms

2 semi-honest parties

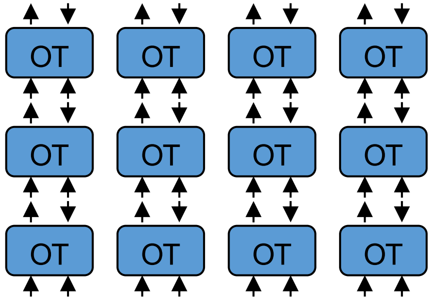
## Garbled Circuits

[Yao 86,…]



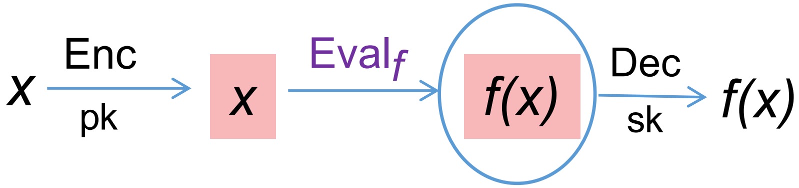

## Linear Secret Sharing

[Goldreich-Micali-Wigderson 87, …]




## Fully Homomorphic Encryption

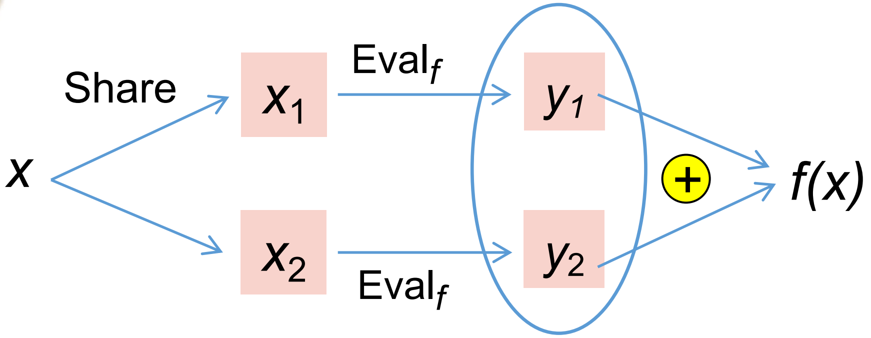
[Gentry 09,…]




new

## Homomorphic Secret Sharing

[Boyle-Gilboa-I 15,…]

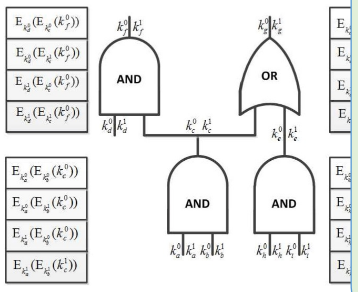


# Secure Computation Paradigms

2 semi-honest parties

## Garbled Circuits

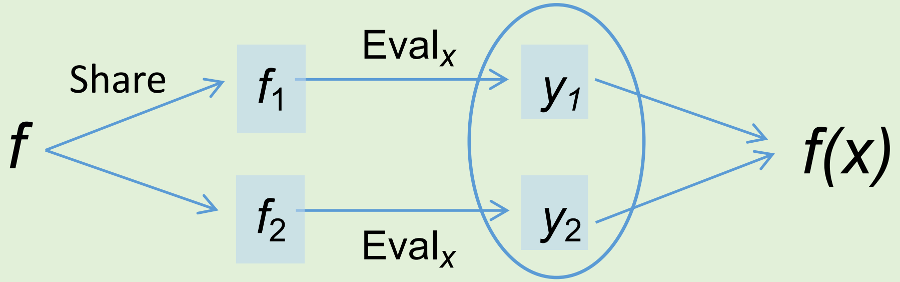
[Yao 86,…]



## Linear Secret Sharing

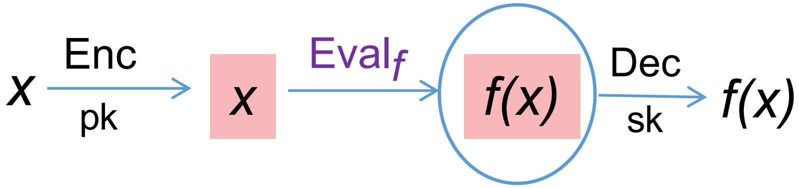
[Goldreich-Micali-Wigderson 87, …]

## Function Secret Sharing




## Fully Homomorphic Encryption

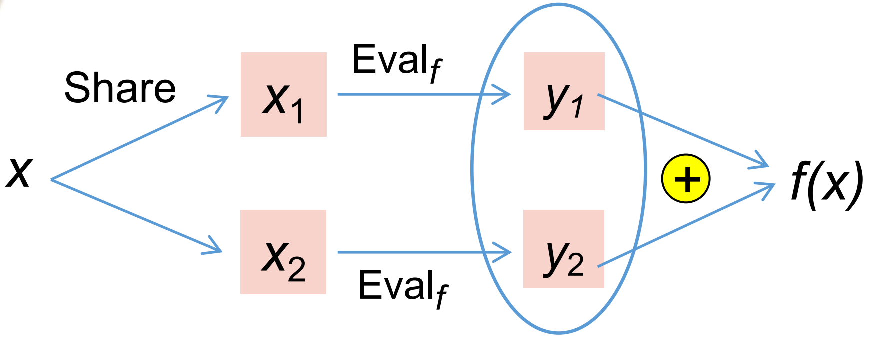
[Gentry 09,…]




## Homomorphic Secret Sharing

[Boyle-Gilboa-I 15,…]

new



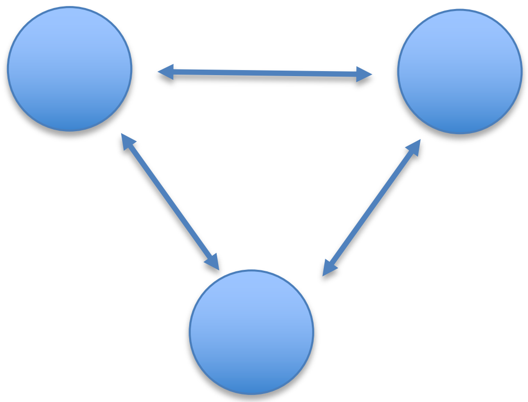
# Current HSS Worlds

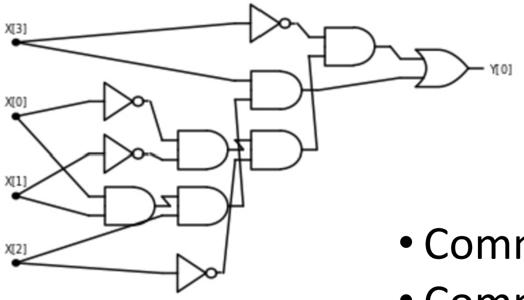
| | | | |
|---|---|---|---|
| "Homomorphia" | | | |
| | – LWE+ | Circuits | [DHRW16, BGI15, BGILT18] |
| "Cryptomania" | | | |
| | – DDH | Branching Programs | [BGI16, BCGIO17, DKK18] |
| | – Paillier | Branching Programs | [FGJS17, OSY21, RS21] |
| | – LWE | Branching Programs | [BKS19] |
| "Lapland" | | | |
| | - LPN | Low-degree polynomials | [BCGI18,BCGIKS19,BCGIKS20,CM21] |
| "Minicrypt" | | | |
| | – OWF | Point Functions<br>Intervals<br>Decision Trees | [GI14, BGI15, BGI16] |
| "Algorithmica" | | | |
| | – None | Linear Functions | [Ben86] |

# Challenge

Honest-majority 3PC

[BGW88, CCD88, ALFNO16]

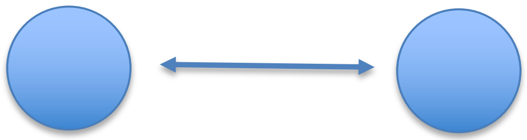





Cost per AND

- Communication: 1 bit per party
- Computation: cheaper…

Dream goal for 2PC



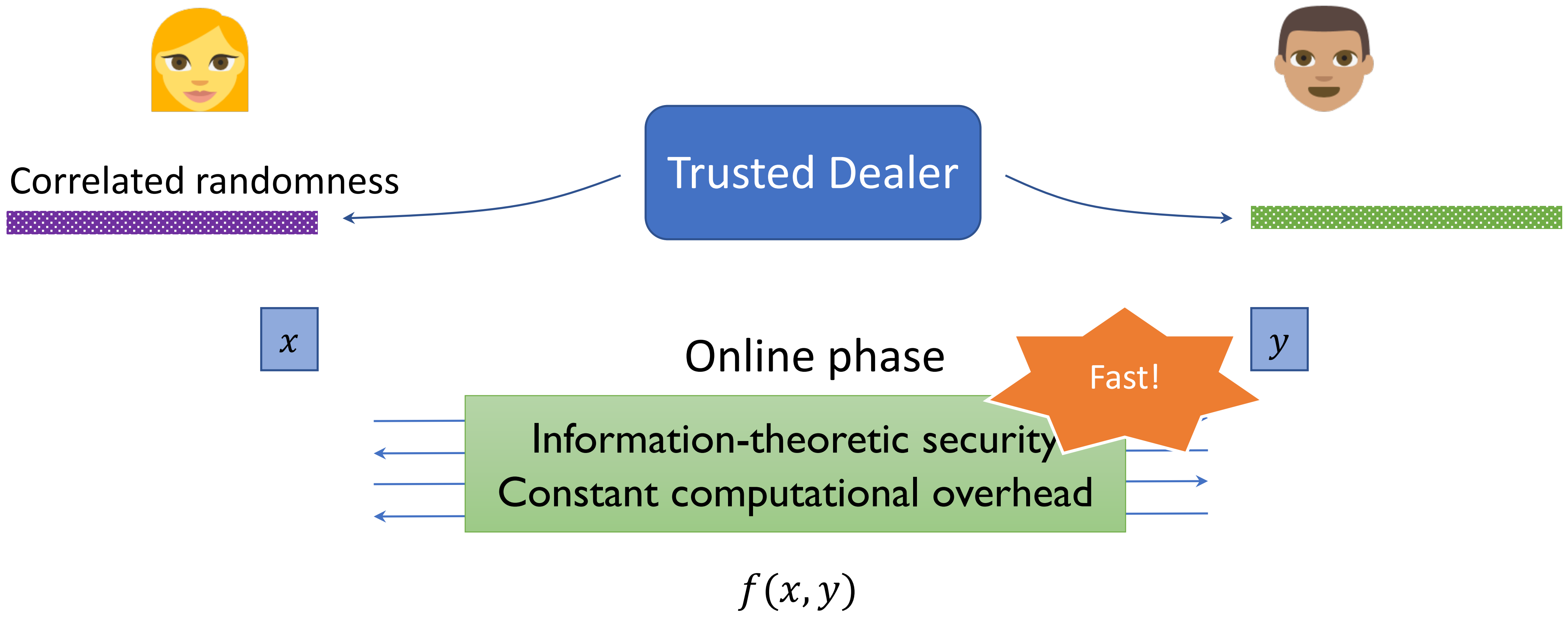
Same?

