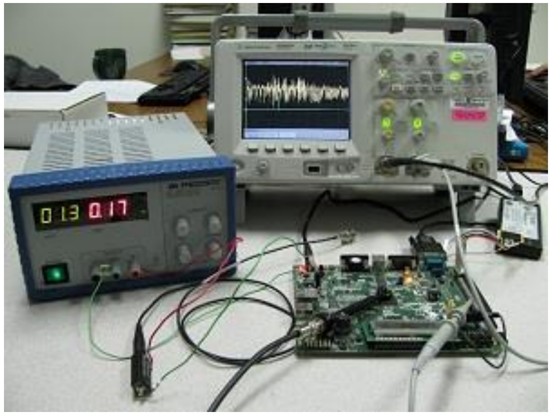
Towards an Open Approach to Side-Channel Resistant Authenticated Encryption



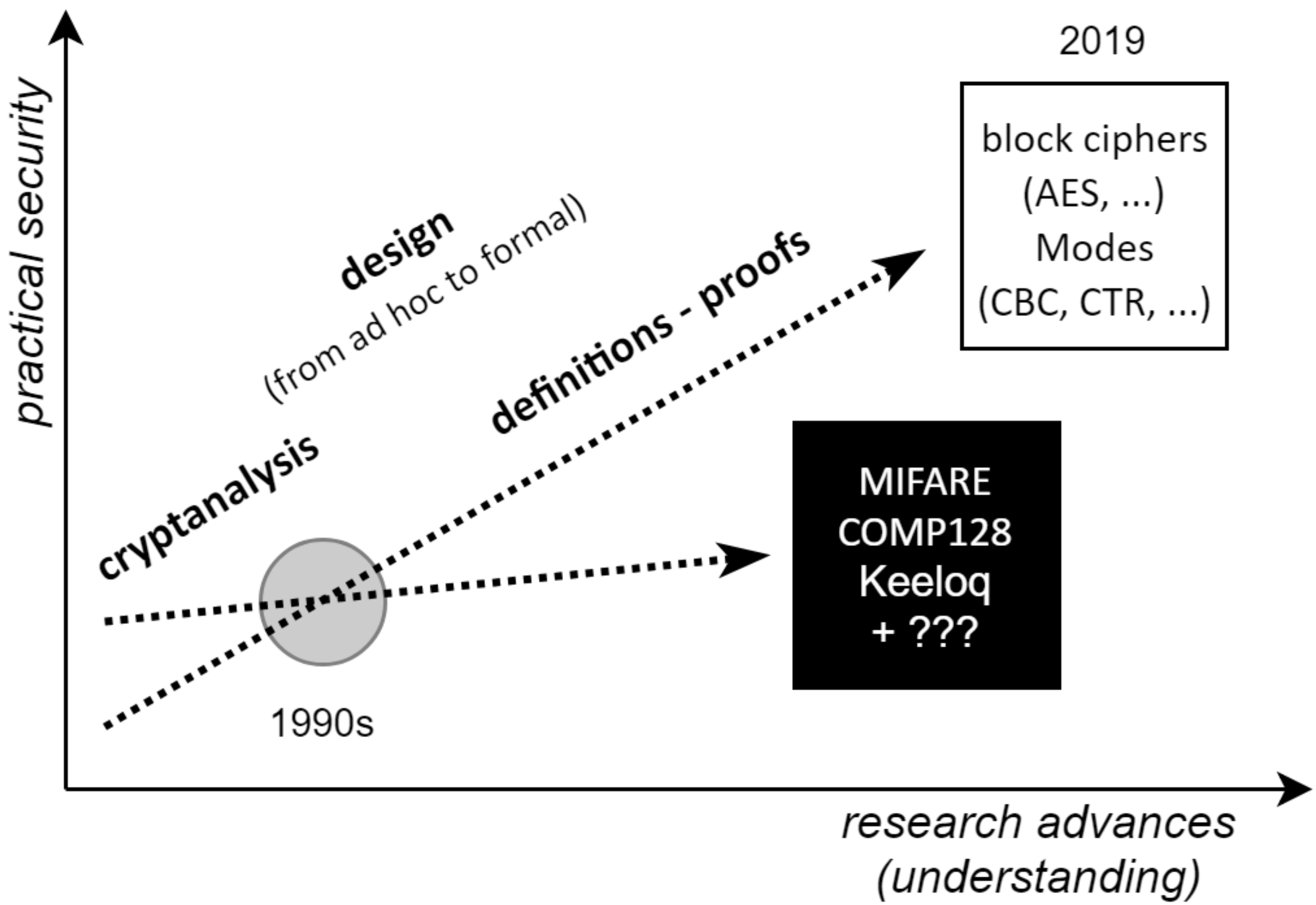
François-Xavier Standaert

UCLouvain, ICTEAM, Crypto Group (Belgium)

