Leveraging Coprocessors as Noise Engines in Off-the-Shelf Microcontrollers

Balazs Udvarhelyi[1,2] and François-Xavier Standaert[1]

1 : UCLouvain, ICTEAM, Crypto Group, Louvain-la-Neuve, Belgium
2 : STMicroelectronics, Diegem, Belgium

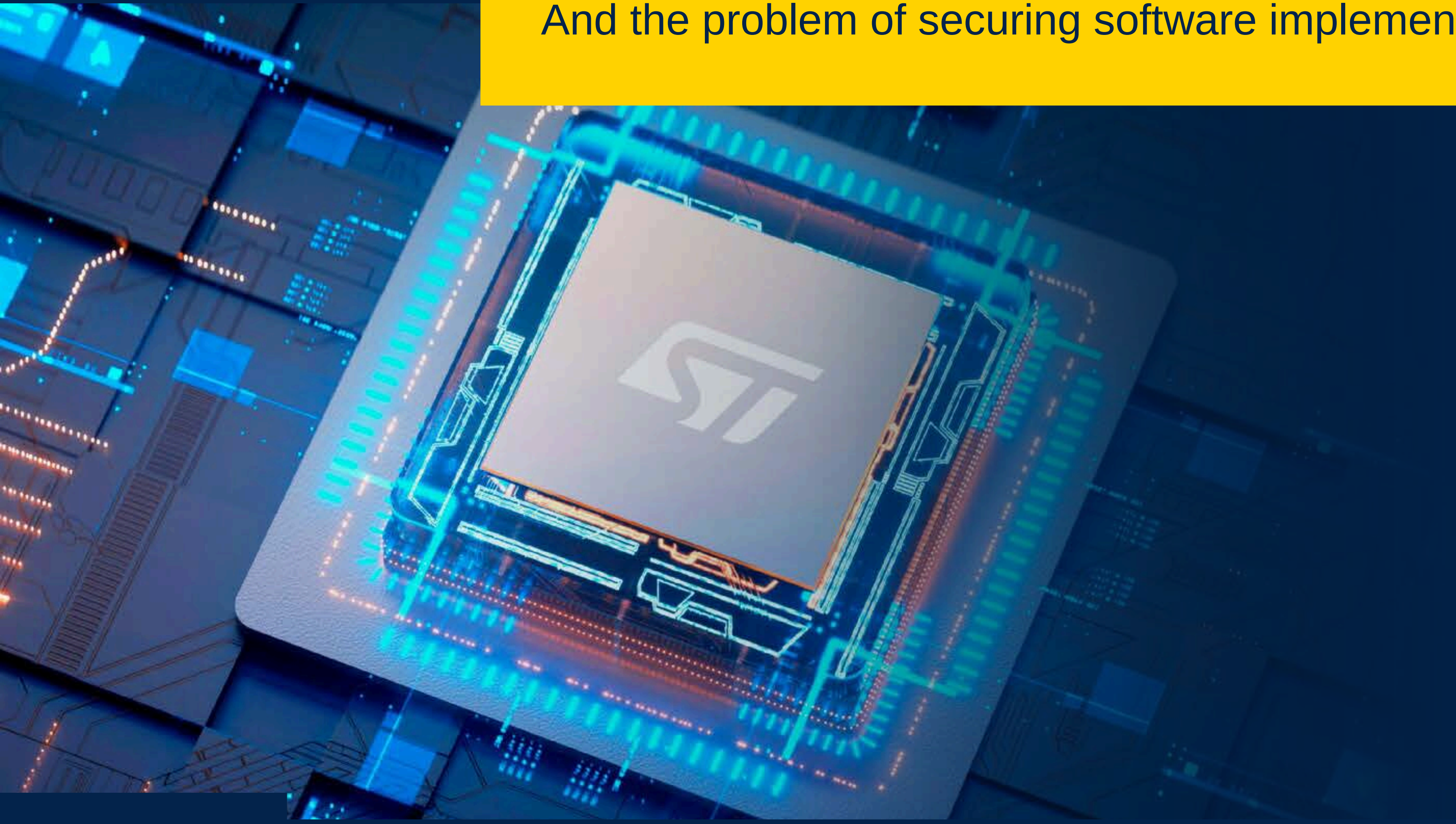
Agenda

1. Side-channel attacks in software
2. Exploiting MCU peripherals
3. Noise engine impact evaluation
4. Attack description & results
5. Conclusions

life.augmented

Side-channel attacks

And the problem of securing software implementations



life.augmented

Side-channel attacks

Designer goals:

- Minimize the information extracted from the leakages

In Software (MCUs):

- Limited & fixed inherent physical noise
- Additional countermeasures needed

Masking:

- Common countermeasure
- Amplifies present noise

Noise amplification countermeasures
need noise to be effective!

life.augmented

Masking: The principle

Computing on shares:

$$x = x_0 \oplus x_1 \oplus x_2 \oplus \ldots \oplus x_n$$

Attack complexity:

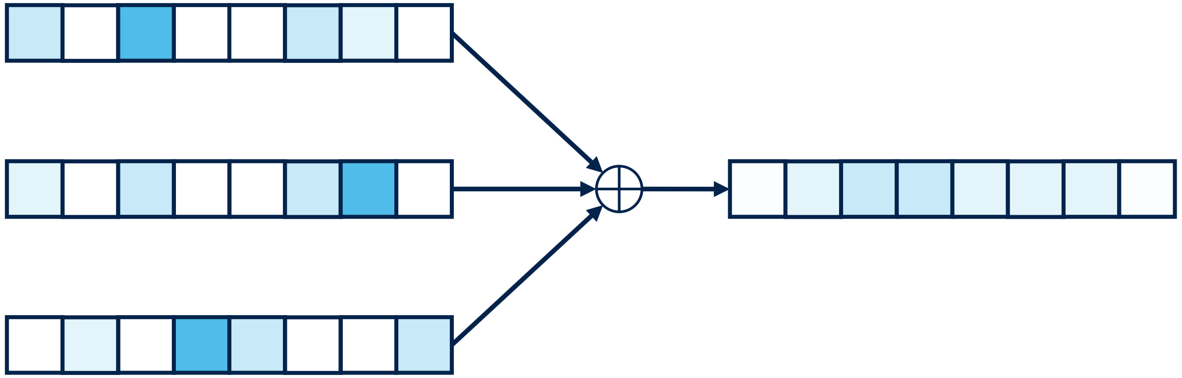
$$N \geq \frac{c}{\mathrm{MI}(X_i, L)^n}$$

Two conditions:

- Shares’ leakages are independent
- MI per share sufficiently low

- Unprotected probability $p(x|l) =$

- Masked probability $p(x|l) =$



life.augmented

Masking in software : The problem

CHES 2021 result: [BS21]

Breaking Masked Implementations with Many Shares on 32-bit Software Platforms

or When the Security Order Does Not Matter

Olivier Bronchain and François-Xavier Standaert

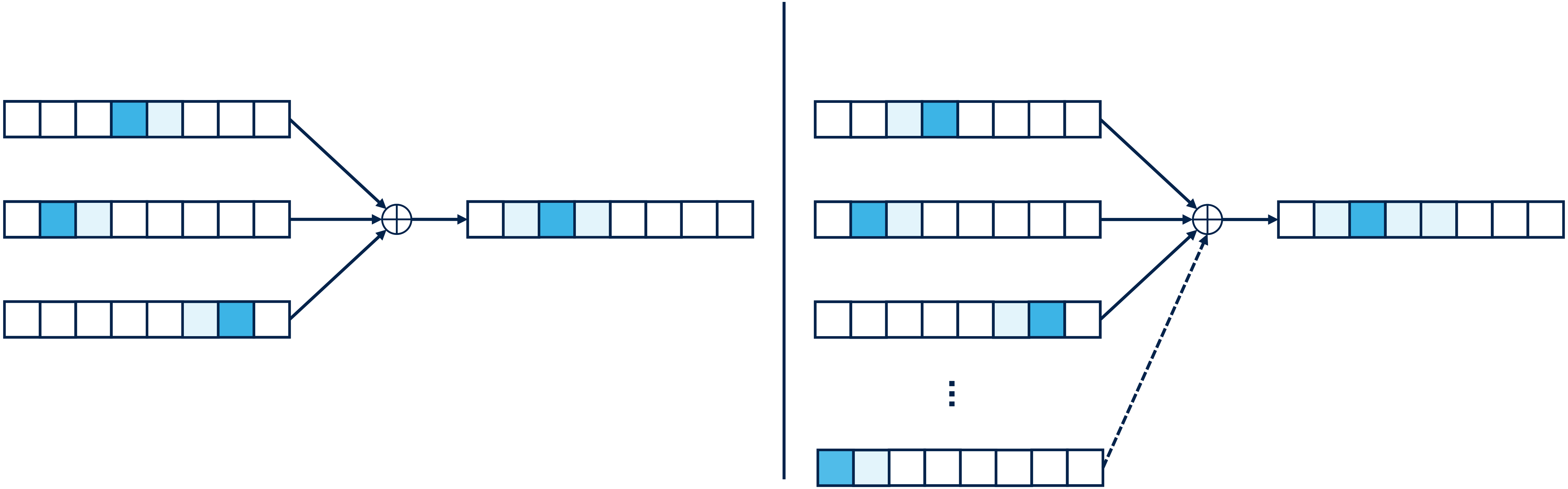
Crypto Group, ICTEAM Institute, UCLouvain, Louvain-la-Neuve, Belgium.
{olivier.bronchain,fstandae}@uclouvain.be

- In low end MCUs : Sufficient noise condition not met
- Slow increase of attack complexity w.r.t. # shares

Masking in software: The problem

Sufficient noise condition not met:

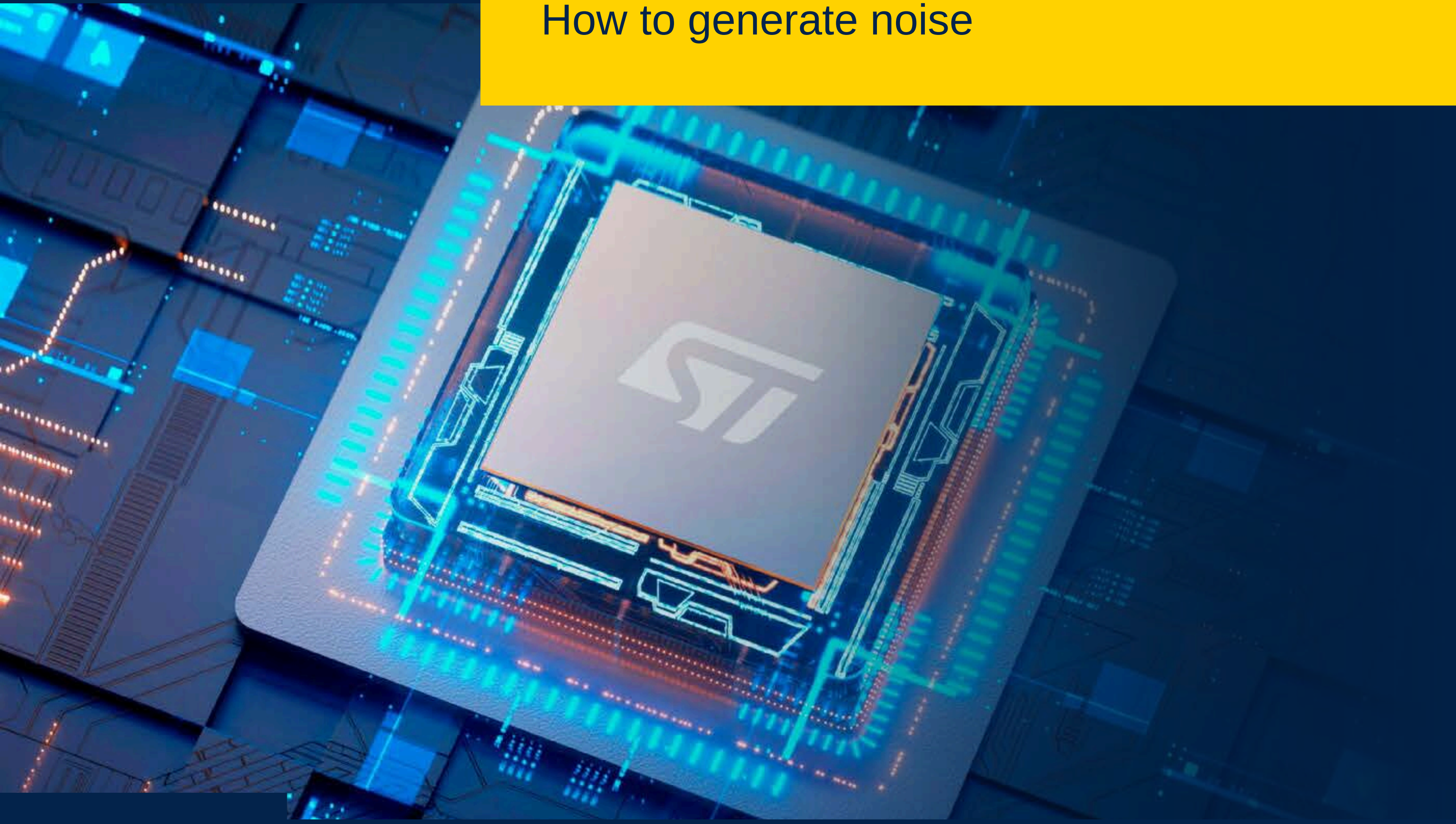
- Masking becomes useless (or at least very costly)



life.augmented

Exploiting MCU peripherals

How to generate noise



life.augmented

Noise engine characteristics

Ideal properties:

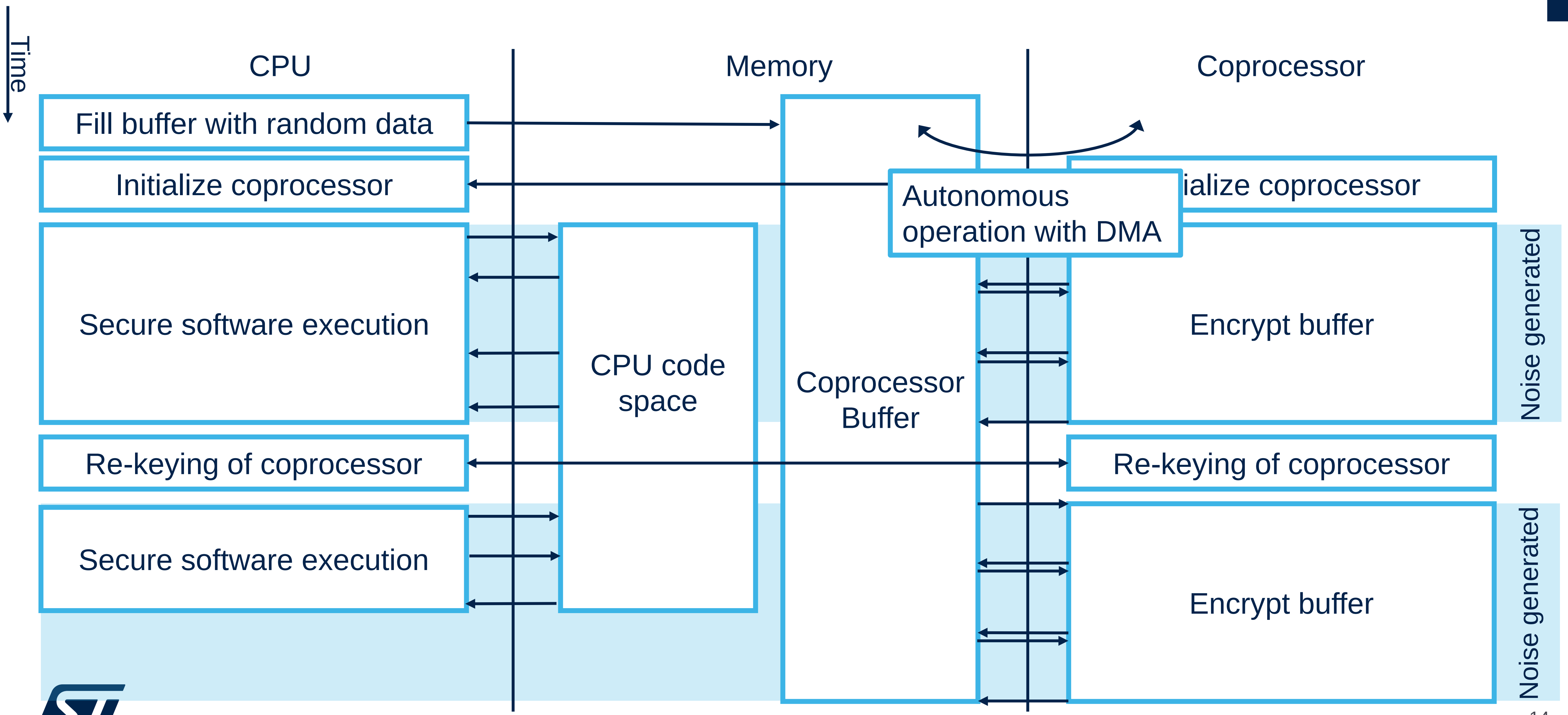
- Pseudorandom states
- Wide bus
- Same power source as CPU
- Continuous/long operation

→ Compatible peripherals are limited!

Our solution:

- AES-128 core
  - 16 cycles i.e. 128bit architecture (i.e. ≈1 round per clock cycle)
- Input and output buffer using DMA
- Autonomous operation during a full buffer of encryption
- Interrupt for reconfiguration of buffer
- Frequent re-keying of coprocessor

Execution scheme



Noise engine impact evaluation

life.augmented

Impact Evaluation

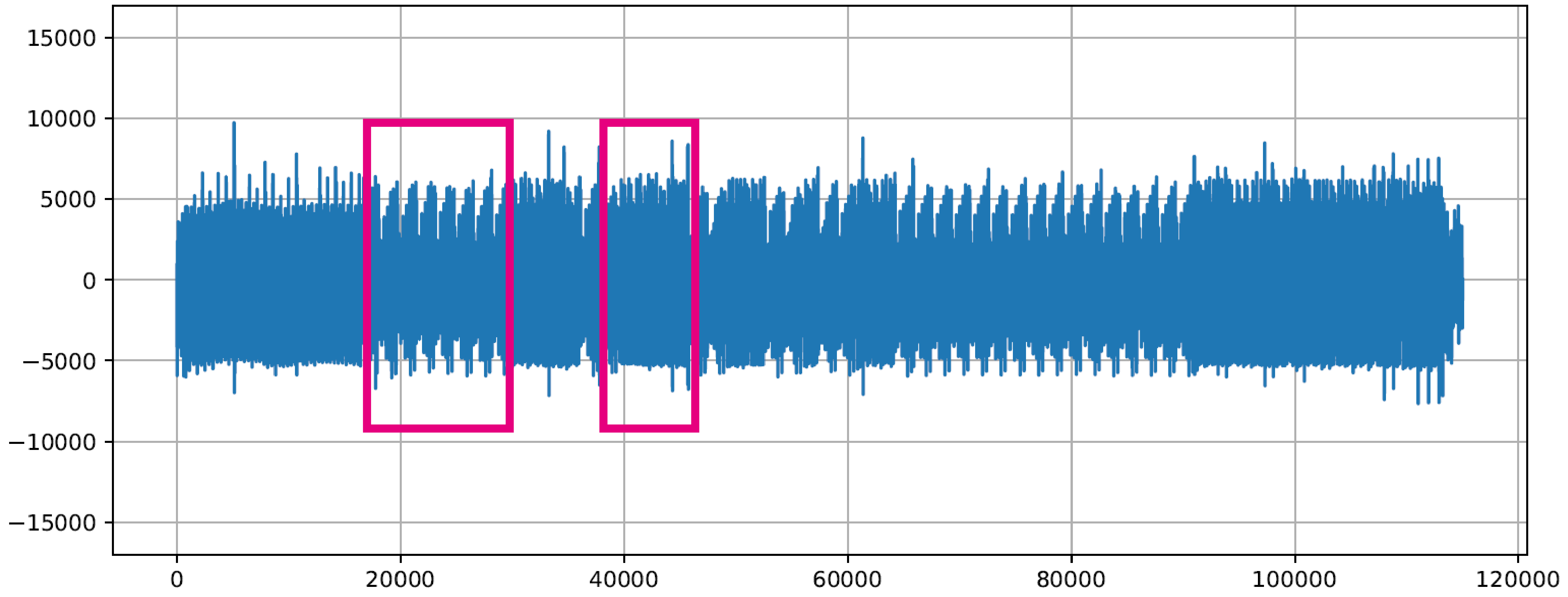
Target:

- Masked bitslice AES of Goudarzi and Rivain [GR17]
- PINI gadgets from [CS20]
- Gadgets' assembly code from [BC22]
- AES coprocessor based noise engine

Measurement process:

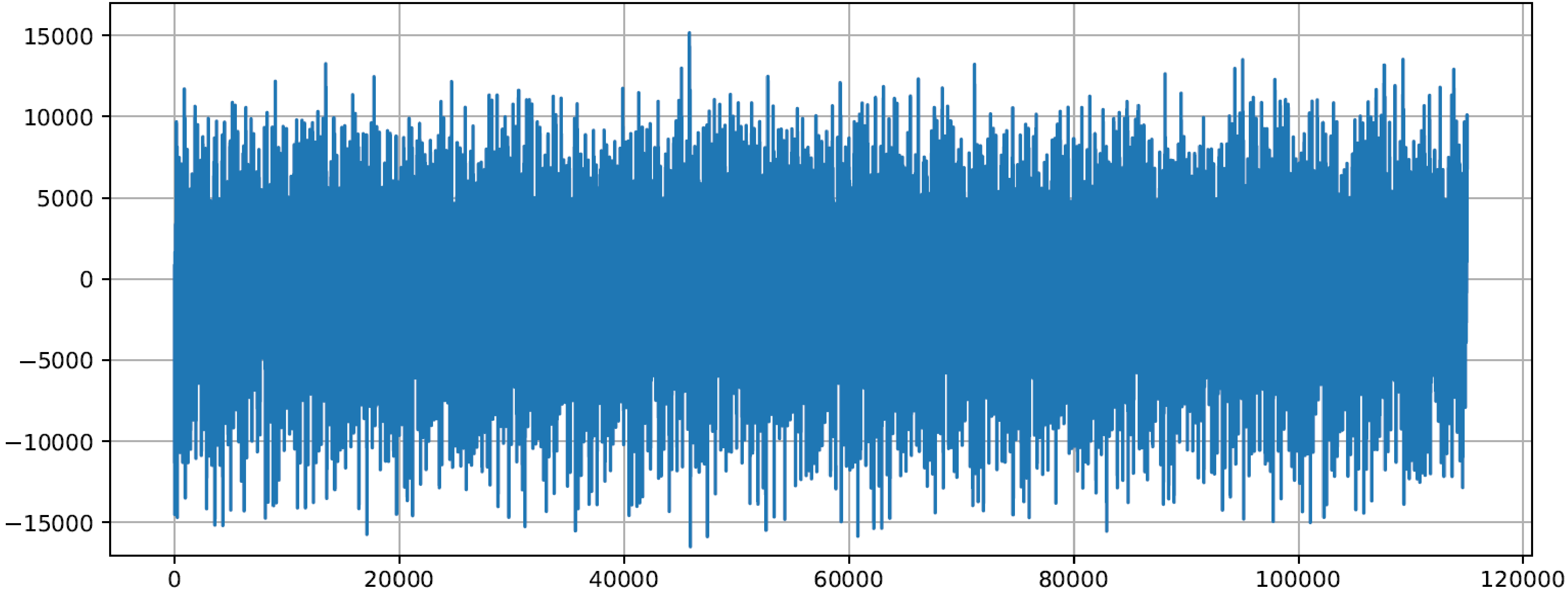
1. Reproduce measurements of [BS21] on ChipWhisperer CW308 with STM32 F0 (without AES coprocessor)
2. Swap daughterboard to STM32 F4 (with AES coprocessor)

Impact on raw leakage



Without noise generation

Impact on raw leakage



With noise generation

Information theoretic metrics

Signal to Noise Ratio:

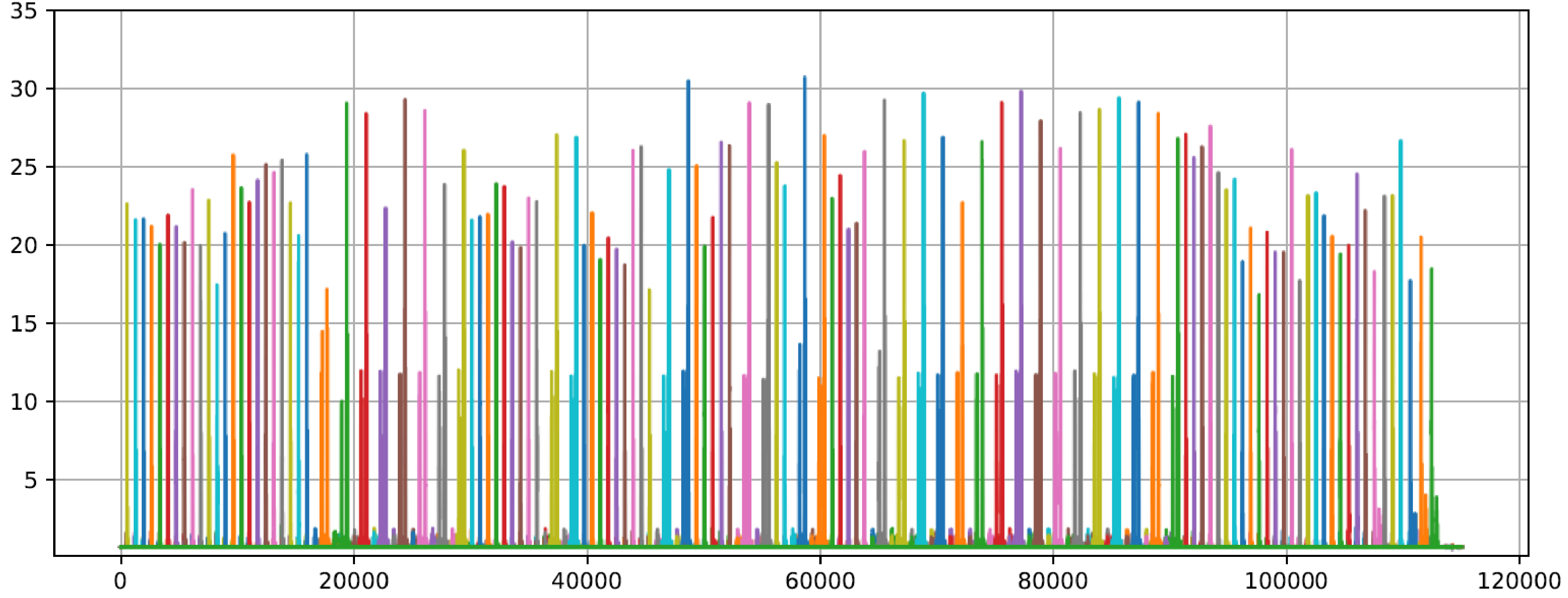
- Inter over intra class variance
- Calculated on 16-bit variables (Avoid impact of algorithmic noise)
- All shares & intermediate states of AES Sbox

- Calculated for each sample as:

$$\hat{\mathrm{SNR}} = \frac{\hat{\mathrm{Var}}_x\left(\hat{\mathrm{E}}_i\left(\boldsymbol{l}_{x,i}\right)\right)}{\hat{\mathrm{E}}_x\left(\hat{\mathrm{Var}}_i\left(\boldsymbol{l}_{x,i}\right)\right)}$$