FHE / HSS: heavy computation

Yao / GMW+ OT extension: heavy communication

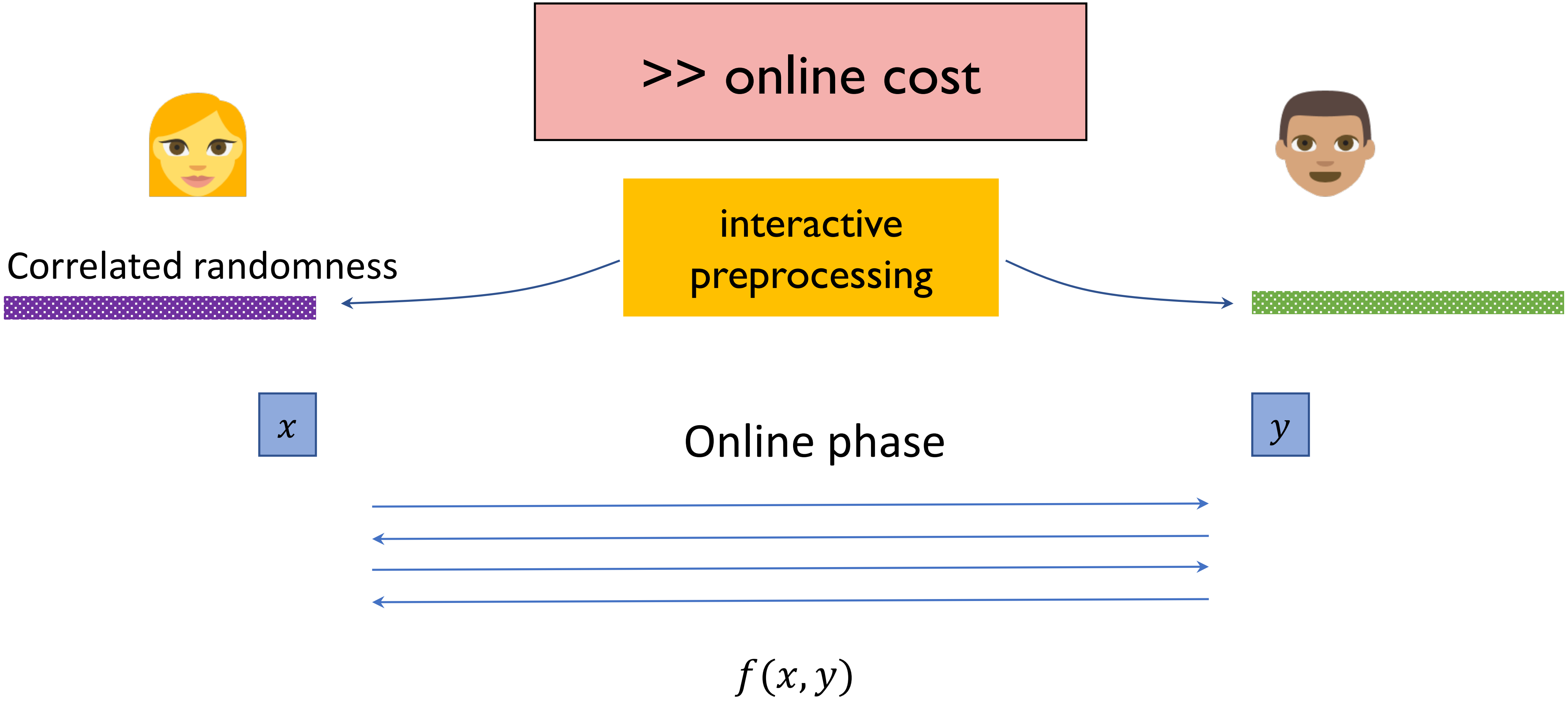
# Meeting challenge using correlated randomness

[Beaver ’91]



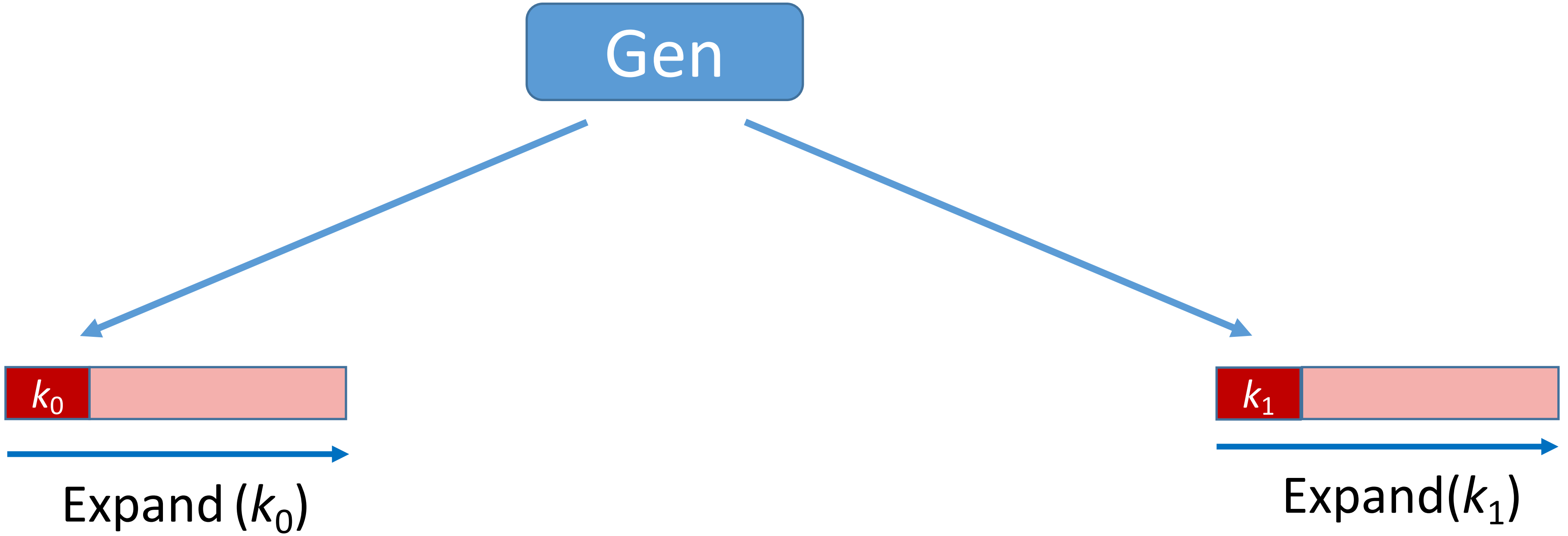

[Bea95, Bea97, IPS08, BDOZ11, BIKW12, NNOB12, DPSZ12, IKMOP13, DZ13, DLT14, BIKK14, LOS14, FKOS15, DZ16, KOS16, DNNR17, Cou19, BGI19, BNO19, CG20, BGIN21,... ]

# Meeting challenge without correlated randomness?



# Pseudorandom Correlation Generator (PCG)

[Boyle-Couteau-Gilboa-I18, BCGI-Kohl-Scholl19]




Target correlation: $(R_0, R_1)$

$\left(\text{Expand}(k_0), \text{Expand}(k_1)\right) \approx (R_0, R_1)$

Also for insiders!

# Secure Computation with Silent Preprocessing

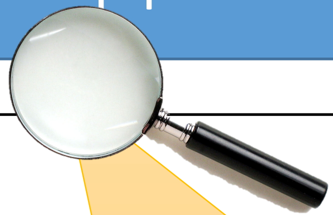
| Phase 1: | Phase 2: | Phase 3: |
|---|---|---|
| cheap PCG seed setup protocol | silent seed expansion | fast, "non-cryptographic" |
| offline | | online |

- Total communication & online computation meet challenge
  - Fast Expand ➔ fully meet challenge!
- Malicious security with **vanishing** amortized cost

# Secure Computation with Silent Preprocessing

| Phase 1: | Phase 2: | Phase 3: |
| --- | --- | --- |
| cheap PCG seed setup protocol | silent seed expansion | fast, "non-cryptographic" |

offline (Phases 1–2) — online (Phase 3)



- ✓ Ad-hoc future interactions
- ✓ Hiding communication pattern
- ✓ Hiding future plans

Concrete cost of setup:
Peter's talk tomorrow

# Secure Computation with Silent Preprocessing

| Phase 1: | Phase 2: | Phase 3: |
| --- | --- | --- |
| cheap PCG seed setup protocol | silent seed expansion | fast, “non-cryptographic” |

offline (Phases 1–2) — online (Phase 3)

## Non-cryptographic online phase?

- Know it when you see it…
- Efficiency: asymptotic and concrete
- “Indistinguishable from info-theoretic”

Main difference from Laconic SFE [QuachWeeWichs18]

# Definitions

# PCG Security Definition: Take I

- Real = $(k_0, \mathrm{Expand}(k_1)) \approx (\mathrm{Sim}(R_0), R_1)$ = Ideal

Securely realizing ideal correlation functionality $(R_0, R_1)$

Good for all applications

Not realizable even for simple correlations

# PCG Security Definition: Take II

- Real = $(k_0, \mathrm{Expand}(k_1)) \approx (\mathrm{Sim}(R_0), R_1)$ = Ideal
- Real = $(k_0, \mathrm{Expand}(k_1)) \approx (k_0, [R_1 \mid R_0 = \mathrm{Expand}(k_0)])$

