When Bad News Become Good News

Towards Usable Instances of Learning with Physical Errors

Davide Bellizia[1] Clément Hoffmann[1] Dina Kamel[1] Pierrick Méaux[2] François-Xavier Standaert[1]

[1] UCLouvain, ICTEAM, Crypto Group, Louvain-la-Neuve, Belgium
firstname.lastname@uclouvain.be
[2] Luxembourg University, SnT, Luxembourg
pierrick.meaux@uni.lu

Monday 19th September, 2022

Learning problems

Learning problems have proven to be interesting computationally hard problems.

- **LPN** - **LWE** - **LWR** - **MLWE** **...**	$\Rightarrow$	- *One-way function* - *Secret-key encryption scheme* - *Post-quantum PKE* - *Identity-based encryption* - *Secure MPC* - *Indistinguishability obfuscation* *...*

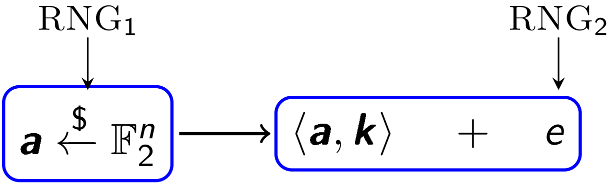
Learning problems - Presentation

$$\boldsymbol{k} \xleftarrow{\$} \mathbb{F}_2^n$$

$$\boxed{\boldsymbol{a} \xleftarrow{\$} \mathbb{F}_2^n} \longrightarrow \boxed{\langle \boldsymbol{a}, \boldsymbol{k} \rangle \quad + \quad e}$$

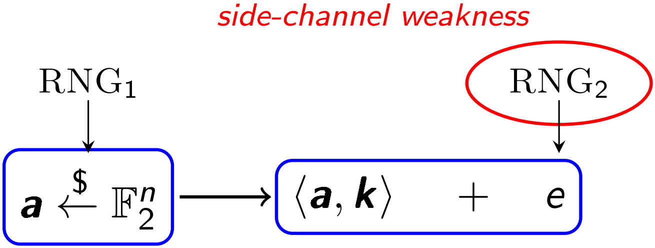
LPN samples

Learning problems - Implementation



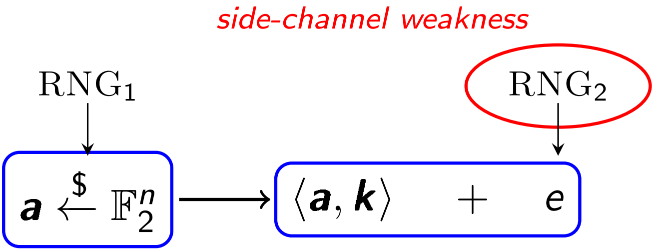
LPN samples

Learning problems - Implementation



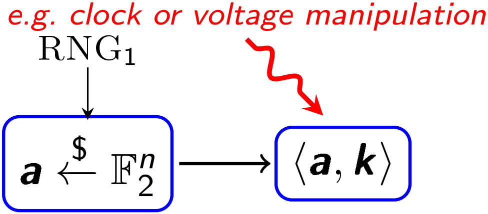
LPN samples

Physical learning problems - Presentation



LPN samples

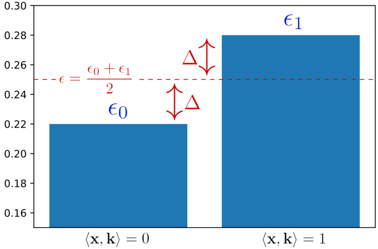
Physical learning problems - Presentation



LP**P**N samples

Physical learning problems - Data dependencies

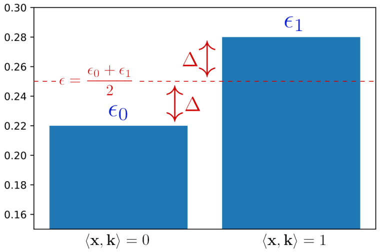
Output data dependencies:



Error probability depending on the correct output value

Physical learning problems - Data dependencies

Output data dependencies:

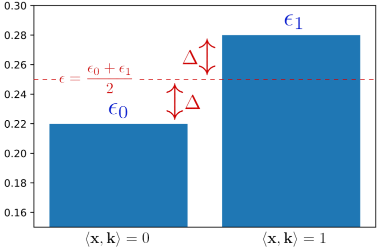


Error probability depending on the correct output value

- Not negligible
- Reduction for LPPN

Physical learning problems - Data dependencies

Output data dependencies:



Error probability depending on the correct output value

- Not negligible
- Reduction for LPPN

Input data dependencies:

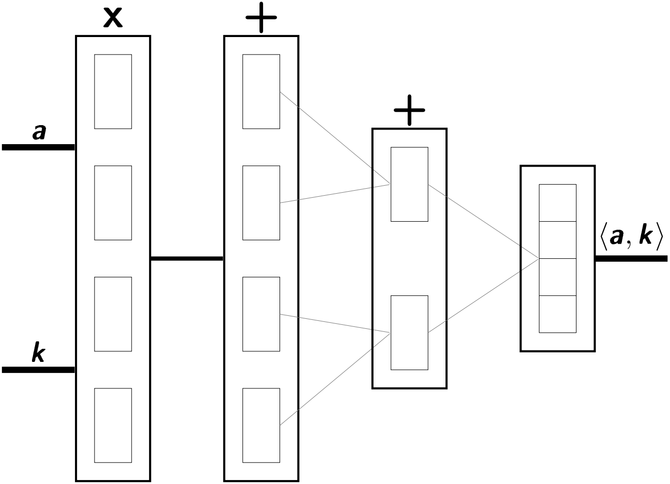
- Computationally hard to exploit
- Can be made small by design

Can this extend to LWE ?

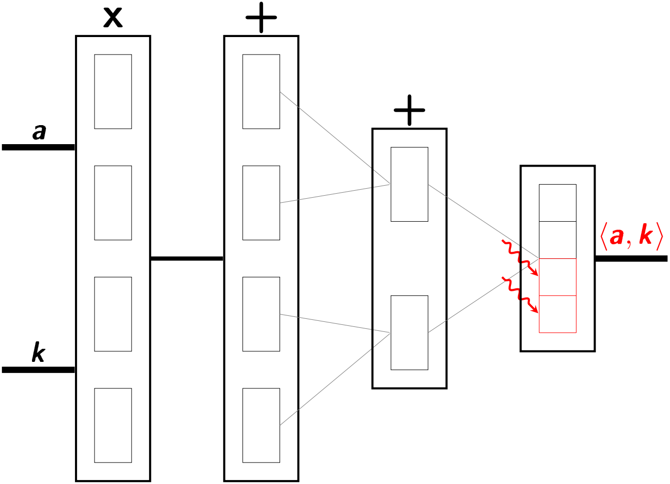
GOAL:

- From $\mathbf{GF_2}$ to **larger rings/fields**
- Error distribution approximating a $\text{CBD}_{2\text{ or }3}$ (used in Kyber)

