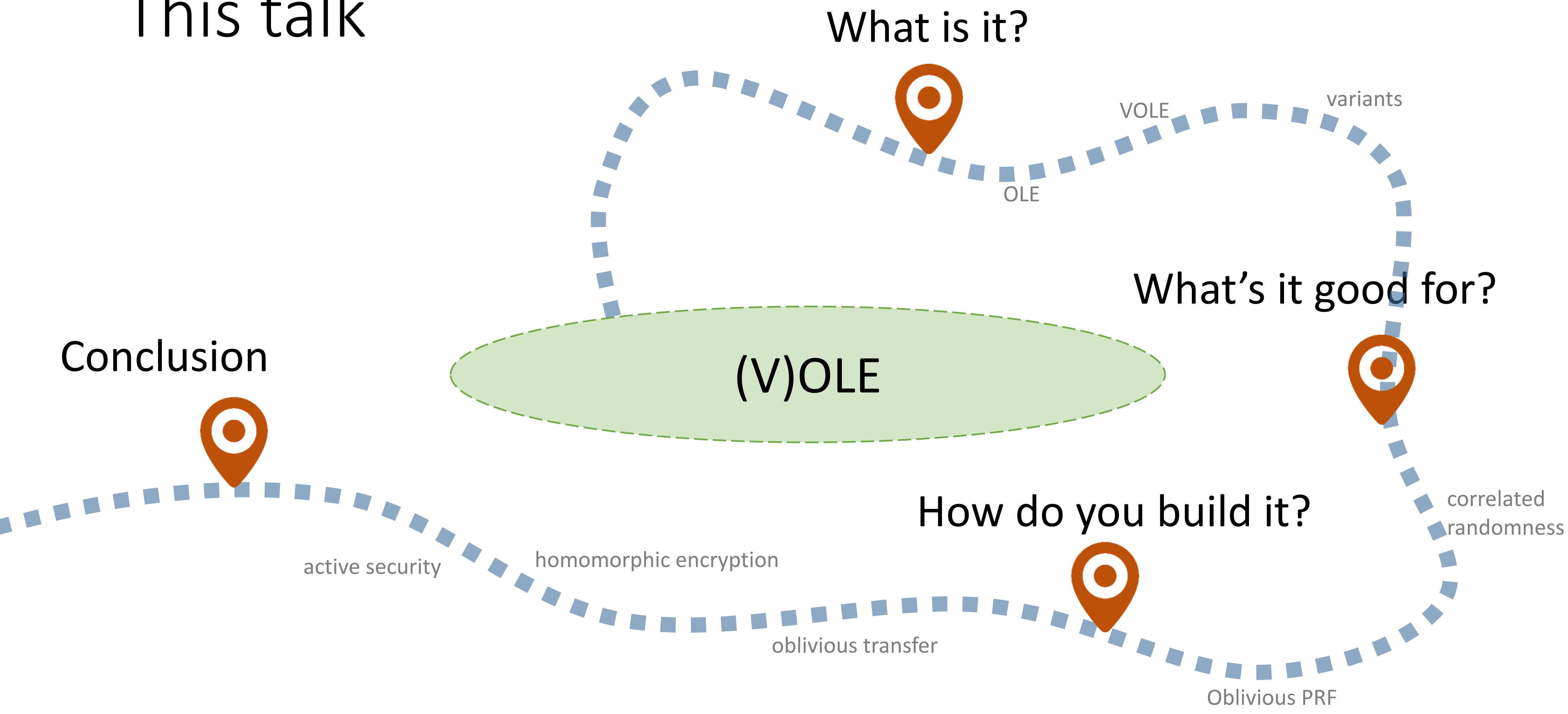
(Vector) Oblivious Linear Evaluation: Basic Constructions and Applications

Peter Scholl

24 January 2022, Bar-Ilan Winter School

AARHUS UNIVERSITY

This talk


What is it?
VOLE
variants
OLE
What's it good for?
(V)OLE
Conclusion
correlated randomness
How do you build it?
active security
homomorphic encryption
oblivious transfer
Oblivious PRF