Securely realizing "corruptible" target correlation

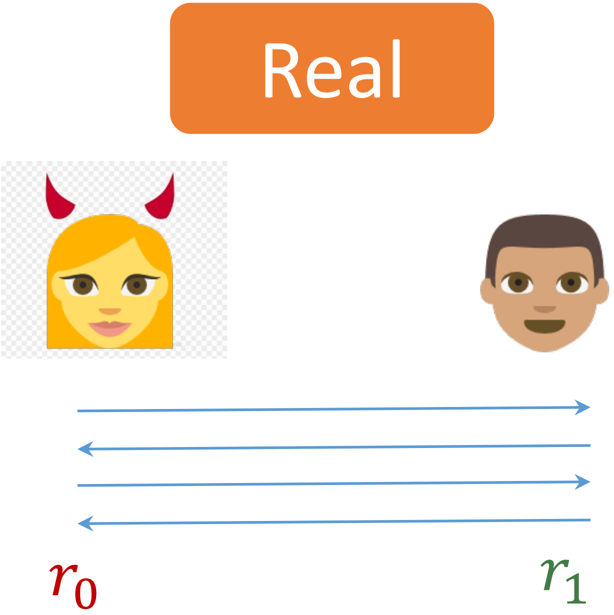
Good for natural applications

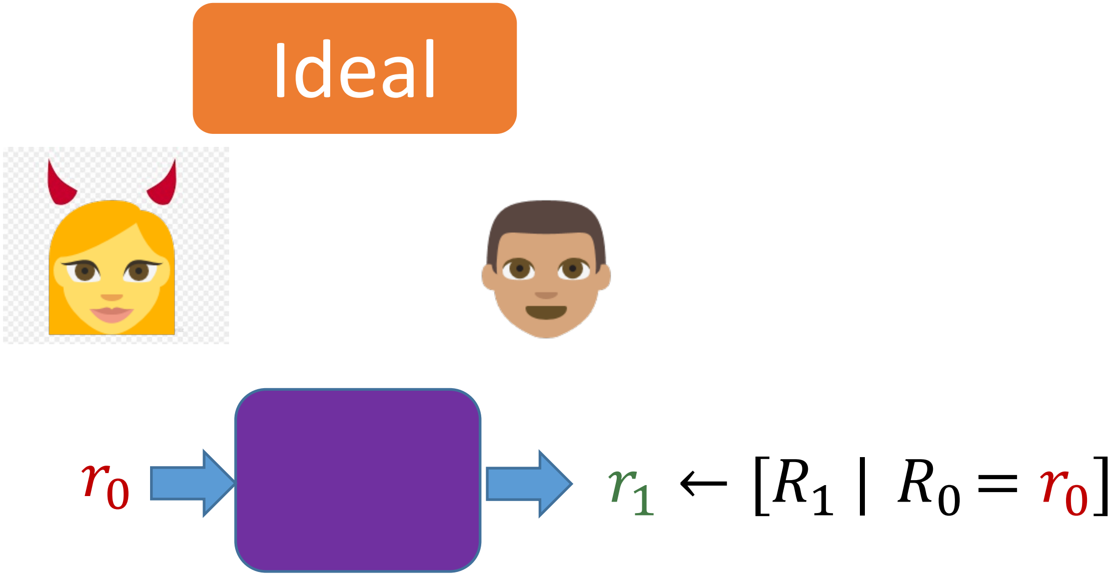
Realizable for useful correlations

# PCG protocol

Naturally extends to n parties

- Combines Setup + Expand
- Sublinear-communication protocol for corruptible version of $(R_0, R_1)$

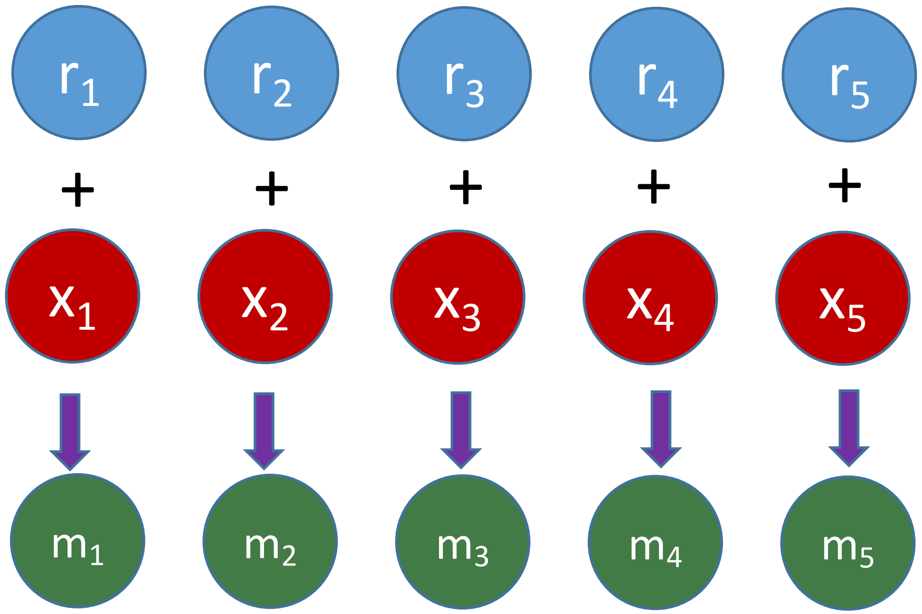


# Correlations

# Useful target correlations: 3+ parties

| | | |
|---|---|---|
| Linear n-party correlations | $(R_1, \dots, R_n) \in_R$ Linear space V<br>N x deg-t Shamir of random secret<br>N x additive shares of 0 | VSS, honest-majority MPC<br>Proactive secret sharing<br>Secure sum / aggregation |



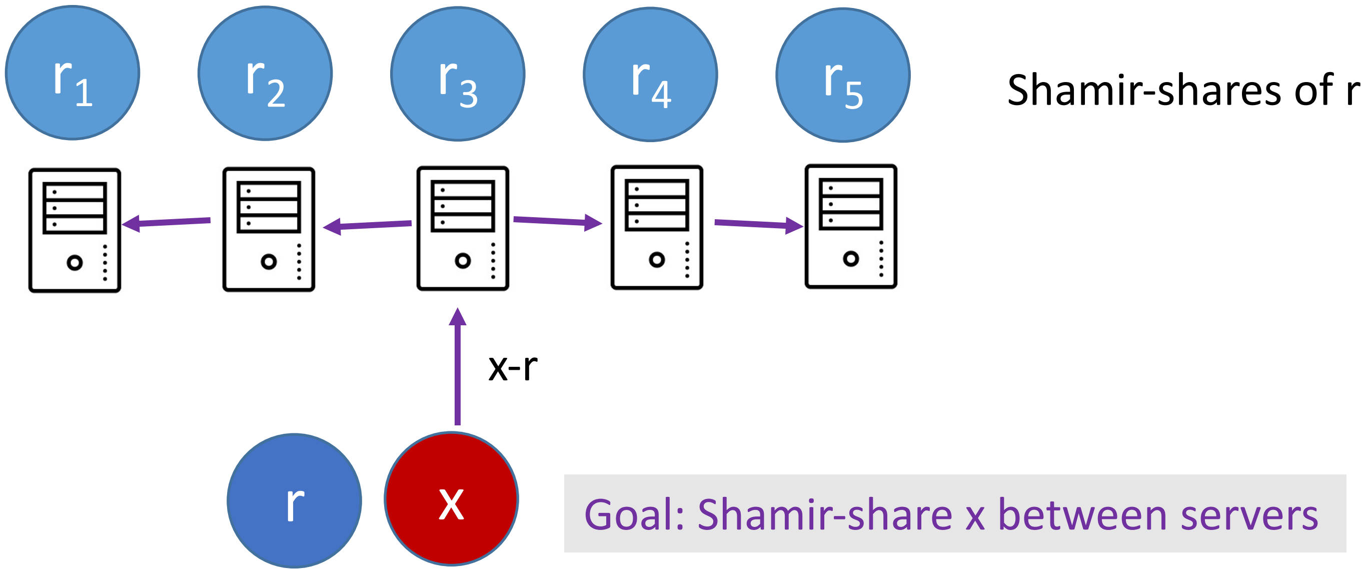

Additive shares of 0

Goal: securely aggregate

Additive shares of $\sum x_i$

# Useful target correlations: 3+ parties

| | | |
|---|---|---|
| Linear n-party correlations | $(R_0, \ldots, R_n) \in_R$ Linear space V<br>N x deg-t Shamir of random secret<br>N x additive shares of 0 | VSS, honest-majority MPC<br>Proactive secret sharing<br>Secure sum / aggregation |




Goal: Shamir-share x between servers

# Useful target correlations: 2+ parties

**Oblivious transfer (OT)**

N x $(s_0,s_1)$ ⇦ OT ⇨ $(c,s_c)$

2PC of Boolean circuits

GMW-style, semi-honest:

2 x bit-OT + 4 comm. bits per AND

**Oblivious Linear-function Evaluation (OLE), mult. triples**

N x (a,b) ⇦ OLE ⇨ (x,ax+b)

2PC of Arithmetic circuits

GMW-style, semi-honest:

2 x OLE + 4 ring elements per MULT

**Vector OLE (VOLE)**

$(\mathbf{a},\mathbf{b})$ ⇦ VOLE ⇨ $(x,\mathbf{a}x+\mathbf{b})$

2PC of scalar-vector product

ZK, batch-OPRF, PSI, ...

(Yesterday - Peter’s talks)

# Useful target correlations: 2+ parties

| | | |
|---|---|---|
| Authenticated Multiplication Triples | $([a_i],[b_i],[c_i], [\alpha a_i],[\alpha b_i],[\alpha c_i])$<br>$c_i=a_i b_i$ | 2PC of Arithmetic circuits<br>SPDZ-style, malicious |
| Truth-Table | Randomly shifted, secret-shared TT | 2PC of "unstructured" functions |
| Additive | R0+R1 = R | Generalizes the above |

# State of the Art

# Current PCG Feasibility Landscape

| | | | |
|---|---|---|---|
| **"Obfustopia"** | iO | General | [HW15, HIJKR16] |
| **"Homomorphia"** | LWE+ | Additive | [DHRW16, BCGIKS19] |
| **"Cryptomania"** | DDH,DCR | Low-depth | [BGI16, BCGIO17, OSY21] |
| **"Lapland"** | LPN | VOLE, OT | [BCGI18, BCGIKS19] |
| | Ring-LPN | OLE, (Auth.) Triples | [BCGIKS20a] |
| | VD-LPN | PCF for VOLE, OT | [BCGIKS20b] |
| **"Minicrypt"** | PRG | Linear | [GI99, CDI05, BBGHIN21] |
| | | Truth table | [BCGIKS19] |

# Current PCG Feasibility Landscape

| | | | |
|---|---|---|---|
| **"Obfustopia"** | iO | General | [HW15, HIJKR16] |
| **"Homomorphia"** | LWE+ | Additive | [DHRW16, BCGIKS19] |
| **"Cryptomania"** | DDH,DCR | Low-depth | [BGI16, BCGIO17, OSY21] |
| **"Lapland"** | LPN<br>Ring-LPN<br>VD-LPN | Constant-degree additive<br>(poly(N) expansion time) | |
| **"Minicrypt"** | PRG | Linear [GI99, CDI05, BBGHIN21]<br>Truth table [BCGIKS19] | |

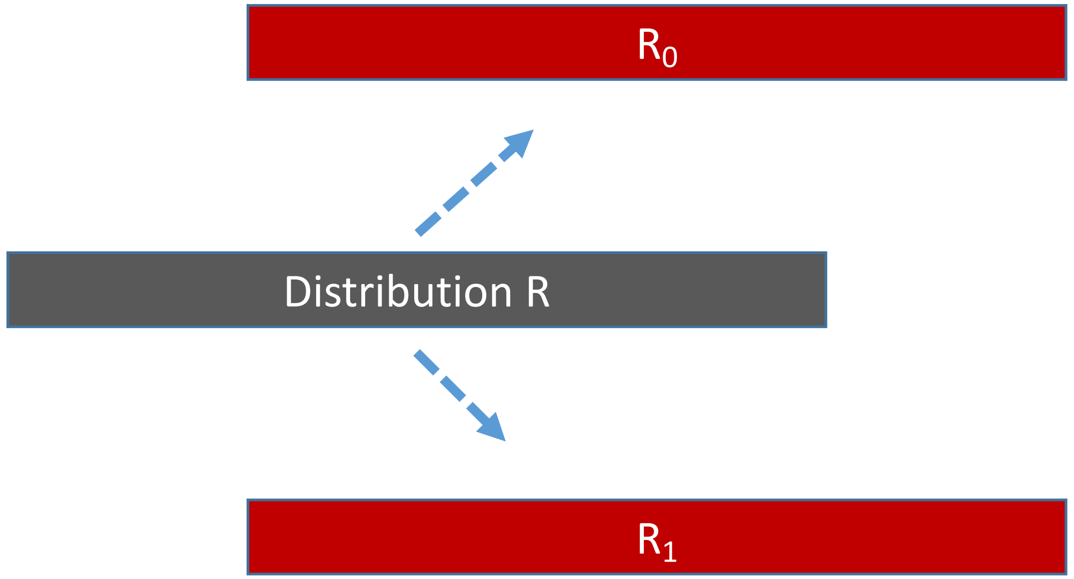
# Good concrete efficiency?