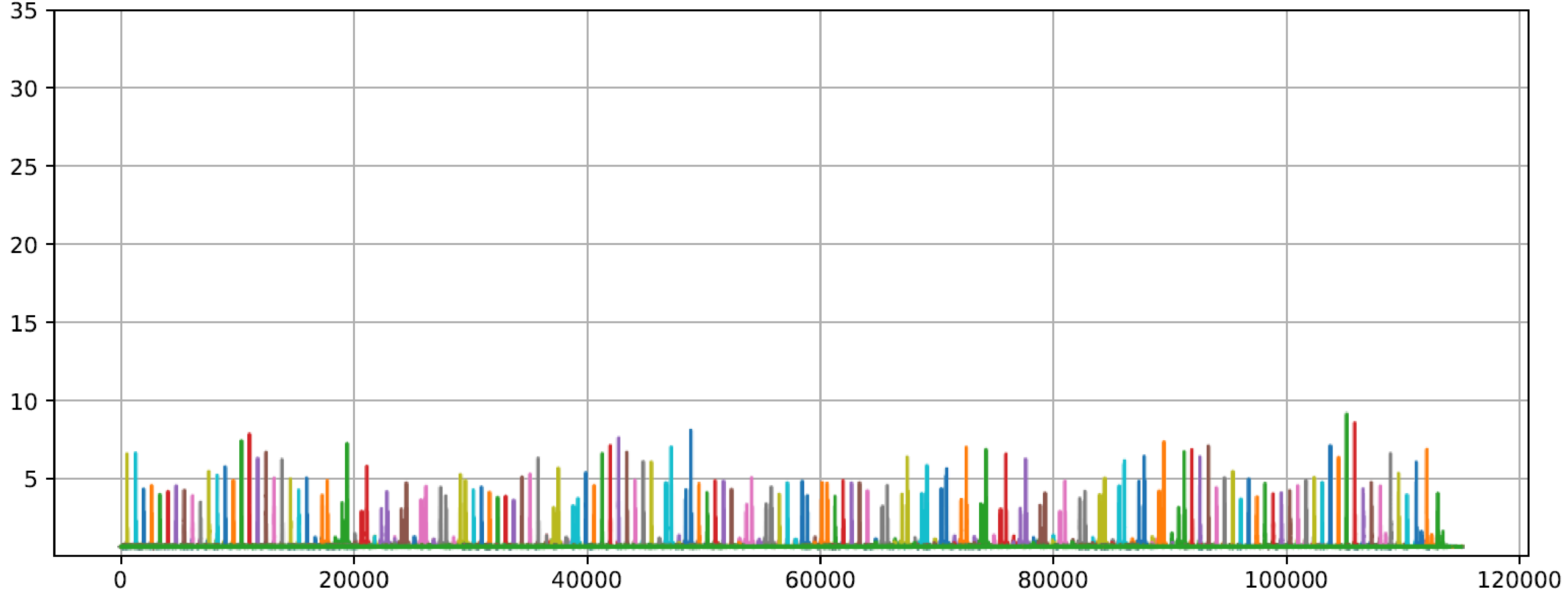
life.augmented

Impact on SNR



Without noise generation

Impact on SNR



With noise generation

Information theoretic metrics

Perceived Information (PI):

- Calculating Mutual Information (MI) is hard → Use bounds
- PI is a lower bound to the MI
- PI is multivariate
- Inversely proportional to attack complexity
- Easy to estimate by sampling the distribution:

$$\hat{\mathrm{PI}}(X, \boldsymbol{L}) = \mathrm{H}(X) + \frac{1}{|\mathcal{L}'|} \sum_{x \in \mathcal{X}} \sum_{\boldsymbol{l} \in \mathcal{L}'_x} \log_2 \hat{\mathrm{p}}[x|\boldsymbol{l}]$$

life.augmented

Impact on PI

Leakage model :

- Regression based LDA [CDSU23]
  → Extension of Gaussian templates :
  Efficient for long traces and large states
- 16-bit models
- ≈2000 POIs per model
- Reduced to 10 dimensions

Results :

- PI of the 8 input words of the Sbox
- Example for 2 shares

	Share #	Word 0	Word 1	Word 2	Word 3	Word 4	Word 5	Word 6	Word 7
No noise	0	1.90	1.53	1.21	1.89	0.81	2.00	2.47	3.12
	1	1.97	1.64	1.29	1.97	0.67	2.12	2.60	3.91
With noise	0	0.95	0.71	0.42	0.72	0.37	0.92	1.06	1.69
	1	1.06	0.67	0.46	0.76	0.30	1.10	1.12	1.78

Impacts on IT metrics

Impact on leakage traces:

- Leakage amplitude is higher
- Different operations are not distinguishable (XOR vs AND gadgets)

Impact on SNR:

- Clear reduction of SNR values
- No alteration of SNR curves' shape

Impact on PI:

- Reduction of PI per share by a factor ≈2
- Same behaviour for datasets with 2+ shares

life.augmented

Attack description & results

life.augmented

Attack description

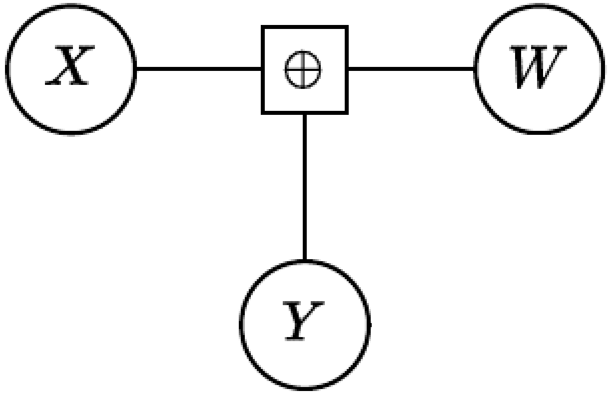
Baseline Template Attack (using RLDA model):

1. Profile each share of Sbox input words
2. Intermediate secret value : Recombine likelihoods on shares
3. Combine likelihoods of all traces

life.augmented

Soft Analytical Side-channel Attacks

- Several intermediate states can leak
- Profiling in the same template is not practical
- SASCA methodology:
  - Profile variables separately
  - Represent variables & relations in a factor graph
  - Use message passing algorithm: Belief propagation