Oblivious linear evaluation (OLE)

Input: $x \in \mathbb{Z}_p$

Input: $a, b \in \mathbb{Z}_p$

⋮

Output: $y = ax + b$

$x \in \mathbb{Z}_p$

$a, b \in \mathbb{Z}_p$

OLE functionality

$y = ax + b$

OLE is secret-shared multiplication

Input: $x \in \mathbb{Z}_p$

Input:
$a \in \mathbb{Z}_p$
$b \leftarrow \mathbb{Z}_p$

$$\boxed{y - b = ax}$$

Variants: random-OLE, vector-OLE

$x \in \mathbb{Z}_p$ → OLE ← $a, b \in \mathbb{Z}_p$

$y = ax + b$

$x \leftarrow \mathbb{Z}_p$, $y = ax + b$ ← $-OLE → $a, b \leftarrow \mathbb{Z}_p$

$x \in \mathbb{Z}_p$ → VOLE ← $\vec{a}, \vec{b} \in \mathbb{Z}_p^n$

$\vec{y} = \vec{a}x + \vec{b}$

A few basic observations

$n \times$ OLE $\Rightarrow$ $\cancel{\Leftarrow}$ $1\times$ VOLE (unconditional, passive security)

- VOLE is easier to build than $n \times$ OLE

\$-OLE $\Rightarrow$ OLE (unconditional, send 3 $\mathbb{Z}_p$ elem.)