| | | | |
|---|---|---|---|
| **“Obfustopia”** | iO | General | [HW15, HIJKR16] |
| **“Homomorphia”** | | | ] |
| **“Cryptomania”** | | | SY21] |
| **“Lapland”** | LPN | VOLE, OT | [BCGI18, BCGIKS19] |
| | Ring-LPN | OLE, (Auth.) Triples | [BCGIKS20a] |
| | VD-LPN | PCF for VOLE, OT | [BCGIKS20b] |
| **“Minicrypt”** | PRG | Linear | [GI99, CDI05, BBGHIN21] |
| | | Truth table | [BCGIKS19] |

Getting better and better…
[SGRR19, BCGIKRS19, YWLZW20, CRR21]

# Generic Construction from HSS

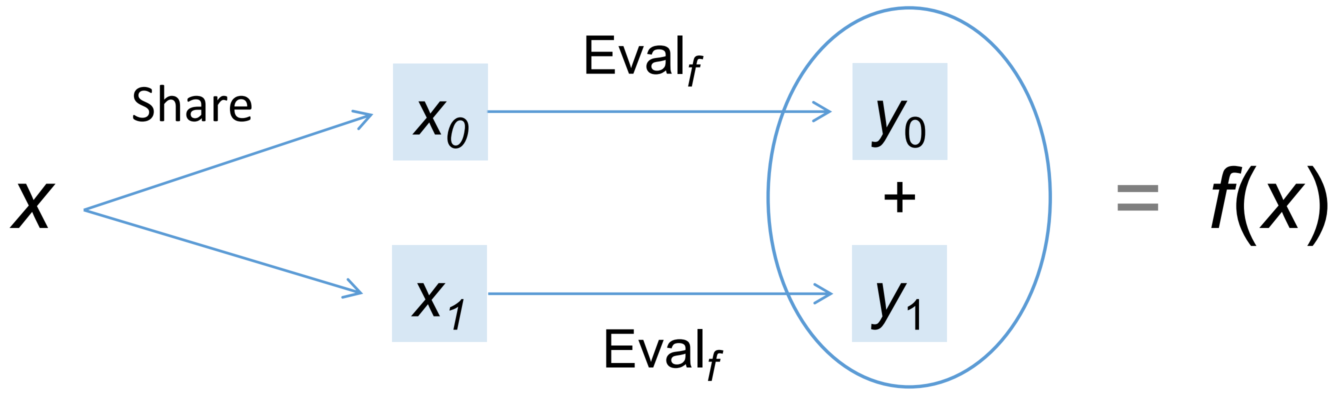
# Additive Correlation




Additive shares

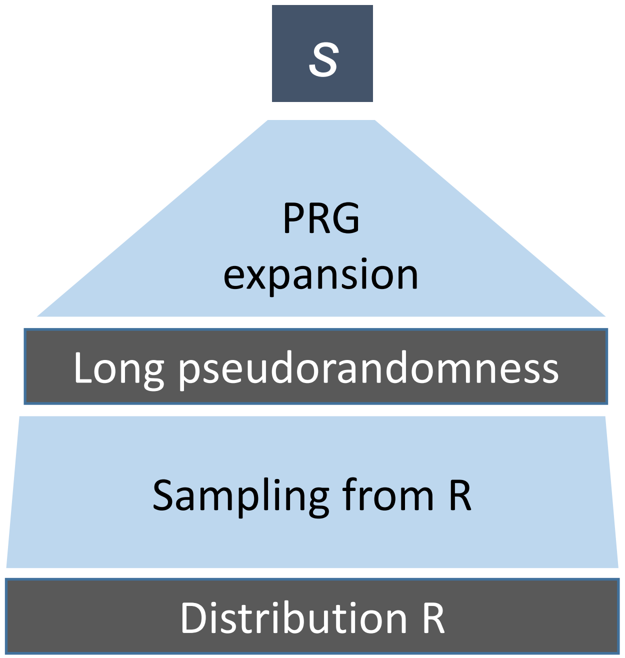
# Homomorphic Secret Sharing (HSS)

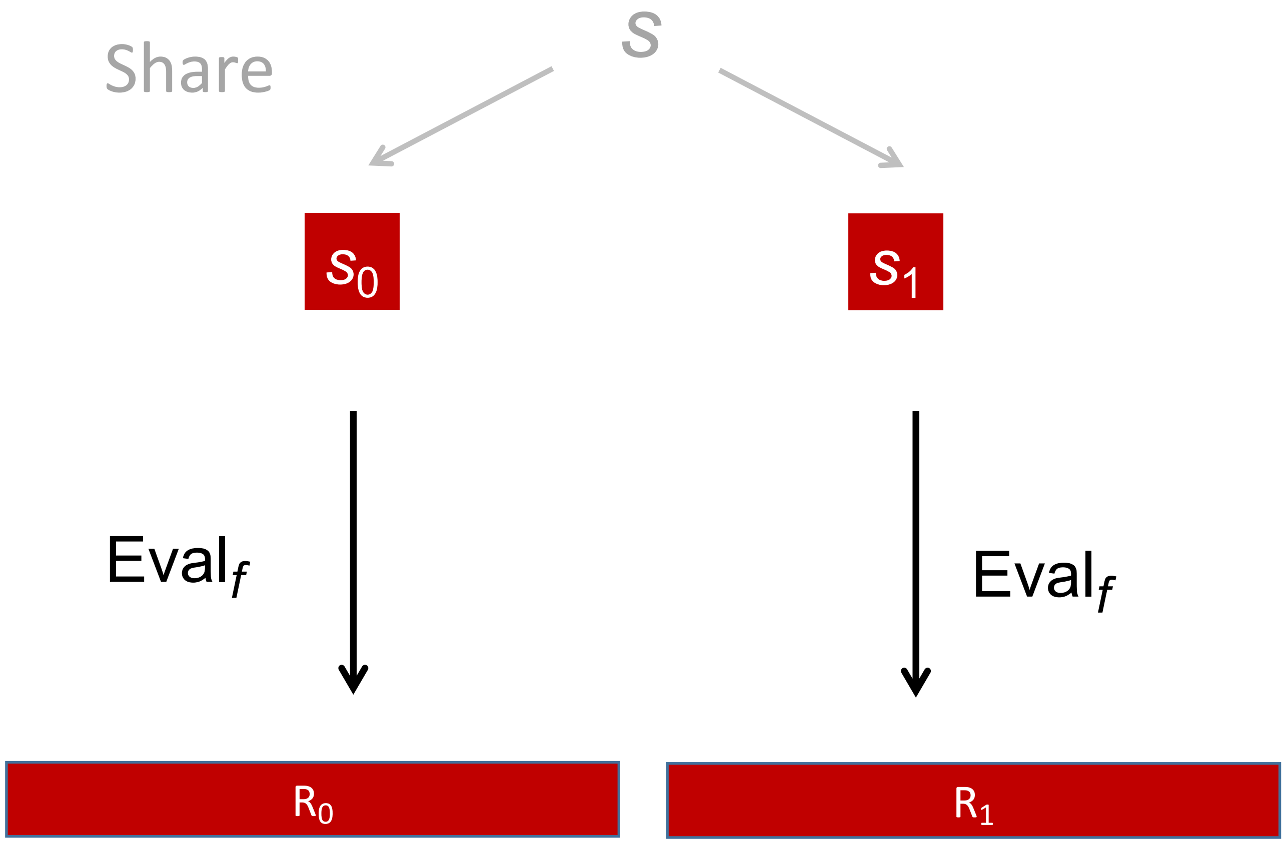
[Benaloh86, Boyle-Gilboa-Ishai16]



# HSS ⇒ PCG for Additive Correlations

Sampling function f:

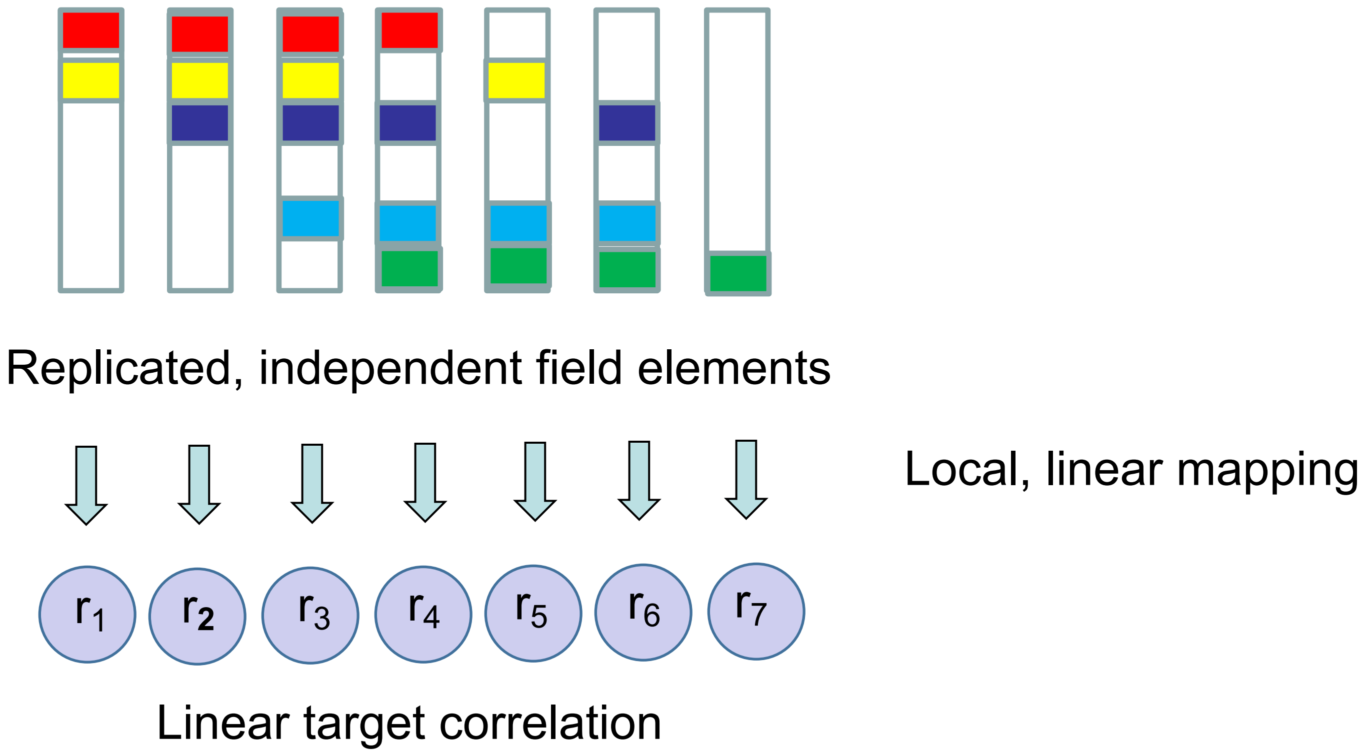


# PCGs in Minicrypt

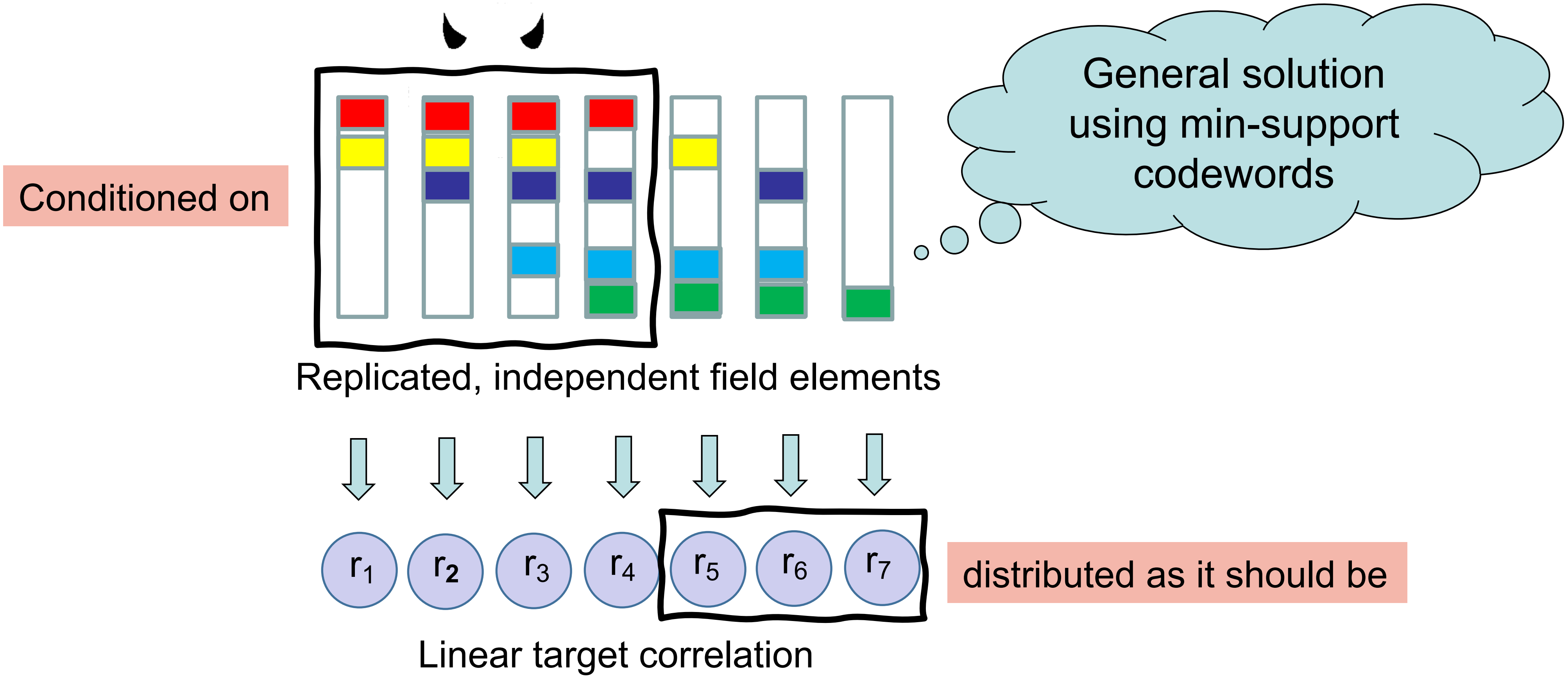
# Linear Multiparty Correlations: Pseudorandom Secret Sharing (PRSS)

[Gilboa-I 99, Cramer-Damgård-I 05]



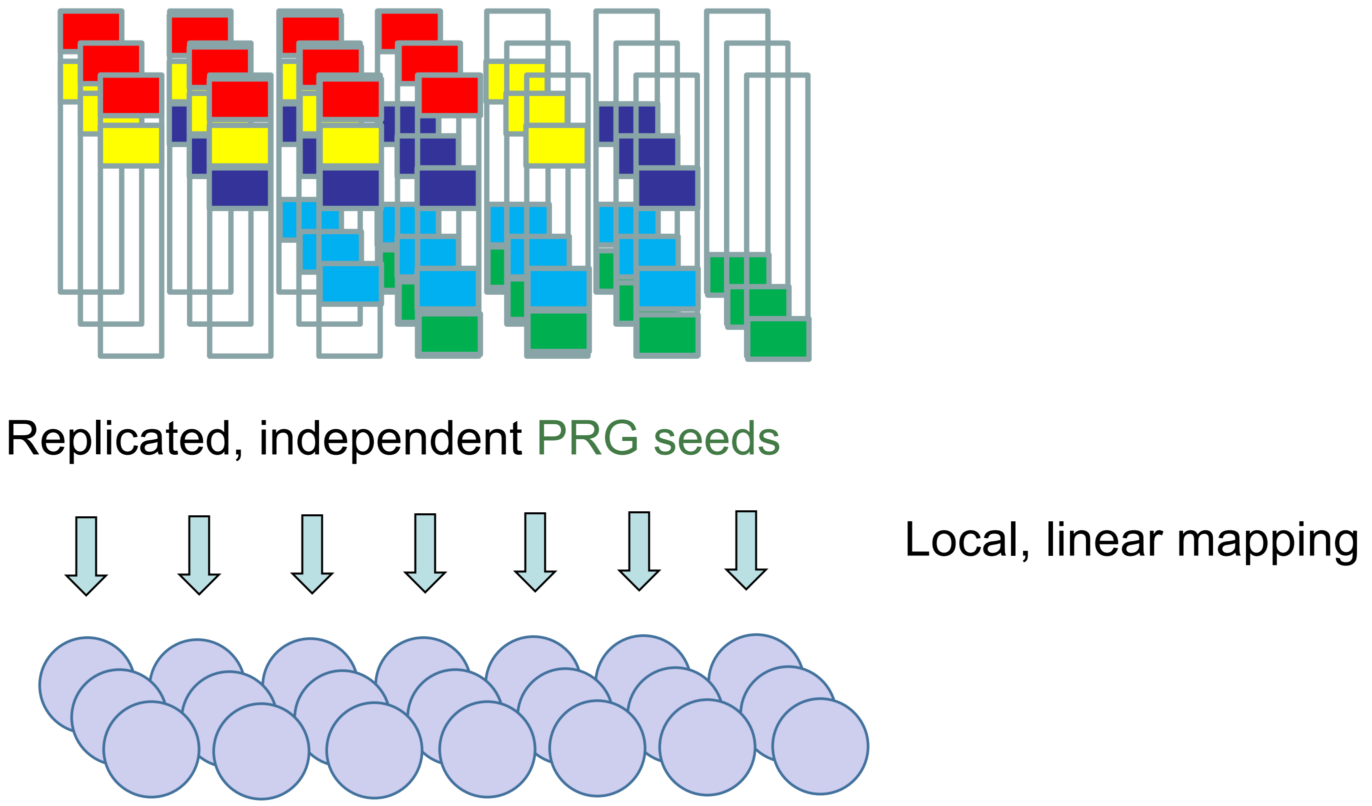
# Linear Multiparty Correlations: Pseudorandom Secret Sharing (PRSS)

[Gilboa-I 99, Cramer-Damgård-I 05]

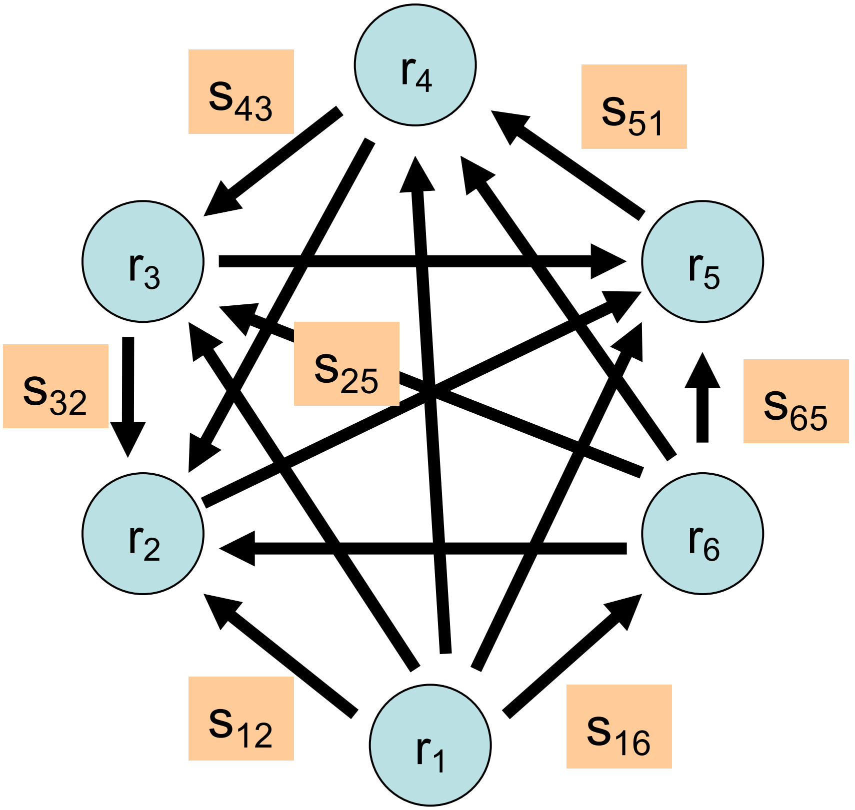


# Linear Multiparty Correlations: Pseudorandom Secret Sharing (PRSS)

[Gilboa-I 99, Cramer-Damgård-I 05]