ASHES 2019, London, UK

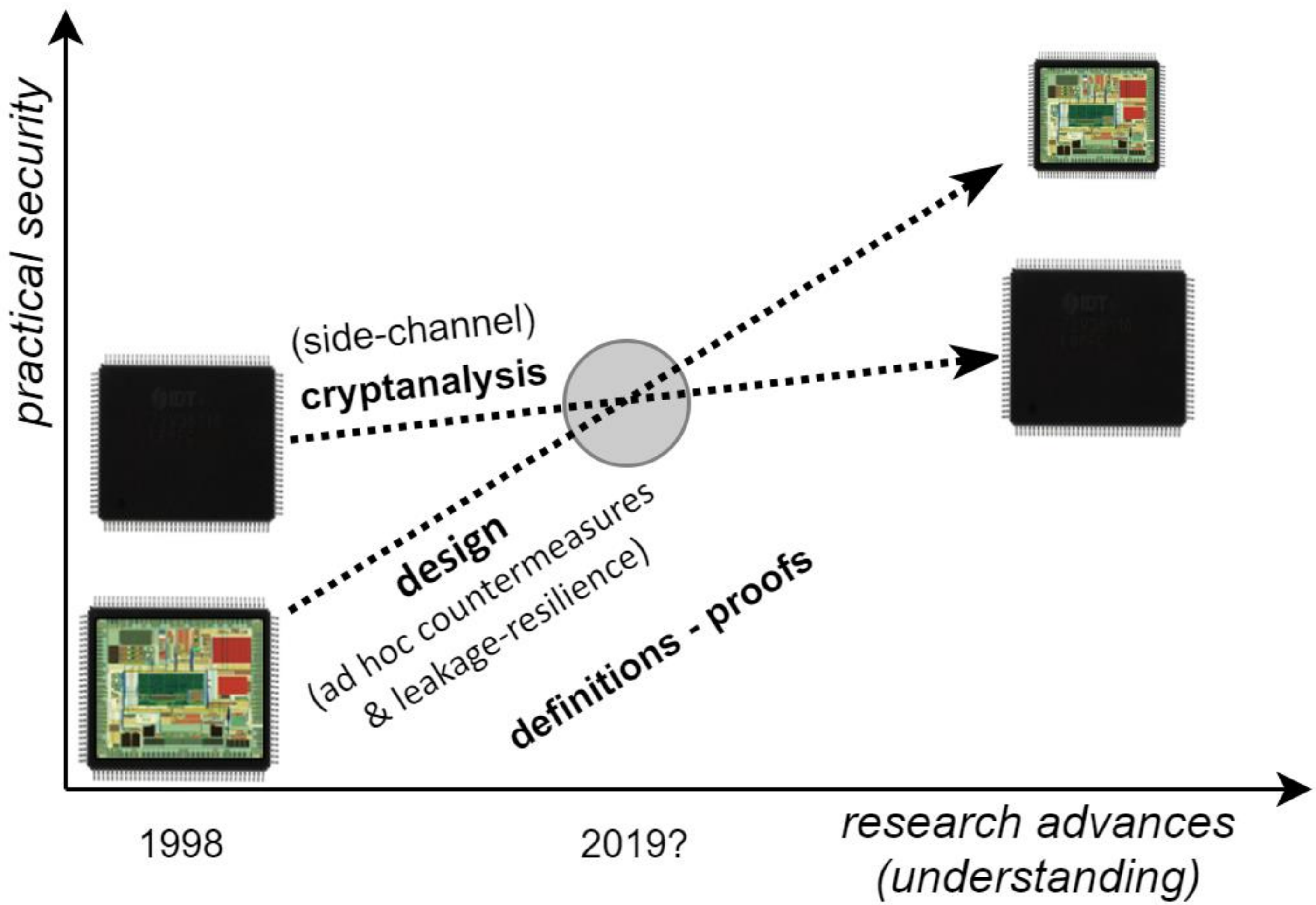
Transparency (as a measure of maturity)

- Block ciphers & symmetric encryption



Transparency (as a measure of maturity) II

- Secure cryptographic implementations



Outline

1. Side-channel **(crypt)analysis**: attacks taxonomy
2. Masking **countermeasure**: security vs. cost
3. Security **definitions** (authenticated encryption)
   a. Nonce-respecting setting (i.e., AEL)
   b. Nonce-misuse setting (i.e., AEmL)

1. Leakage-resistant AE **designs** (& implementations)
   - Level 0: no mode-level leakage-resistance
   - Level 1: re-keyed modes (including sponges)
   - Level 2: level 1 + strengthened init./final.
   - Level 3: level 2 + two-passes

2. Conclusions (& the need of open evaluations)

Acknowledgments

IV

C. Archambeau
J. Balasch
G. Barthe
S. Belaïd
D. Bellizia
F. Berti
O. Bronchain
G. Cassiers
C. Dobraunig
A. Duc
F. Dupressoir
F. Durvaux
S. Duval
S. Dziembowski
S. Faust
P.-A. Fouque
B. Gierlichs
C. Glowacz
D. Goudarzi
B. Grégoire
V. Grosso
S. Guilley
T. Güneysu
Chun Guo
Qian Guo
G. Herold
A. Journault
D. Kamel
G. Leander
L. Lerman
G. Leurent
I. Levi
T. Malkin
S. Mangard
D. Masny
C. Massart
P. Méaux
M. Medwed
C. Momin
A. Moradi
M. Naya-Plasencia
A. Olshevsky
Y. Oren
E. Oswald
C. Paglialonga
O. Pereira
T. Peters
C. Petit
K. Pietrzak
R. Poussier
E. Prouff
F. Regazzoni
M. Renauld
O. Reparaz
M. Rivain
T. Schneider
J. Schüth
P.-Y. Strub
N. Veyrat-Charvillon
S. Vivek
Weijia Wang
C. Whitnall
Yu Yu
M. Yung