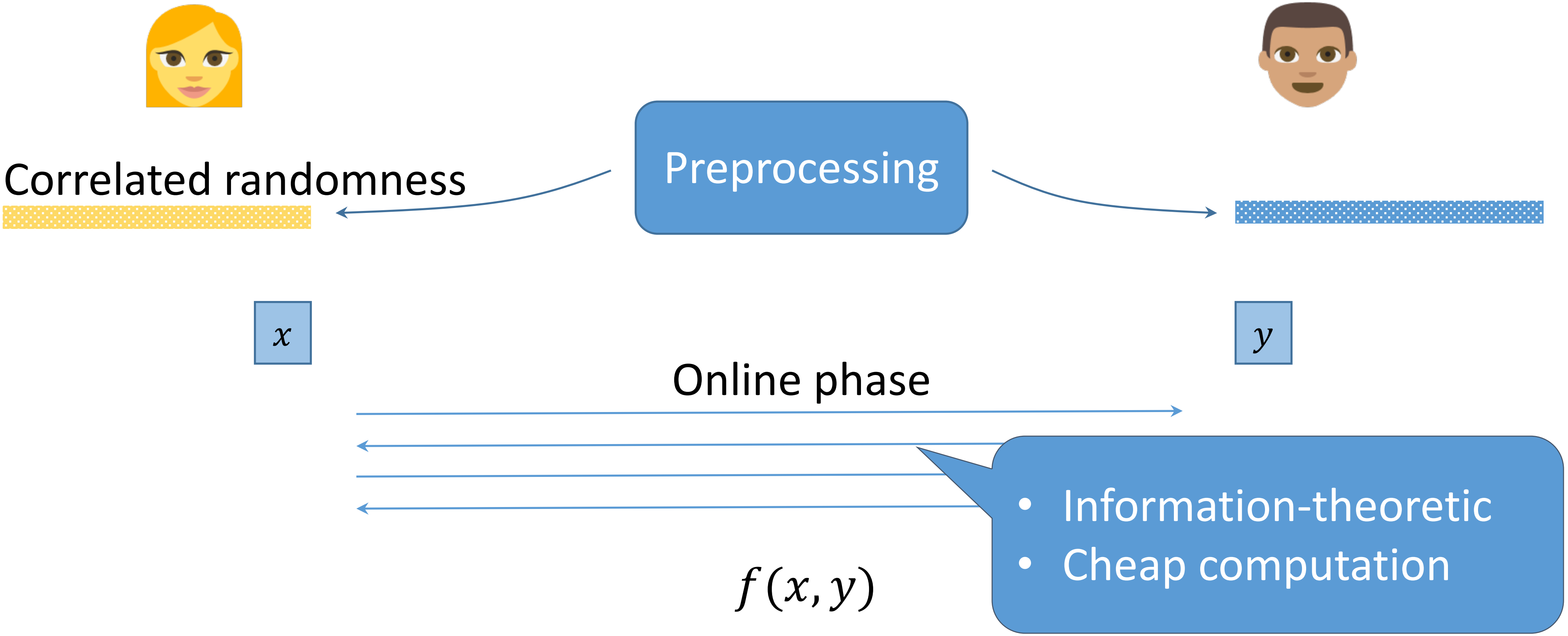
- \$-(V)OLE is enough

OLE $\Rightarrow$ Oblivious Transfer (unconditional)

- Public-key crypto is necessary [IR 89]

Motivation: Secure Computation with Preprocessing

[Beaver '91]



Example: multiplication triples from OLE

x, x', y, y' ← 2x \$-OLE → a, a', b, b'

$$y - b = \boxed{ax}$$
$$y' - b' = \boxed{a'x'}$$

$$(\boxed{x + a'}) \cdot (\boxed{x' + a}) = \boxed{xx'} + \boxed{aa'} + \boxed{ax} + \boxed{a'x'}$$

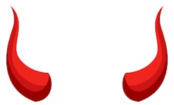
$$\boxed{u} \cdot \boxed{v} = \boxed{w}$$

(V)OLE for correlated randomness

- Scalar/vector triples, matrix triples
  - Build from VOLE
- Multi-party correlations:
  - From pairwise instances of (V)OLE
  - Other approaches: depth-1 homomorphic encryption [DPSZ 12]
- Authenticated secret shares:
  - Use VOLE to generate information-theoretic MACs
  - Key part of SPDZ protocols [DPSZ 12, KOS 16, KPR 18, …]

Application: Oblivious Pseudorandom Functions

PRF F



Guess b

x

y_b

$b \leftarrow \{0,1\}$
$K \leftarrow \{0,1\}^\lambda$

$y_0 = F(K, x)$
$y_1 = \$(x)$

Oblivious PRF

K

x

$\vdots$

$F(K, x)$

$F(K, y)$ remains pseudorandom for any $y \neq x$

Vector-OLE ⇒ Batch OPRF evaluation [BCGIKS 19]

$s \leftarrow \mathbb{F}_q$ → VOLE ← $a_i \in \mathbb{F}_q$

$t_i = a_i s + b_i$ ← VOLE ← $b_i \leftarrow \mathbb{F}_q$

Keys $K_i := (s, t_i)_i$

$$\boxed{F(K_i, a_i) \coloneqq H(t_i - a_i s)}$$

Output $H(b_i)$

- ❖ Relaxed OPRF: related keys, leakage
- ❖ Secure if H is a random oracle
  - Or variant of correlation-robustness

Random Vector-OLE ⇒ Batch OPRF evaluation

$s \leftarrow \mathbb{F}_q$

$t_i{}' = r_i s \ + b_i$

$-VOLE

$r_i \leftarrow \mathbb{F}_q$

$b_i \leftarrow \mathbb{F}_q$

$d_i = a_i - r_i$

$t_i = t_i' + d_i s$

Keys $K_i := (s, t_i)_i$

Output $H(b_i)$

- Optimal communication: 1 $\mathbb{F}_q$ element
  - (given $-VOLE)

Applications of OPRF

- Random 1-out-of-q OT
  - Correlated randomness, e.g. masked truth tables [DKSSZZ 17]
- Password-authenticated key exchange, e.g. OPAQUE [JKX 18]
  - Batch OPRF seems less useful
- Private set intersection
  - Reducing use of public-key crypto [KKRT 16, KMPRT 17, …]
  - With polynomial-based encoding [GPRTY 21, Sec 7.1]
    - Simple protocol, communication: |input|

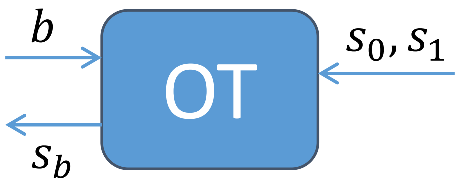
Constructing VOLE, “non-silently”