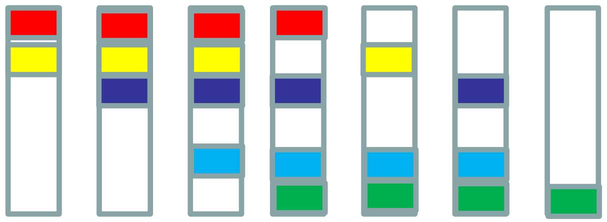


# Additive Shares of 0

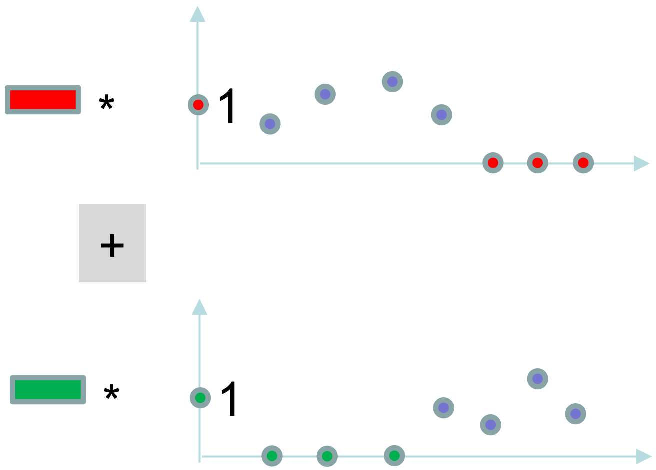



$r_i = \Sigma\ inbox_i - \Sigma\ outbox_i$

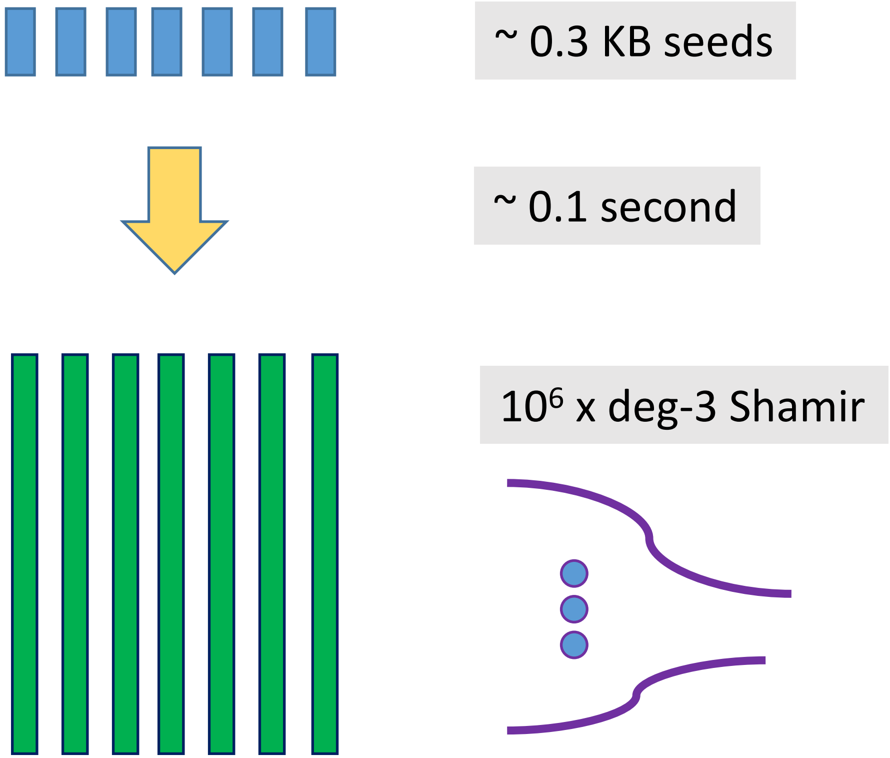
# Degree-d Shamir Shares



$\binom{n}{d}$ replicated elements
each given to n-d parties



# Concrete efficiency: n=7, d=3, $N=10^6$



# Generalized PRSS from Covering Designs

[Benhamooda-Boyle-Gilboa-Halevi-I-Nof 21]

- Goal: avoid $\binom{n}{d}$ overhead when security threshold t < degree d
  - O(n) share size for constant t regardless of degree
  - Application: Efficient MPC with share packing
- Construction from covering designs
  - (n, m, t)-cover: m-subsets of [n] covering all t-subsets
  - (n, d+1, t)-cover of size k ➔ PRSS with k(n-d)(d+1) storage
  - Tight up to a (d+1) factor

# Generalized PRSS from Covering Designs

[Benhamooda-Boyle-Gilboa-Halevi-I-Nof 21]

| $(n, m, t)$ | Baseline cover size | Best known cover size | Lower bound cover size | CDI seeds per party | PRSS seeds per party |
|---|---|---|---|---|---|
| $(9, 3, 1)$ | 3 | 3 | 3 | 8 | 7 |
| $(15, 5, 1)$ | 3 | 3 | 3 | 14 | 11 |
| $(15, 5, 2)$ | 49 | 13 | 13 | 91 | 48 |
| $(48, 16, 1)$ | 3 | 3 | 3 | 47 | 33 |
| $(48, 16, 2)$ | 15 | 13 | 13 | 1081 | 143 |
| $(48, 16, 4)$ | 495 | 252 | 173 | 178365 | 2772 |
| $(48, 20, 4)$ | 490 | 87 | 60 | 178365 | 1052 |
| $(48, 20, 6)$ | 5168 | 1280 | 459 | $1.07 \cdot 10^6$ | 15467 |
| $(49, 24, 2)$ | 31 | 7 | 7 | 1128 | 90 |
| $(49, 24, 4)$ | 245 | 38 | 31 | 194580 | 484 |
| $(49, 24, 8)$ | 12219 | 4498 | 968 | $3.7 \cdot 10^8$ | 57281 |
| $(72, 24, 2)$ | 15 | 12 | 12 | 2485 | 196 |
| $(72, 24, 4)$ | 495 | 180 | 126 | 971635 | 2940 |
| $(72, 24, 6)$ | 18564 | 4998 | 1419 | $1.4 \cdot 10^8$ | 81634 |

# 2-Party PCG in Minicrypt: Truth-Table Correlation

[BCGIKS19]

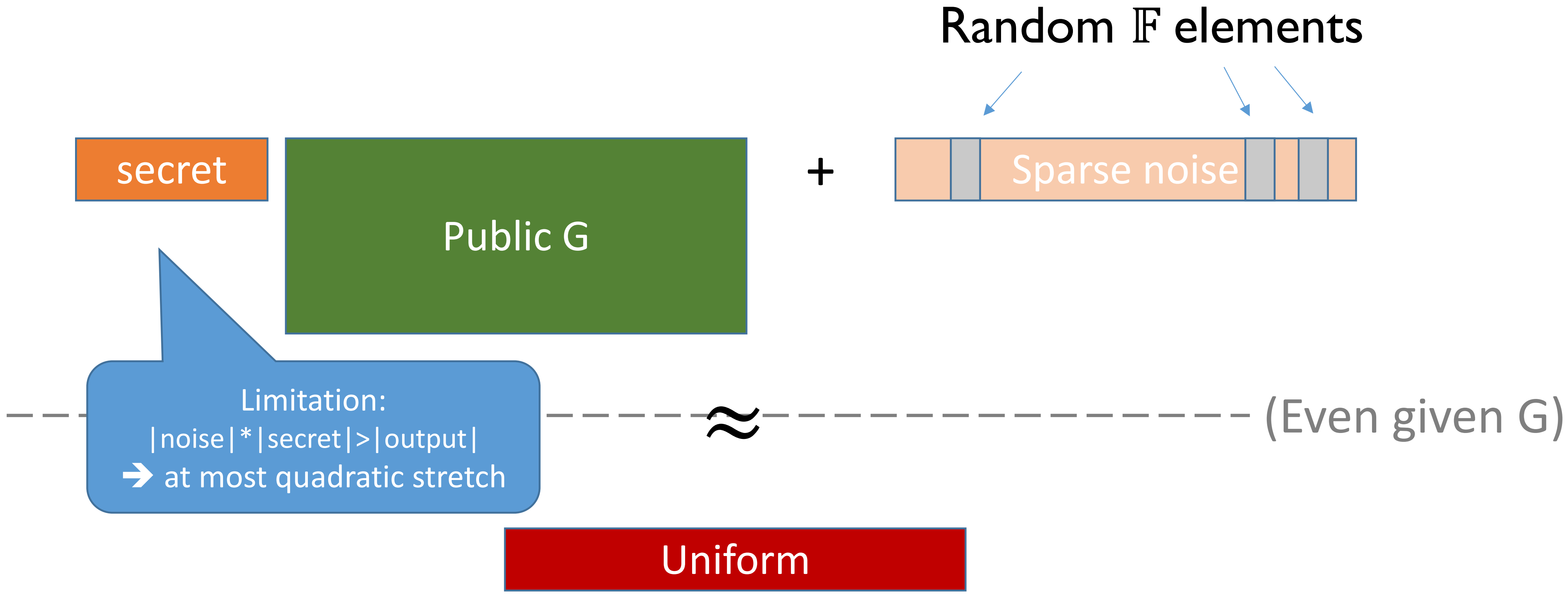
- Truth-table correlation for $g$: additive sharing of $\left(\mathrm{TT}_g \ll \mathrm{r}, \mathrm{r}\right)$
  - Authenticate via a random multiplier for malicious security
- Recall: DPF = FSS for a point function $f_{a,b}\colon [N] \to \mathbb{G}$
  - $a = r, b = 1,$ give PCG for additive shares of random unit vector $e_r$
  - Convert to TT correlation via matrix-vector multiplication
    - Matrix is circulant ➔ (offline) Expand time = $\tilde{O}(N)$
    - Alternatively: *locally* expand online in time $O(N)$
    - Authentication almost for free
- Comparison with "FSS gates" [BGI19, BCGGIKR21] (Elette's talk)
  - Works for *every* gate $g$
  - Infeasible for large input domains

# Part II:

# PCGs in Lapland

# Learning Parity with Noise (LPN) over $\mathbb{F}$ [BFKL93]

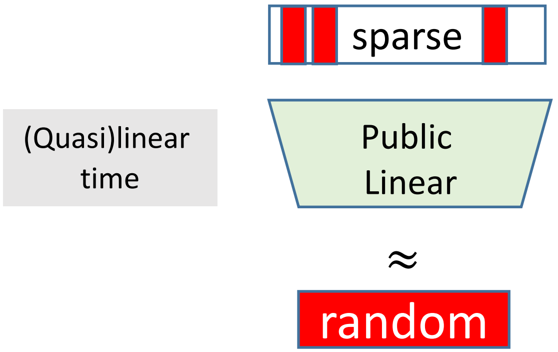
(LWE with **low-Hamming** noise)




Parameterized by **G** & by **noise distribution**

# LPN-based PCGs: Tools

**(Dual) LPN**




Also over large fields / rings

**Compressed secret-sharing of (N,w) sparse vector**


Distributed Point Function
Function Secret Sharing
[GI14,BGI15,BGI16]