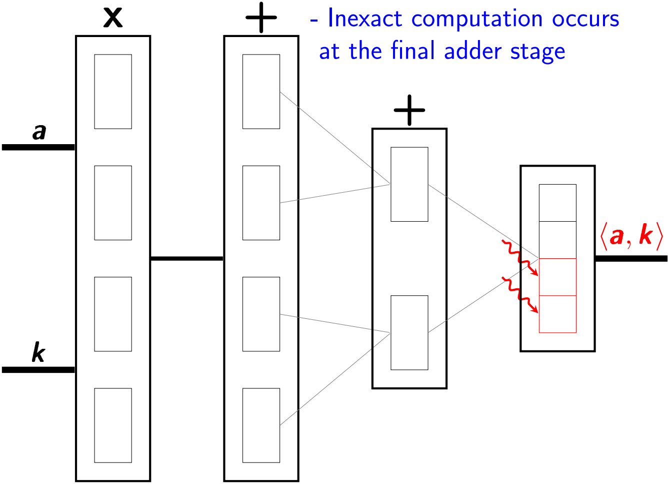
Inner product structure



Natural solution - Presentation

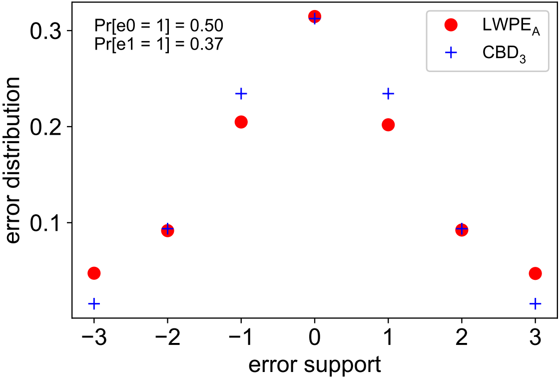


Natural solution - Presentation



- Inexact computation occurs at the final adder stage

Natural solution - Results



LWPE_A: Error distribution approximating CBD_3.

Bad news - Mathematical data dependencies (1/2)

TOY EXAMPLE:

- $\langle \boldsymbol{a}, \boldsymbol{k} \rangle = 0$
- modulo 4

Bad news - Mathematical data dependencies (1/2)

TOY EXAMPLE:

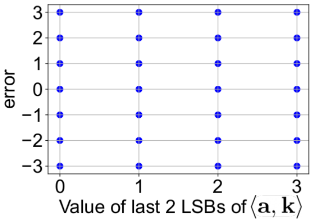
- $\langle \boldsymbol{a}, \boldsymbol{k} \rangle = 0$
- modulo 4

wrong support

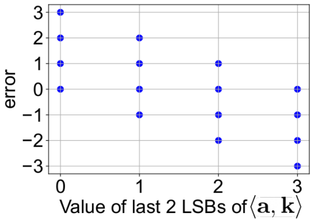
$$\langle \boldsymbol{a}, \boldsymbol{k} \rangle = \overline{00} \longrightarrow \begin{cases} \overline{00} \rightarrow e = 0 \\ \overline{01} \rightarrow e = 1 \\ \overline{10} \rightarrow e = 2 \\ \overline{11} \rightarrow e = 3 \end{cases}$$