Taxonomy of VOLE protocols

Oblivious Transfer

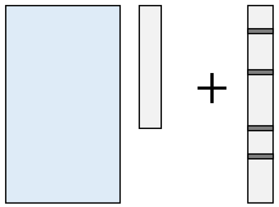
Homomorphic Encryption

"Non-silent"



"Silent"

- Mostly based on LPN
- Require "seed" VOLEs to bootstrap



(V)OLE from Oblivious Transfer [Gilboa 99]

$x \in \mathbb{Z}_q$

$a, b \in \mathbb{Z}_q$

Bit-decompose $x = \sum_{i=1}^{m} 2^{i-1} x_i$

Sample $b_i \in \mathbb{Z}_q$ s.t.
$b = \sum_i 2^{i-1} b_i \mod q$

x_1 → OT ← $b_i, b_i + a$

y_1

⋮

x_m → OT ← $b_m, b_m + a$

y_m

Output $y = \sum_i 2^{i-1} y_i$

$y_i = b_i + a x_i$
$\Rightarrow y = b + ax$

Repeat for VOLE
[KOS 16]

AARHUS UNIVERSITY

(V)OLE from Oblivious Transfer [Gilboa 99]

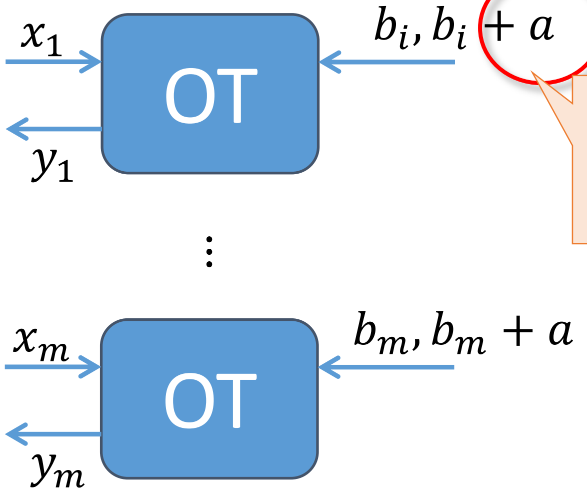
- Perfectly secure
- Each output: $m = \log q$ calls to OT on m-bit strings
  - Computational cost: cheap via OT extension [IKNP 03]
  - Communication: $\geq m^2$ bits
- Active security?

(V)OLE from Oblivious Transfer: active security?

$x \in \mathbb{Z}_q$

$a, b \in \mathbb{Z}_q$

Bit-decompose $x = \sum_i 2^{i-1} x_i$



Sample $b_i \in \mathbb{Z}_q$ s.t.

mod q

Bob uses $a' \neq a$:
Output becomes $y + (a' - a)x_1$

Output $y = \sum_i 2^{i-1} y_i$

VOLE: lightweight correctness check

x, y_i

a_i, b_i

Goal: check that $y_i = a_i x + b_i$, for all i

Random challenges

$$\xrightarrow{\chi_1, \dots, \chi_n \in \mathbb{Z}_q}$$

$$a^* = \sum_i \chi_i a_i + a_{n+1}, \quad b^* = \sum_i \chi_i b_i + b_{n+1}$$

$$\xleftarrow{a^*, b^*}$$

$y^* = \sum \chi_i y_i + y_{n+1}$

Check $y^* = a^* x + b^*$

Intuitio[illegible]

- To pa[illegible]heck when y_i is incorrect, Bob must guess χ_i
- Succ[illegible]with pr. $1/p$

Problems with selective failure

- Recall: corrupt Bob can induce error:

$$y' = y + (a' - a)x_1$$

  - Error depends on secret bit x_1!
  - Even if VOLE is correct, leaks that $x_1 = 0$

- Solutions:
  - 1) Relaxed VOLE: allow small leakage on x [KOS 16], [WYKW 21]
  - 2) Privacy amplification via leftover hash lemma [KOS 16]