OLE, Triples
Truth-table, PCF

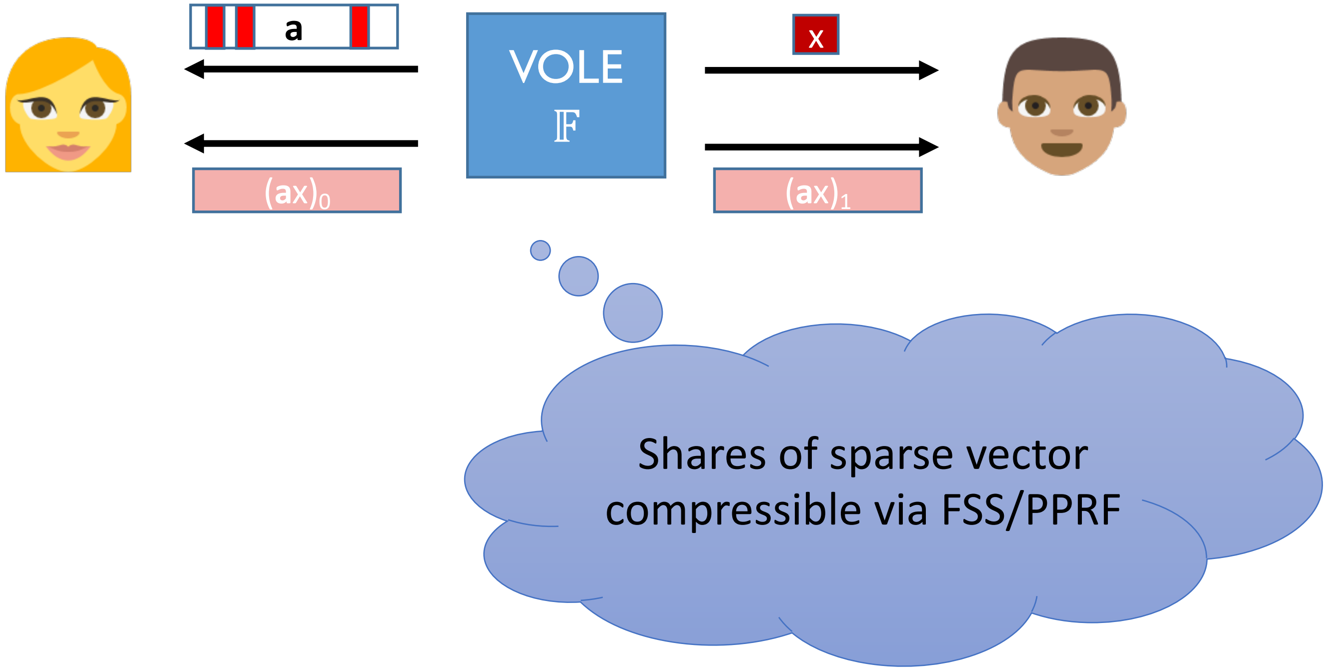
Puncturable PRF
[KPTZ13,BW13,BGI14]

VOLE, OT

w·log(N) PRG seeds
O(N) x PRG calls expansion

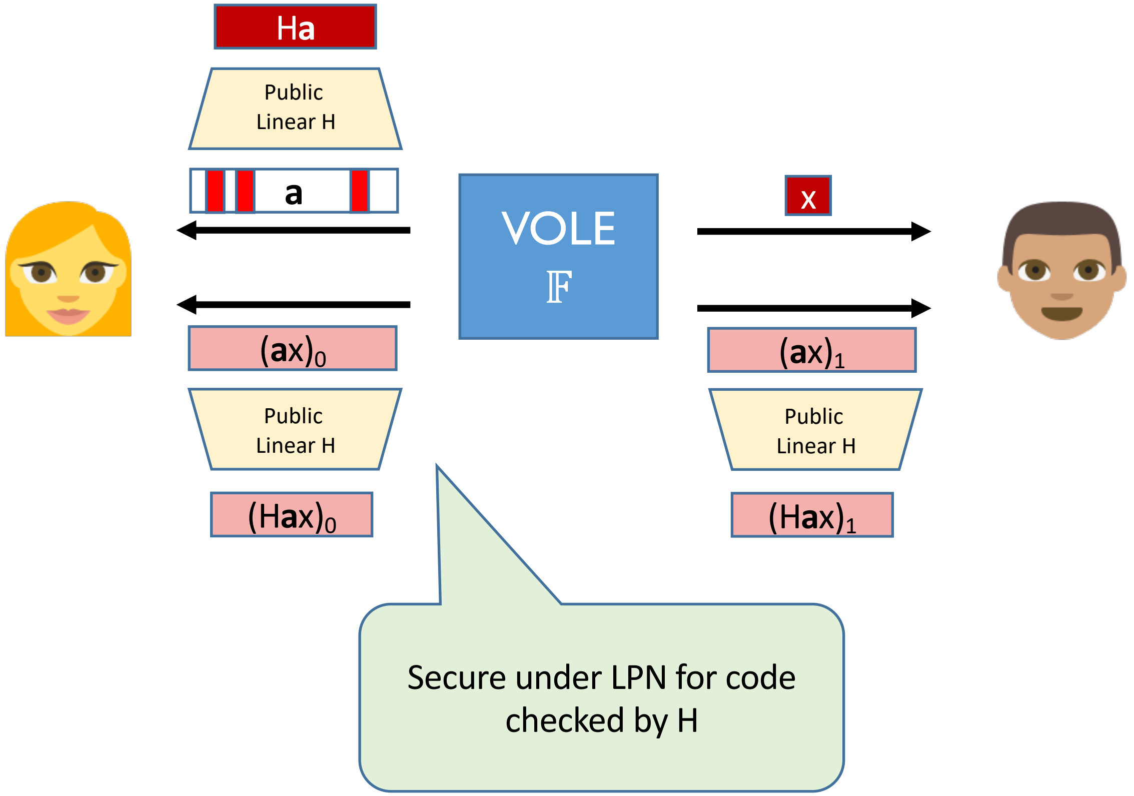
# Recall: **VOLE** correlation

# Idea: s p a r s e **VOLE** is compressible!



# PCG for VOLE from LPN

[Boyle-Couteau-Gilboa-I18]



# PCG for VOLE → PCG for OT

[Boyle-Couteau-Gilboa-I-Kohl-Scholl19, +Rindal19]

- Use VOLE over $\mathbb{F}_{2^\lambda}$ ($\lambda = 128$ in practice)
  - VOLE sender = OT receiver, **b** = sender's share of **ax**
- Pick entries of **a** from base field, x and **b** from extension field
- Each bit $a_i$ selects between $b_i$ (known) and $x+b_i$ (unknown)
  - For each received $c_i=a_ix+b_i$, VOLE sender knows one of $(c_i,c_i+x)$
  - Destroy correlations between unknown strings via hash function, a-la [IKNP03]

"Silent OT Extension"

# PCG for degree-d correlations from LPN

Goal: generate [p(r)] for degree-d polynomial map p

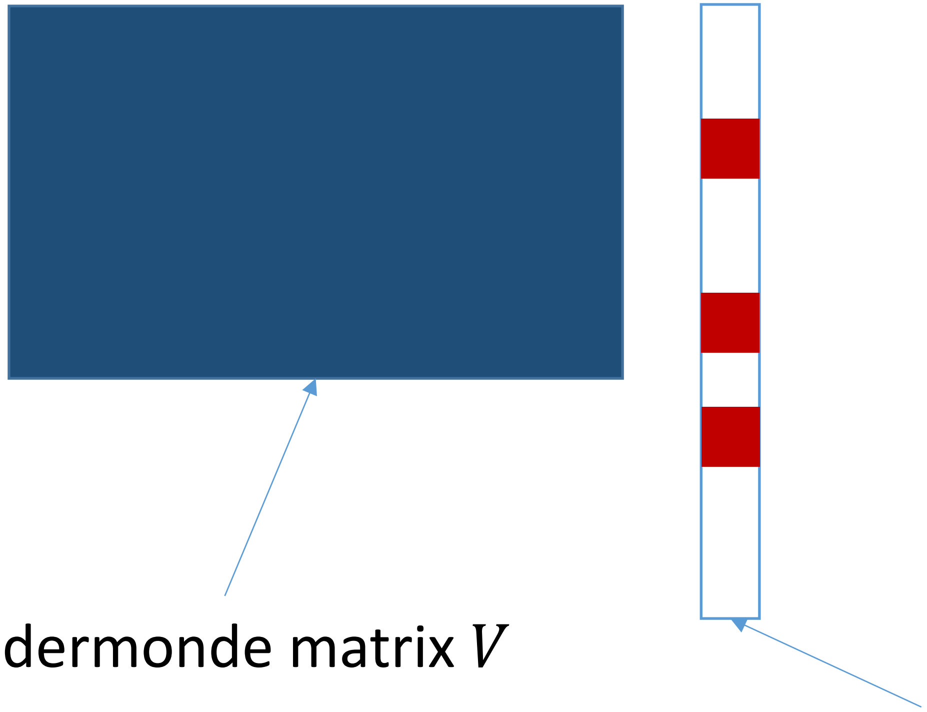
- Pick a random sparse **a**
- **Gen:** Use FSS to additively share **a**, **axa**, **axaxa**, … , **(a)**$^d$
- **Expand:** Write **p(Ha)** as a linear function L of shared values, and apply L to shares

Problem: poor concrete efficiency

- Even for OLE or triples, and with circulant H, takes $\Omega(N^2)$ computation

# Towards PCGs for triples

- **Idea:** Use evaluations of *sparse polynomials* $s, s'$ *and* $s \cdot s'$




**Good news:**

$s(\alpha_i) \cdot s'(\alpha_i) = (s \cdot s')(\alpha_i)$

Expand requires time $\tilde{O}(N)$

**Bad news:**

LPN broken by algebraic decoding techniques

# Arithmetic ring-LPN assumption

- **Idea:** Defeat algebraic decoding attacks by *building on ring-LPN*

**Ring-LPN assumption:** $R_p = \mathbb{Z}_p[X]/F(X)$:

$$(a, a \cdot e + f) \approx (a, \$)$$

$a \leftarrow R_p, \quad e, f$ $t$-sparse in $R_p$

$F(X)$ *splits into linear factors* $\Rightarrow R_p \cong \mathbb{Z}_p^N$

**Splittable ring-LPN:**

- Slightly better known attacks
- Requires slightly more noise

# PCG for triples from Ring-LPN

$$(a \cdot e + f) \cdot (a \cdot e' + f')$$
$$= a^2 \cdot ee' + a \cdot (ef' + fe') + ff'$$

- Share $ee', ef', fe', ff'$ via FSS
- Expand via polynomial multiplication + multi-evaluation

$\Rightarrow$ time $\tilde{O}(N)$

Security based on (splittable) ring-LPN

# Cost analysis and extensions

- **Cost:** for $N$ triples over $\mathbb{Z}_p$
  - $O(t^2)$ DPF keys
  - $O(Nt^2)$ PRG calls + $O(N \log N)$ arithmetic operations

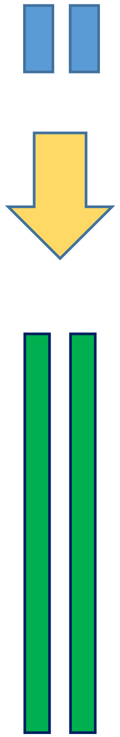
$O(Nt)$ using regular noise

- **Extensions:**
  - Extends to authenticated multiplication triples with < 2x overhead
  - Matrix triples, degree-2 correlations (less efficient)
  - Multi-party correlations (only non-authenticated)