Attack description

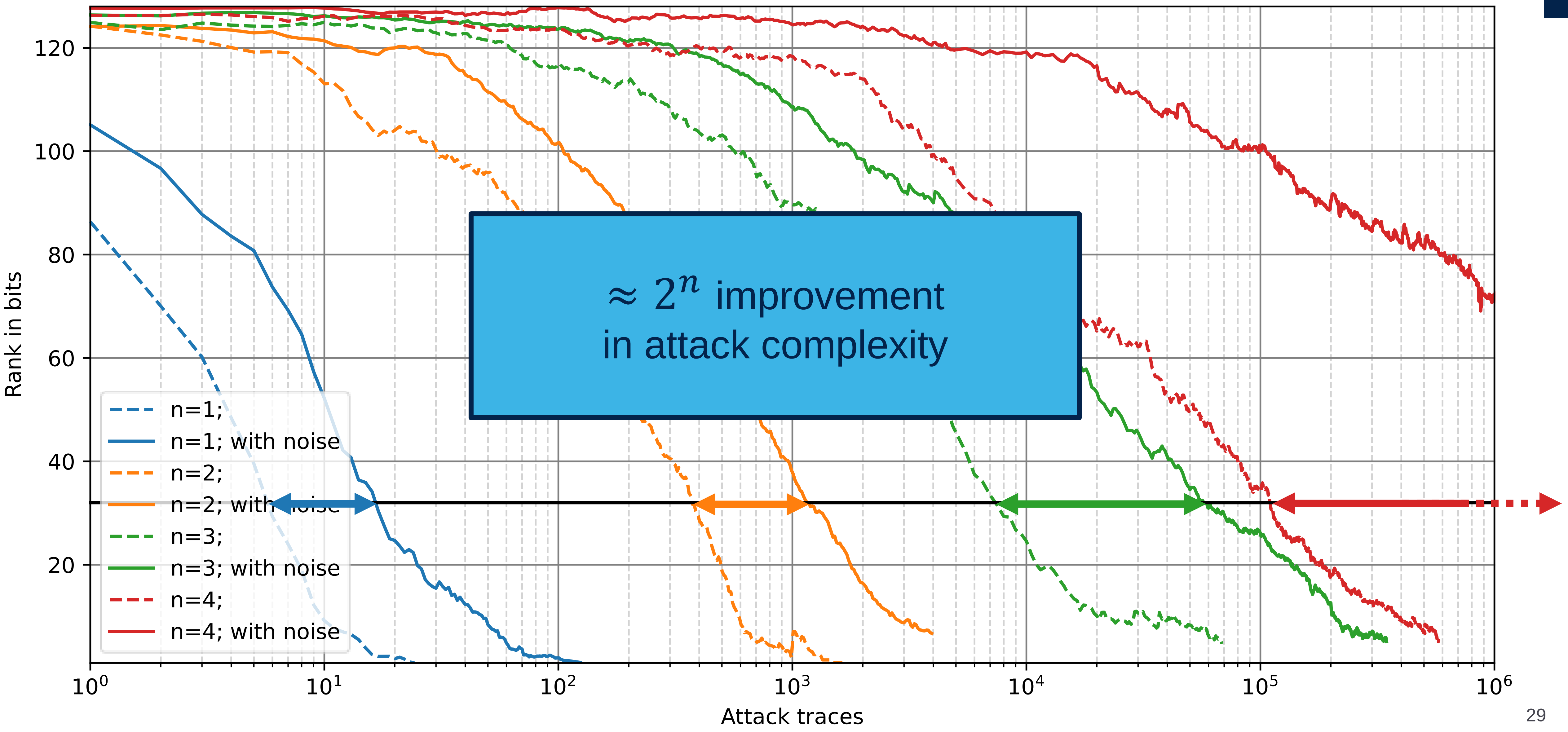
Baseline Template Attack:

1. Profile each share of Sbox input words
2. Intermediate secret value : Recombine likelihoods on shares
3. Combine likelihoods of all traces

SASCA of [BS21]:

1. Repeat 1. & 2. of Baseline attack for each Sbox variable
2. Run BP algorithm on Sbox
3. Combine likelihoods of all traces
4. Evaluate both attacks with histogram based rank estimation [PSG16].

Baseline attack results



SASCA results

Results :

- Very limited improvements over baseline attack
- Difference between attacks is smaller for higher orders of masking

Discussion:

- Lower PI on shares than [BS21]
  - STM32 F4 has smaller technology node (90nm vs 180nm)
- Propagation through factor graph is similar to masking
  As masking is not effective without noise:
  → SASCA is more effective when leakage is high

life.augmented

Conclusions

- Algorithmic noise can be generated with MCU peripherals
- Impact grows with higher orders of masking → Potential reduction of # shares
- Limited time overheads
- Can be combined with other countermeasures (shuffling, random delays)

Thank you!

life.augmented

Bibliography

- BC22: Olivier Bronchain and Gaëtan Cassiers. Bitslicing arithmetic/Boolean masking conversions for fun and prot with application to lattice-based kems. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2022(4):553-588, 2022.
- BS21: Olivier Bronchain and François-Xavier Standaert. Breaking masked implementations with many shares on 32-bit software platforms or when the security order does not matter. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2021(3):202-234, 2021.
- CDSU23:Gaëtan Cassiers, Henri Devillez, François-Xavier Standaert, and Balazs Udvarhelyi. Efficient regression-based linear discriminant analysis for sidechannel security evaluations: Towards analytical attacks against 32-bit implementations. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2023(3):270-293, 2023
- CS20: Gaëtan Cassiers and François-Xavier Standaert. Trivially and efficiently composing masked gadgets with probe isolating non-interference. *IEEE Trans. Inf. Forensics Secur.*, 15:2542-2555, 2020
- GR17: Dahmun Goudarzi and Matthieu Rivain. How fast can higher-order masking be in software? In *EUROCRYPT (1)*, volume 10210 *of Lecture Notes in Computer Science*, pages 567-597, 2017.
- PSG16: Romain Poussier, François-Xavier Standaert, and Vincent Grosso. Simple key enumeration (and rank estimation) using histograms: An integrated approach. In *CHES*, volume 9813 of *Lecture Notes in Computer Science*, pages 61-81. Springer, 2016

Our technology starts with You

life.augmented