Acknowledgments & cautionary note

C. Archambeau
J. Balasch
G. Barthe
S. Belaïd
D. Bellizia
F. Berti
O. Bronchain
G. Cassiers
C. Dobraunig
A. Duc
F. Dupressoir
F. Durvaux
S. Duval
S. Dziembowski
S. Faust
P.-A. Fouque
B. Gierlichs
C. Glowacz
D. Goudarzi
B. Grégoire
V. Grosso
S. Guilley
T. Güneysu
Chun Guo
Qian Guo
G. Herold
A. Journault
D. Kamel
G. Leander
L. Lerman
G. Leurent
I. Levi
T. Malkin
S. Mangard
D. Masny
C. Massart
P. Méaux
M. Medwed
C. Momin
A. Moradi
M. Naya-Plasencia
A. Olshevsky
Y. Oren
E. Oswald
C. Paglialonga
O. Pereira
T. Peters
C. Petit
K. Pietrzak
R. Poussier
E. Prouff
F. Regazzoni
M. Renauld
O. Reparaz
M. Rivain
T. Schneider
J. Schüth
P.-Y. Strub
N. Veyrat-Charvillon
S. Vivek
Weijia Wang
C. Whitnall
Yu Yu
M. Yung

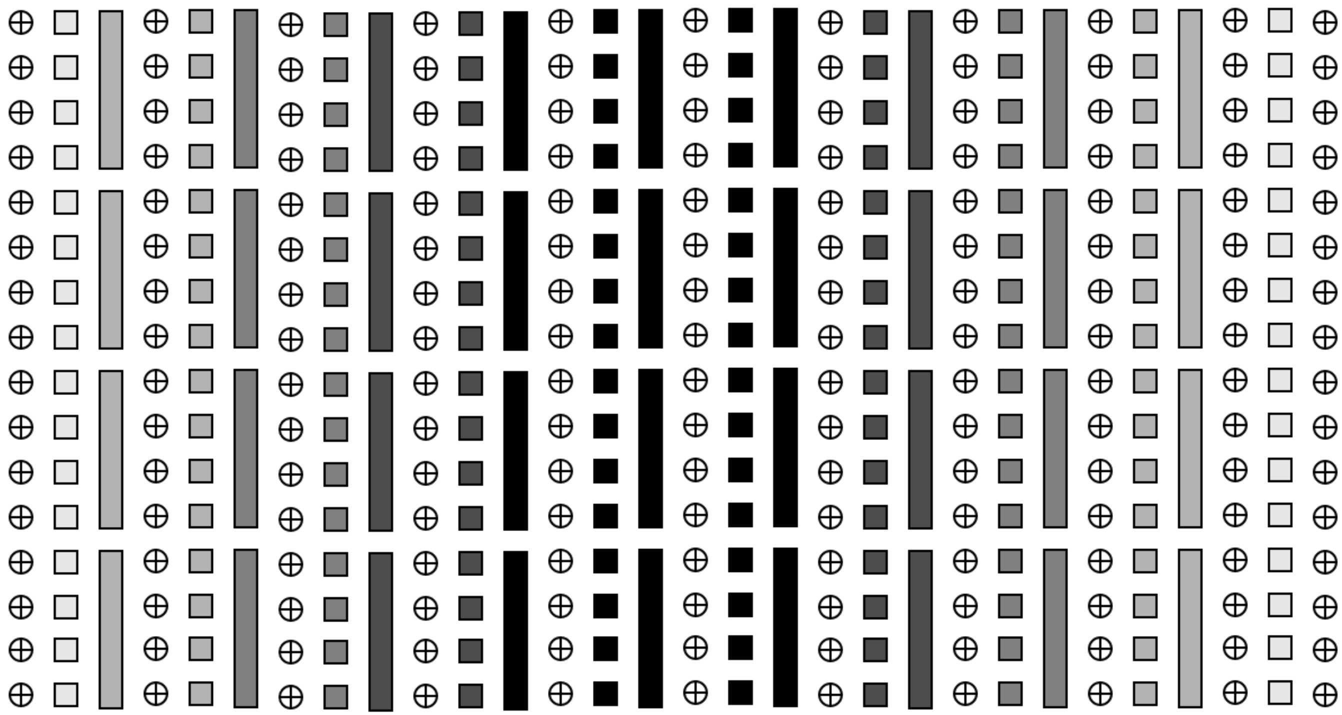
- Mixing (very) different abstraction levels
  - Hopefully in a consistent manner (*be forgiving if not*)