Bad news - Mathematical data dependencies

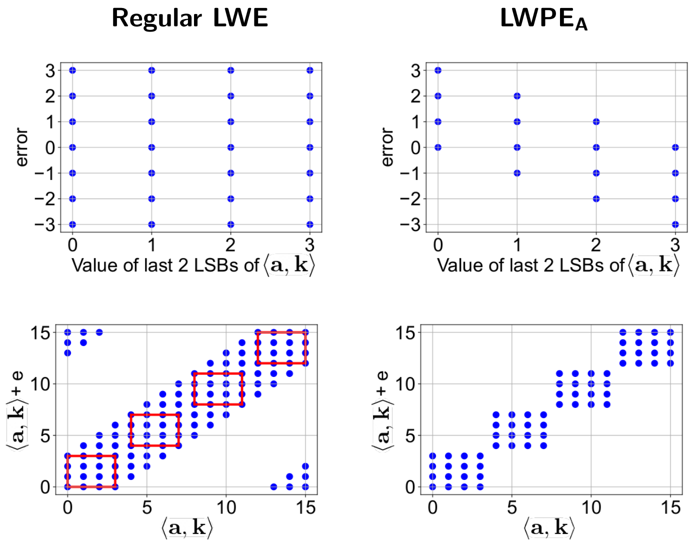
Regular LWE



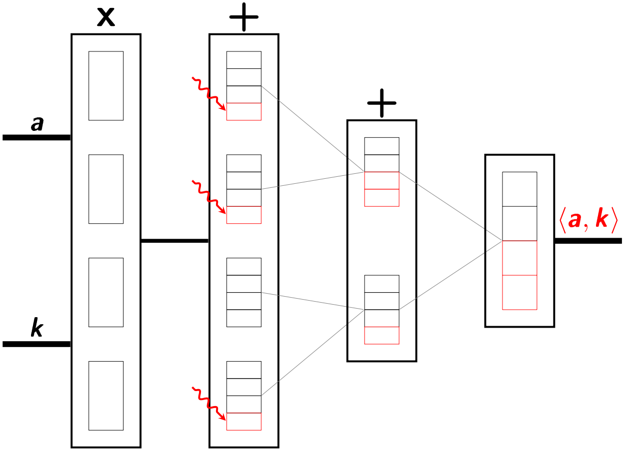
$\mathsf{LWPE_A}$



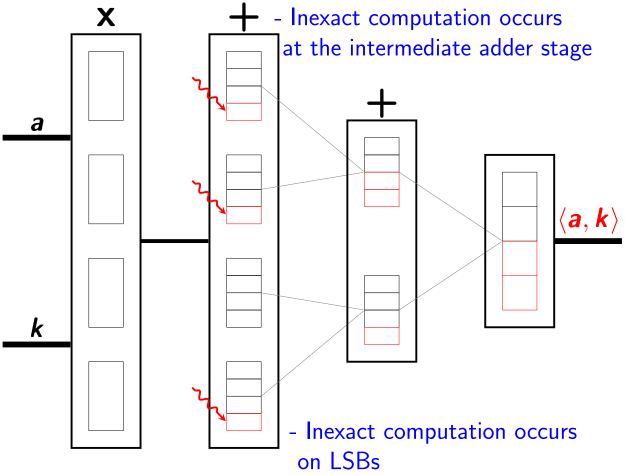
Bad news - Mathematical data dependencies



$\mathrm{LWPE_B}$

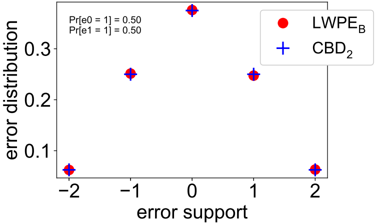


LWPE$_B$

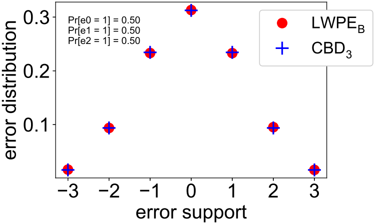


x
+
- Inexact computation occurs
at the intermediate adder stage
a
k
+
⟨a, k⟩
- Inexact computation occurs
on LSBs