# Multi-party multiplication triples

- Goal: PCG for *additive* n-out-of-n shares of *N* multiplication triples
  - Online communication scales linearly with n
- Idea: Use n(n-1) instances of 2-party PCG for triples
  - Separately share each term $a_i b_j$
  - Requires 2-party PCG to be programmable
  - Does not work with PCG for OT, or authenticated triples
- Workarounds for authenticated triples:
  - Use 3-party DPF [Abram-Scholl22] (less efficient)
  - Use (unauthenticated) multiplication triples + fully-linear IOP [Boyle-Gilboa-I-Nof21]

# Concrete efficiency: VOLE and OT



~ 10 KB seeds

~ 100 KB 2-round seed generation [BCGIKRS19]

~ 1 second

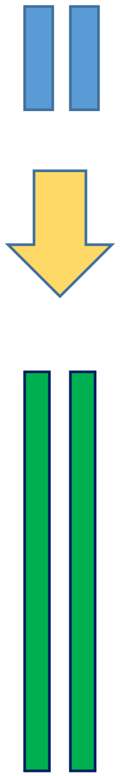
Using quasi-cyclic codes

Length-$10^6$ VOLE over 128-bit field

$10^6$ x 128-bit OT

~ 20x speedup via LPN-friendly LDPCs [Couteau-Rindal-Raghuraman21]

# Concrete efficiency: OLE and Triples



~ 1 MB seeds

~ 4 MB
seed generation
(bootstrapped)

~ 10 / 20 seconds

$10^6$ x 128-bit OLE /
Authenticated Triples

Non-silent alternatives:

Overdrive [KPR18]

Leviosa [HIVM19]

x100-x1000 communication

comparable run time

# Pseudorandom Correlation Functions (PCF)

[Boyle-Couteau-Gilboa-I-Kohl-Scholl20]

- Goal: securely generate correlation instances on the fly
  - Pair of correlated (weak) PRFs $(f_{k_0}(r), f_{k_1}(r))$
  - Security against insiders

- GGM-style reduction to PCG does not apply…

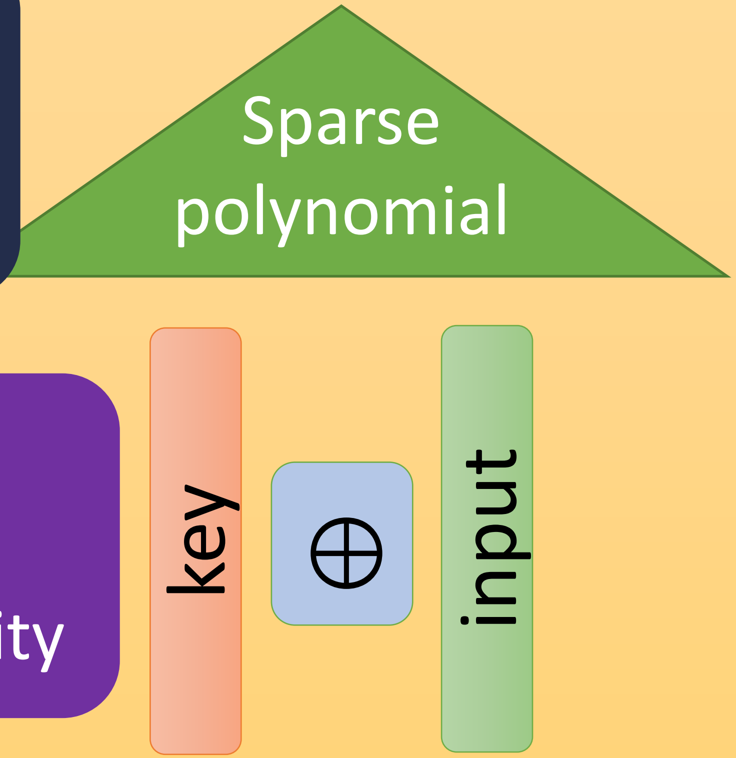
- PCF for VOLE from WPRF $f_k$ and FSS:
  - Pick random key $k$ and scalar $x$
  - Give $k$ to $P_0$, $x$ to $P_1$
  - Use FSS to share $x \cdot f_k$
  - Challenge: use PRG-based FSS!

# MPC-friendly WPRF Candidate

Best possible security: $2^{\sqrt{n}}$
[Hellerstein-Servedio07]

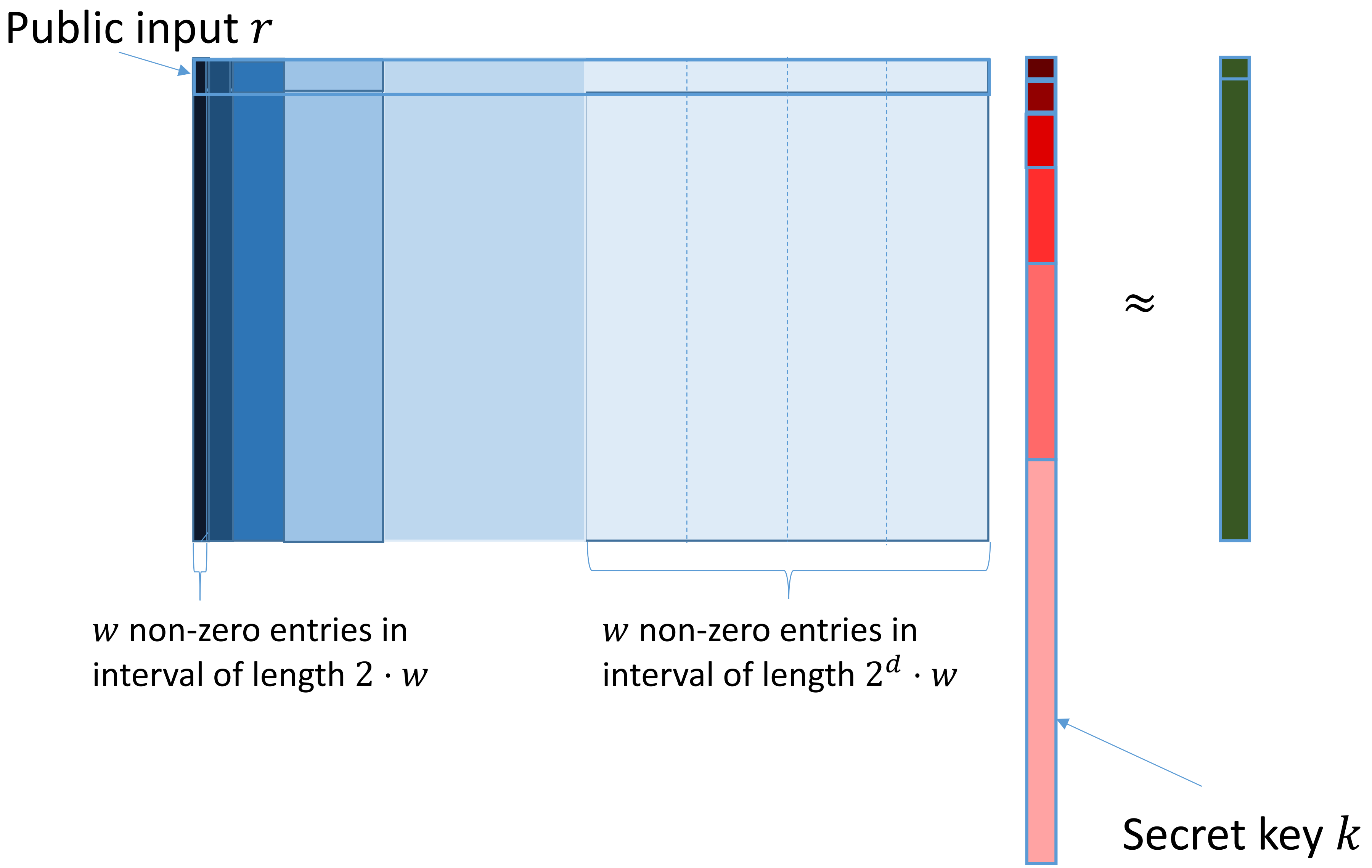
$$f_k(x) = \bigoplus_{i=1}^{D} \bigoplus_{j=1}^{w} \bigwedge_{h=1}^{i} (x_{i,j,h} \oplus k_{i,j,h})$$

Secure under variable-density variant of LPN




Applications:
- PCF
- XOR-RKA security

# Variable-density LPN



# Concrete efficiency: PCF

- PCFs for OT / VOLE from VDLPN ($< 10^9$ instances) [BCGIKS20]
  - key size: $\approx 120\text{kB}$ ($\approx 2\text{MB}$ conservative)
  - evaluation: $8{,}000$ PRG calls / instance => ≈ 20,000 instances / second / core

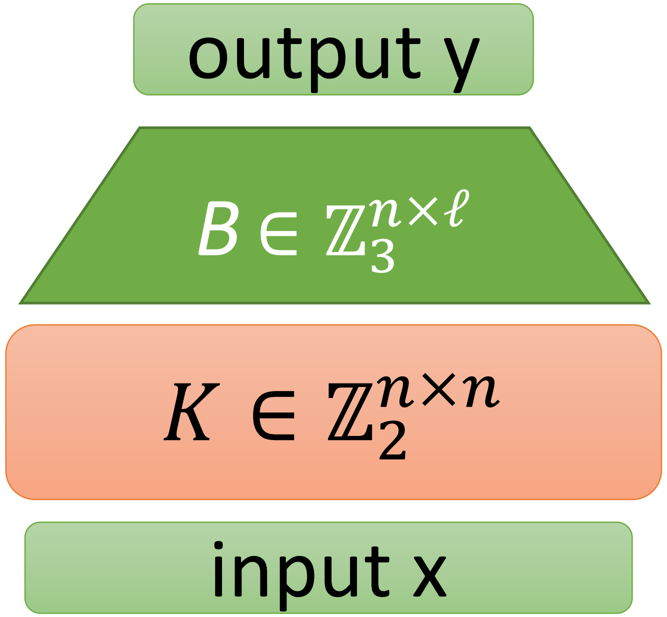
- PCFs from number-theoretic assumptions [Orlandi-Scholl-Yakoubov21]
  - Public-key setup, small keys
  - Slow evaluation

Peter tomorrow

# Application: MPC-friendly symmetric crypto

"2-3-WPRF" candidate

[Boneh-I-Passelègue-Sahai-Wu18]




$n = 256, \ell = 81$

Secure protocol $[K],[x_i] \rightarrow [y_i]$

[Dinur-Goldfeder-Halevi-I-Kelkar-Sharma-Zaverucha 21]

With preprocessing:

Online cost 1024 bits, 2 rounds

Using PCGs for VOLE/OT, amortized preprocessing cost: 353 bits

Main trick: converting random OT over $\mathbb{Z}_3$ to "double-sharing" ($[r]_2, [r]_3$) deterministically conditioned on OT sender's inputs being distinct.

→ 1.5n OT instances produce n double-shares

→ 1.377n bits to communicate good subset

# Remaining challenges

## Better PCGs

- More correlations?
  - Garbled circuits, FSS keys, …
- Multi-party binary or authenticated triples
- Smaller seeds, faster expansion and seed generation
- Scalable PCG for Shamir-shares

## Better understanding of LPN-style assumptions

- Which codes?
- Which noise patterns?

## Better PCFs

# The End

- Questions?