Outline

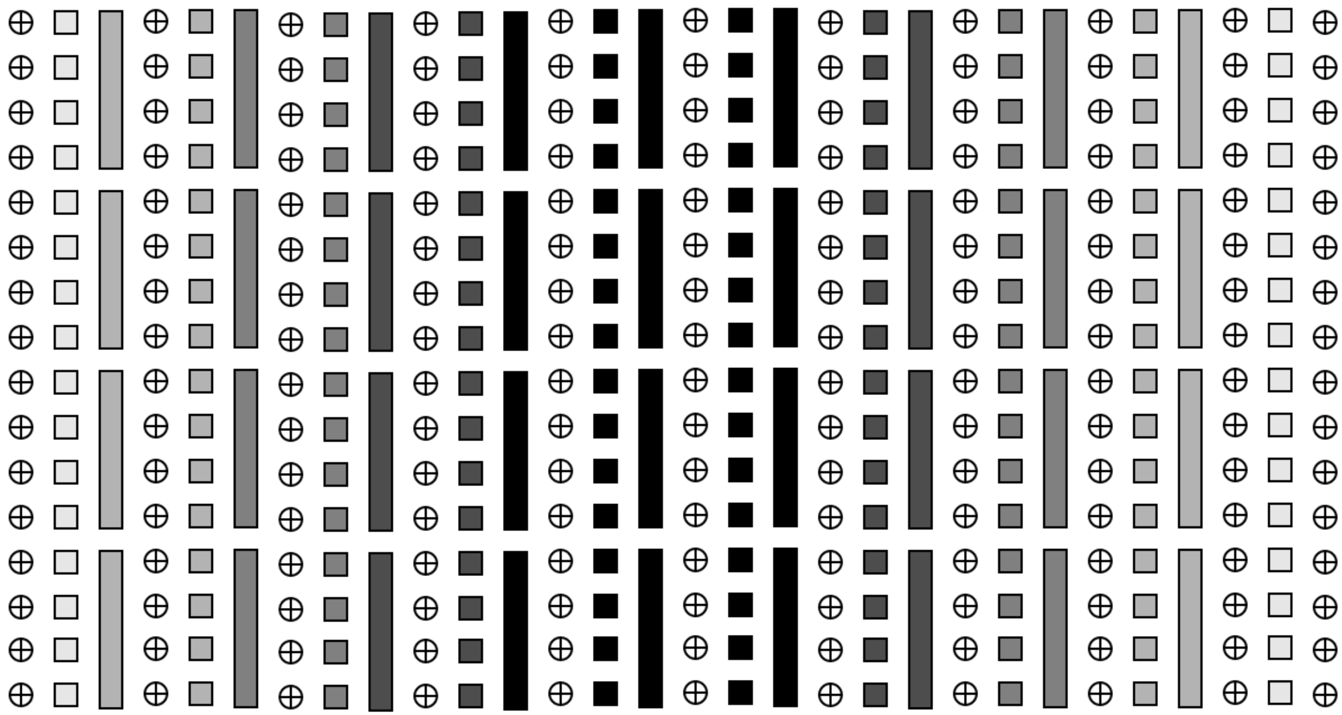
1. Side-channel **(crypt)analysis**: attacks taxonomy
2. Masking **countermeasure**: security vs. cost
3. Security **definitions** (authenticated encryption)
    a. Nonce-respecting setting (i.e., AEL)
    b. Nonce-misuse setting (i.e., AEmL)

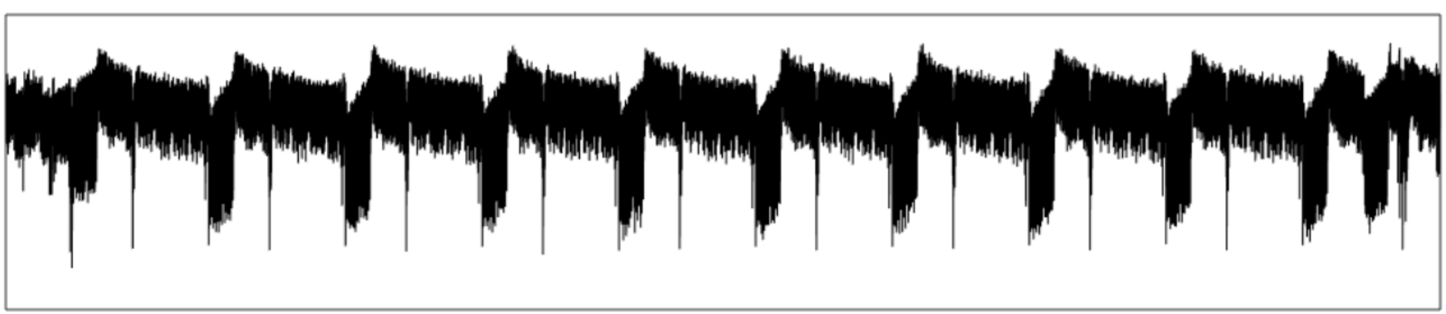
1. Leakage-resistant AE **designs** (& implementations)
    - Level 0: no mode-level leakage-resistance
    - Level 1: re-keyed modes (including sponges)
    - Level 2: level 1 + strengthened init./final.
    - Level 3: level 2 + two-passes
2. Conclusions (& the need of open evaluations)

AES Rijndael: $y = \mathrm{AES}_K(x)$



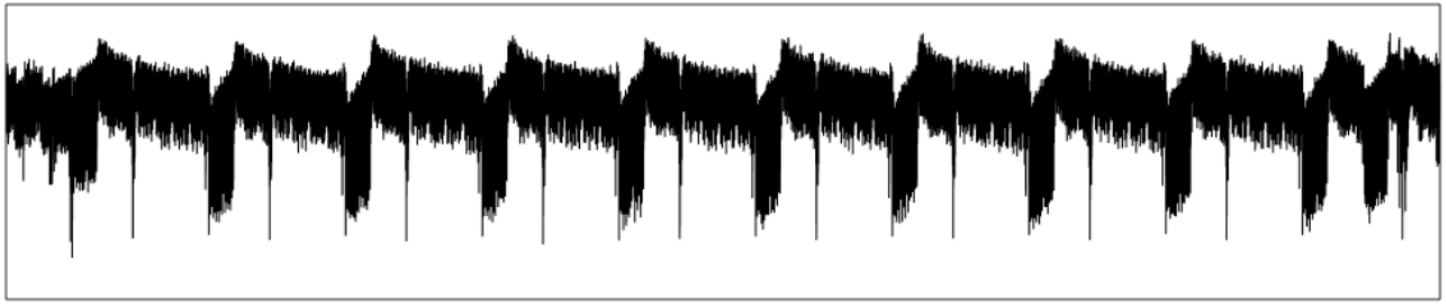
Leaking AES: $y = \text{AES}_K(x) \rightarrow \boldsymbol{L}$