Results



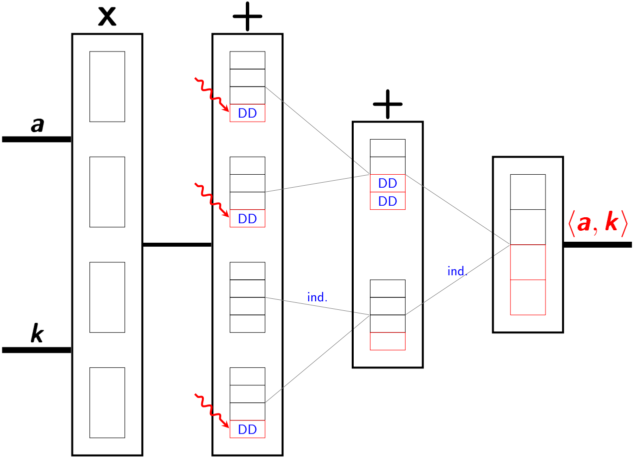
CBD_2



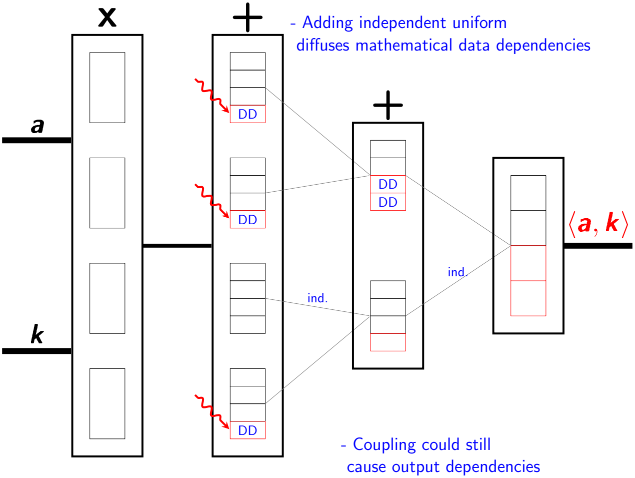
CBD_3

$LWPE_B$: Error distribution

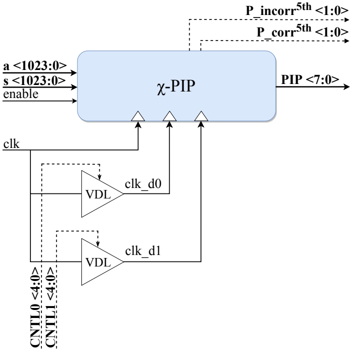
Good news - physical data dependencies



Good news - physical data dependencies



FPGA prototype and data dependencies



LWPE FPGA prototype (dashed lines are only for configuration and testing).

Empirical verification that **data dependencies** cannot be observed

Conclusion - What did we gain?

Interesting potential against leakage

Conclusion - What did we gain?

Interesting potential against leakage

- **Linear** overhead in the shares number

Conclusion - What did we gain?

Interesting potential against leakage

- **Linear** overhead in the shares number
- Trivial composition (key-homomorphic)

$$\langle \boldsymbol{a}, \boldsymbol{k} \rangle = \langle \boldsymbol{a}, \boldsymbol{k}_0 \rangle + \langle \boldsymbol{a}, \boldsymbol{k}_1 \rangle + \cdots + \langle \boldsymbol{a}, \boldsymbol{k}_{\boldsymbol{d}-1} \rangle$$

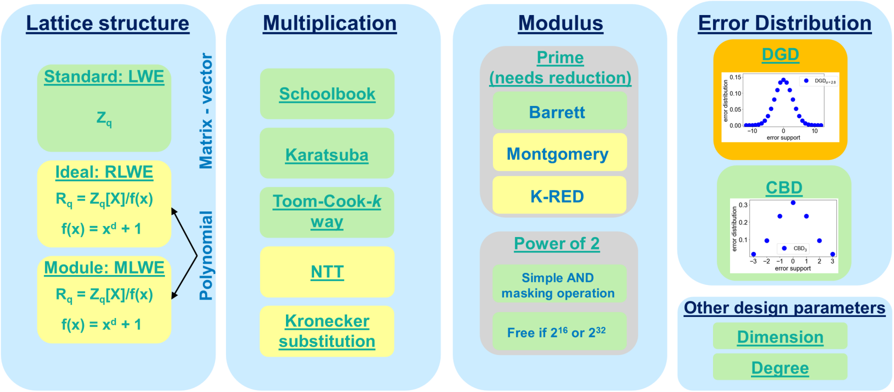
Conclusion - What did we gain?

Interesting potential against leakage

- **Linear** overhead in the shares number
- Trivial composition (key-homomorphic)
- Inherently good against glitches

$$\langle \boldsymbol{a}, \boldsymbol{k} \rangle = \langle \boldsymbol{a}, \boldsymbol{k}_0 \rangle + \langle \boldsymbol{a}, \boldsymbol{k}_1 \rangle + \cdots + \langle \boldsymbol{a}, \boldsymbol{k}_{\boldsymbol{d}-1} \rangle$$

Conclusion - Next steps (1/2)



Find an application of this **design space** (e.g. CPE encryption, signature)

Conclusion - Next steps (2/2)

Theoretical work:

- Understanding the impact of physical data dependencies
- **Reduction** towards standard learning problems

Appendix - Physical function

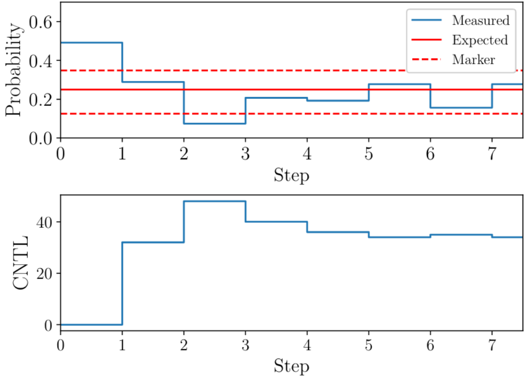


Figure 6: LPPN processor calibration: error probability (top) and control signal (bottom).