(V)OLE from OT: Summary

- Simple protocol with lightweight computation
  - Leveraging fast OT extension techniques
- Expensive communication
  - At least m^2 bits, where $m = \log q$
- Active security almost for free
  - If leakage on x is OK

VOLE from Homomorphic Encryption

Linearly homomorphic encryption

- PKE scheme $(KeyGen, Enc, Dec)$, encrypts vectors over $\mathbb{Z}_p$

For $\vec{a} \in \mathbb{Z}_p^n$, write $[\vec{a}] := \mathrm{Enc}_{\mathrm{pk}}(\vec{a})$

- Linear homomorphism:
  - Can compute $[\vec{a}] + [\vec{b}]$ or $\vec{c} \cdot [\vec{a}]$, for $\vec{c} \in \mathbb{Z}_p^n$, s.t.

$$\mathrm{Dec}\big([\vec{a}] + [\vec{b}]\big) = \vec{a} + \vec{b}$$
$$\mathrm{Dec}(\vec{c} \cdot [\vec{a}]) = \vec{c} \cdot \vec{a}$$

Component-wise product

Examples of Linearly Homomorphic Encryption

More on Wednesday!

- Paillier encryption
  - Each ciphertext encrypts a $\mathbb{Z}_N$ element $(N = pq)$

- DDH
  - ElGamal in the exponent: poly-size plaintexts in $\mathbb{Z}$
  - Class groups: $\mathbb{Z}_p$ for large prime p [CL 15]

- Ring Learning With Errors (RLWE) [LPR 10]
  - Natively encrypts a vector in $\mathbb{Z}_p^m$

Naïve VOLE from Linearly Homomorphic Encryption

$x \in \mathbb{Z}_p$

$\vec{a}, \vec{b} \in \mathbb{Z}_p^m$

$pk, sk \leftarrow Gen(1^\lambda)$ $\xrightarrow{pk, [x]}$

$\xleftarrow{[\vec{y}] = \vec{a} \cdot [x] + [\vec{b}]}$

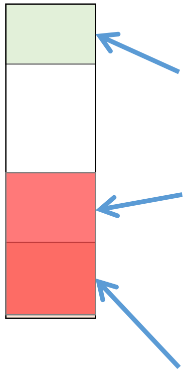
$\vec{y} = Dec_{sk}([\vec{y}])$

Security:

- Alice: CPA security
- Bob: circuit privacy

Circuit privacy in homomorphic encryption

❖ In RLWE, message hidden by “noise”:



❖ After computing $\vec{a} \cdot [x] + [\vec{b}]$:

- ➢ Noise depends on $\vec{a}$ and $\vec{b}$

❖ Classic solution:

- ➢ “Noise flooding”
- ➢ Requires much larger ciphertexts

Optimization: ”Gentle noise flooding” [dCHIV 21]

- Encrypt t-out-of-n sharing of message
- A few leaked coordinates don’t matter

What about active security?

- What can go wrong?
  - Alice/Bob could send garbage ciphertexts…
- What about correctness check as in OT?
  - Selective failure is more subtle
  - Error may depend on ciphertext noise/secret key
- Solution: zero-knowledge proofs
  - Alice: proof of plaintext knowledge
  - Bob: proof of correct multiplication

ZK proofs for homomorphic encryption

- RLWE is more challenging than number-theoretic assumptions

- Proof of plaintext knowledge
  - Naïve sigma protocol: soundness ½
  - Various optimizations [BCS 19], amortization [BBG 19]
  - Still computationally expensive, often need larger parameters

- Proof of correct multiplication
  - Even worse! Tricky to amortize
  - Can be avoided, assuming linear-only encryption [BISW 18, KPR 18]

Conclusion: Basic constructions and applications

- OLE and VOLE are core building blocks of secure computation
  - Correlated randomness
  - Special-purpose applications like OPRF, private set intersection
  - Next talk: zero knowledge
- Non-silent protocols: OT, AHE
  - Important, even if silent protocols win ☺
  - Open question: improving RLWE parameters and efficiency
    - Especially for active security

AARHUS UNIVERSITY

Thank you!

?