Leakage function definition

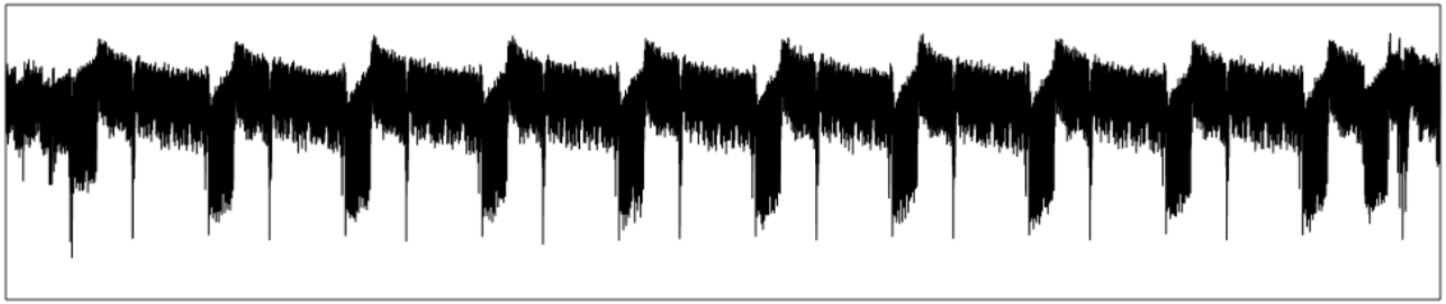
- Leakages are vectors: $\boldsymbol{L} = (L^1, L^2, \dots, L^t)$
  - Made of many samples ($t \approx 10^3$-10^6)



Leakage function definition

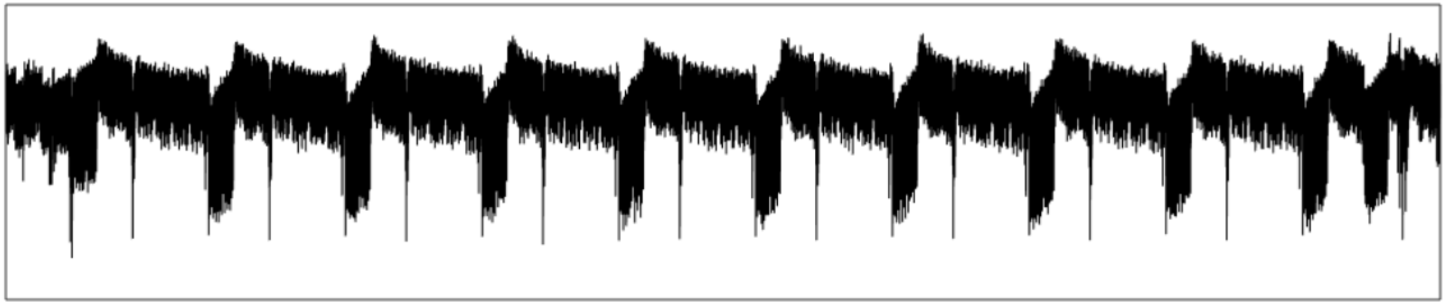
- Leakages are vectors: $\boldsymbol{L} = (L^1, L^2, \ldots, L^t)$
  - Made of many samples ($t \approx 10^3$-10^6)

- Leakages are noisy: $\boldsymbol{L}(x, K) \approx \boldsymbol{\delta}(x, K) + \boldsymbol{N}$
  - Signal-to-Noise Ratio: $\mathrm{SNR}^i = \dfrac{\mathrm{var}(\delta_x^i)}{\mathrm{var}(N^i)}$



Leakage function definition

- Leakages are vectors: $\boldsymbol{L} = (L^1, L^2, \ldots, L^t)$
  - Made of many samples ($t \approx 10^3$-10^6)
- Leakages are noisy: $\boldsymbol{L}(x, K) \approx \boldsymbol{\delta}(x, K) + \boldsymbol{N}$
  - Signal-to-Noise Ratio: $\mathrm{SNR}^i = \frac{\mathrm{var}(\delta_x^i)}{\mathrm{var}(N^i)}$
- The shape of $\boldsymbol{\delta}$ & $\boldsymbol{N}$ is technology-dependent
  - Their exact representation is unknown



Basic facts (I)

- Computing less means leaking less
  - E.g., unprotected **32-bit** implem. (**HW** leakages)

# rounds	# ops. / round	# samples / op.	MI (bits) / sample	λ (bits) / trace
10	100	5	$\frac{\textbf{log(32) = 5}}{\left(1+\frac{1}{\mathrm{SNR}}\right)}$	$\frac{\textbf{25,000}}{\left(1+\frac{1}{\mathrm{SNR}}\right)}$

Basic facts (I)

- Computing less means leaking less
  - E.g., unprotected **32-bit** implem. (**HW** leakages)

# rounds	# ops. / round	# samples / op.	MI (bits) / sample	λ (bits) / trace
10	100	5	$\frac{\log(32) = 5}{\left(1 + \frac{1}{\mathrm{SNR}}\right)}$	$\frac{25{,}000}{\left(1 + \frac{1}{\mathrm{SNR}}\right)}$

  - Unprotected **128-bit** implem. (**HW** leakages)

# rounds	# ops. / round	# samples / op.	MI (bits) / sample	λ (bits) / trace
10	1	5	$\frac{\log(128) = 7}{\left(1 + \frac{1}{\mathrm{SNR}}\right)}$	$\frac{350}{\left(1 + \frac{1}{\mathrm{SNR}}\right)}$

Consequence (for theoretical analysis) 1.4

- Games that give the adversary the ability to compare the leakages of two identical device states are in general trivial to win. For example, given a keyed *offline leakage oracle* $\boldsymbol{L}(.,K)$:

$$\Pr\left[A_{\mathrm{SC}}^{\boldsymbol{L}(.,K)}\left(x_0, x_1, \boldsymbol{L}(x_b, K)\right) = b | K, b \leftarrow \$\right] \approx 1$$

  - Just compare $\boldsymbol{L}(x_b, K)$ with $\boldsymbol{L}(x_0, K)$ and $\boldsymbol{L}(x_1, K)$
  - (SC stands for « state comparison » attack)

Consequence (for theoretical analysis)

- Games that give the adversary the ability to compare the leakages of two identical device states are in general trivial to win. For example, given a keyed *offline leakage oracle* $\boldsymbol{L}(.,K)$:

$$\Pr\left[A_{\mathrm{SC}}^{\boldsymbol{L}(.,K)}\left(x_0, x_1, \boldsymbol{L}(x_b, K)\right) = b | K, b \leftarrow \$\right] \approx 1$$

  - Just compare $\boldsymbol{L}(x_b, K)$ with $\boldsymbol{L}(x_0, K)$ and $\boldsymbol{L}(x_1, K)$
  - (SC stands for « state comparison » attack)

⇒ Distinguishing games without anything fresh *and* secret in the challenge are trivial to win

Basic facts (II)

- Key recovery attacks may not easily exploit all leakage samples (since A needs to guess the state), leading to reduced « effective » λ's, e.g.,

exploited # rounds	# ops. / round	# samples / op.	MI (bits) / sample	eff. λ (bits) / trace & subkey
1	1	$\approx$ **2 (indep.)**	$\frac{\log(128) = 7}{\mathbf{100}}$	$\frac{14}{100}$

- One key byte recovered in $\approx \frac{128}{0.14} \approx 1000$ traces

- Key recovery attacks may not easily exploit all leakage samples (since A needs to guess the state), leading to reduced « effective » λ's, e.g.,

exploited # rounds	# ops. / round	# samples / op.	MI (bits) / sample	eff. λ (bits) / trace & subkey
1	1	≈ **2 (indep.)**	$\frac{\log(128) = 7}{\mathbf{100}}$	$\approx \left(\frac{14}{100}\right)^{\boldsymbol{d}}$

- With the masking countermeasures (see next)

Basic facts (II)

- Key recovery attacks may not easily exploit all leakage samples (since A needs to guess the state), leading to reduced « effective » λ's, e.g.,

exploited # rounds	# ops. / round	# samples / op.	MI (bits) / sample	eff. λ (bits) / trace & subkey
1	1	$\approx$ **2 (indep.)**	$\frac{\log(128) = 7}{\mathbf{100}}$	$\approx \left(\frac{14}{100}\right)^{d}$

- With the masking countermeasures (see next)
- (*128-bit example, 32-bit case significantly harder*)

Basic facts (III)

- (q, r)-bounded SCAs are « continuous » attacks
  - with q different message blocks per key
  - and each measurement repeated r times

⇒ Typical success probability (e.g., for key recovery):

$$\Pr\left[A_{\mathrm{KR}}\left(x_1, \boldsymbol{L}(x_1, K), \dots, x_q, \boldsymbol{L}\left(x_q, K\right)\right) \rightarrow K | K \leftarrow \$\right] \approx 2^{-128+q\cdot\lambda(r)}$$

- (q, r)-bounded SCAs are « continuous » attacks
  - with q different message blocks per key
  - and each measurement repeated r times

⇒ Typical success probability (e.g., for key recovery):

$$\Pr\left[A_{\mathrm{KR}}\left(x_1, \boldsymbol{L}(x_1, K), \dots, x_q, \boldsymbol{L}\left(x_q, K\right)\right) \rightarrow K | K \leftarrow \$\right] \approx 2^{-128+q\cdot\lambda(r)}$$

- There are two main types of attacks (jargon)
  - SPA: q is a small constant (e.g., thanks to re-keying)
  - DPA: q can be large & is adversarially chosen

Basic facts (III)

- (q, r)-bounded SCAs are « continuous » attacks
  - with q different message blocks per key
  - and each measurement repeated r times

⇒ Typical success probability (e.g., for key recovery):

$$\Pr\left[A_{\mathrm{KR}}\left(x_1, \boldsymbol{L}(x_1, K), \dots, x_q, \boldsymbol{L}\left(x_q, K\right)\right) \rightarrow K | K \leftarrow \$\right] \approx 2^{-128+q\cdot\lambda(r)}$$

- There are two main types of attacks (jargon)
  - SPA: q is a small constant (e.g., thanks to re-keying)
  - DPA: q can be large & is adversarially chosen
- Larger r's can improve the SNR (average the noise)

Summarizing (taxonomy of attacks) 1.7

- Key Recovery (KR) attacks (with known/chosen x_i's)

$$\Pr\left[A_{\mathrm{KR}}\left(x_1, \boldsymbol{L}(x_1, K), \dots, x_q, \boldsymbol{L}\left(x_q, K\right)\right) \rightarrow K | K \leftarrow \$\right] \approx 2^{-128+q\cdot\lambda(r)}$$

  - May require large amounts of leakage vectors to succeed

Summarizing (taxonomy of attacks)

- Key Recovery (KR) attacks (with known/chosen x_i's)

$$\Pr\left[A_{\mathrm{KR}}\left(x_1, \boldsymbol{L}(x_1, K), \dots, x_q, \boldsymbol{L}\left(x_q, K\right)\right) \rightarrow K \mid K \leftarrow \$\right] \approx 2^{-128+q\cdot\lambda(r)}$$

  - May require large amounts of leakage vectors to succeed
    - Or have bounded success probability in case of SPA

Summarizing (taxonomy of attacks) 1.7

- Key Recovery (KR) attacks (with known/chosen x_i's)

$$\Pr\left[A_{\text{KR}}\left(x_1, \boldsymbol{L}(x_1, K), \dots, x_q, \boldsymbol{L}(x_q, K)\right) \to K | K \leftarrow \$\right] \approx 2^{-128+q\cdot\lambda(r)}$$

  - May require large amounts of leakage vectors to succeed
    - Or have bounded success probability in case of SPA

- State Comparison (SC) attacks (with keyed oracle)
  - $\Pr\left[A_{\text{SC}}^{\boldsymbol{L}(.,K)}\left(x_0, x_1, \boldsymbol{L}(x_b, K)\right) = b | K, b \leftarrow \$\right] \approx$ 1 anyway

Summarizing (taxonomy of attacks)

- Key Recovery (KR) attacks (with known/chosen x_i's)

$$\Pr\left[A_{\mathrm{KR}}\left(x_1, \boldsymbol{L}(x_1, K), \dots, x_q, \boldsymbol{L}(x_q, K)\right) \to K | K \leftarrow \$\right] \approx 2^{-128+q\cdot\lambda(r)}$$

  - May require large amounts of leakage vectors to succeed
    - Or have bounded success probability in case of SPA

- Message Comparison (MC) attacks (with fresh challenge)

$$\Pr\left[A_{\mathrm{MC}}^{\boldsymbol{L}(.,.)}\left(x_0, x_1, \boldsymbol{L}(x_b, K)\right) = b | K, b \leftarrow \$\right] \approx 2^{-128+\mathrm{D}\left(\boldsymbol{L}(x_0,K);\boldsymbol{L}(x_1,K)\right)}$$

  - Significanly simpler than KR - but not trivial for all x_0, x_1 (!)
  - Depends on similarity of the message blocks' leakages

- State Comparison (SC) attacks (with keyed oracle)
  - $\Pr\left[A_{\mathrm{SC}}^{\boldsymbol{L}(.,K)}\left(x_0, x_1, \boldsymbol{L}(x_b, K)\right) = b | K, b \leftarrow \$\right] \approx$ 1 anyway

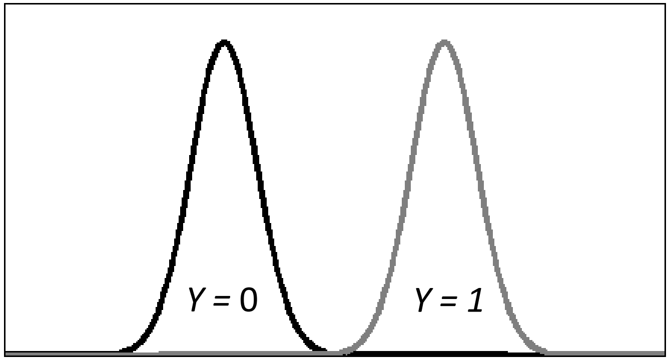
Outline

1. Side-channel **(crypt)analysis**: attacks taxonomy
2. Masking **countermeasure**: security vs. cost
3. Security **definitions** (authenticated encryption)
   a. Nonce-respecting setting (i.e., AEL)
   b. Nonce-misuse setting (i.e., AEmL)

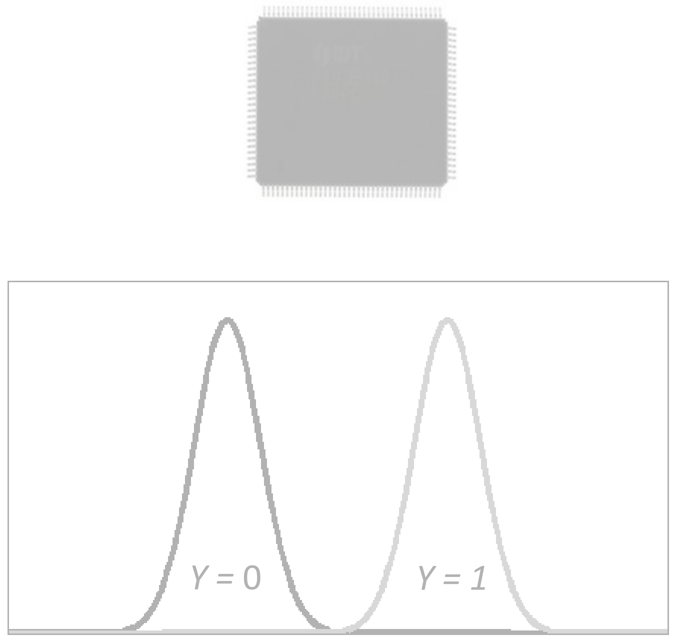
1. Leakage-resistant AE **designs** (& implementations)
   - Level 0: no mode-level leakage-resistance
   - Level 1: re-keyed modes (including sponges)
   - Level 2: level 1 + strengthened init./final.
   - Level 3: level 2 + two-passes

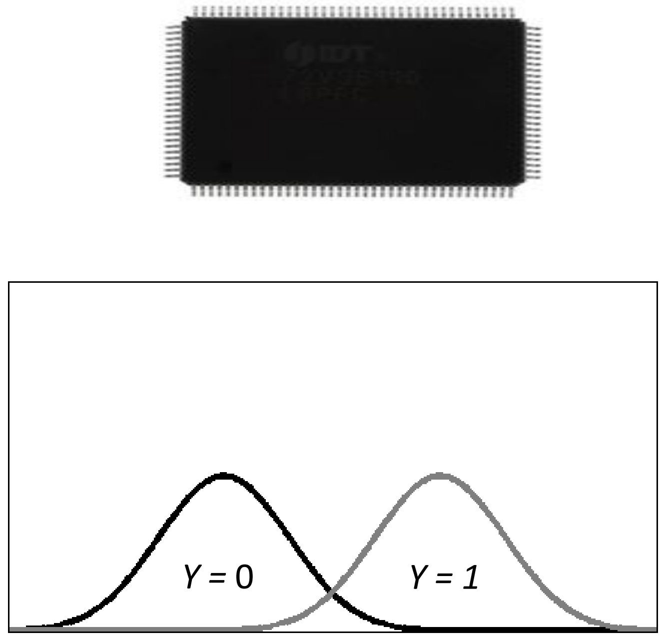
2. **Conclusions** (& the need of open evaluations)

Noise (hardware) is not enough

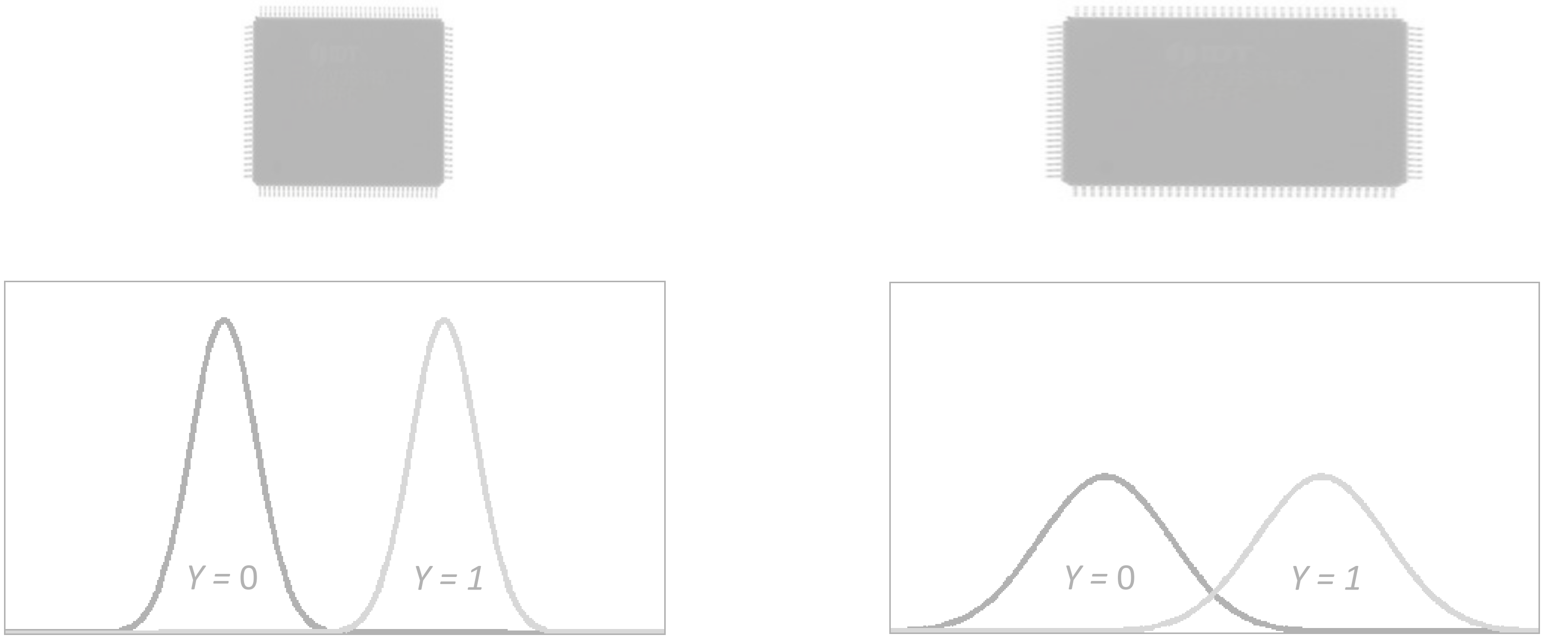


Noise (hardware) is not enough





Noise (hardware) is not enough



- Additive noise ≈ cost × 2 ⇒ security × 2
  ⇒ not a good (crypto) security parameter
- ≈ same holds for all hardware countermeasures

Masking (≈ noise amplification)

- Example: Boolean encoding

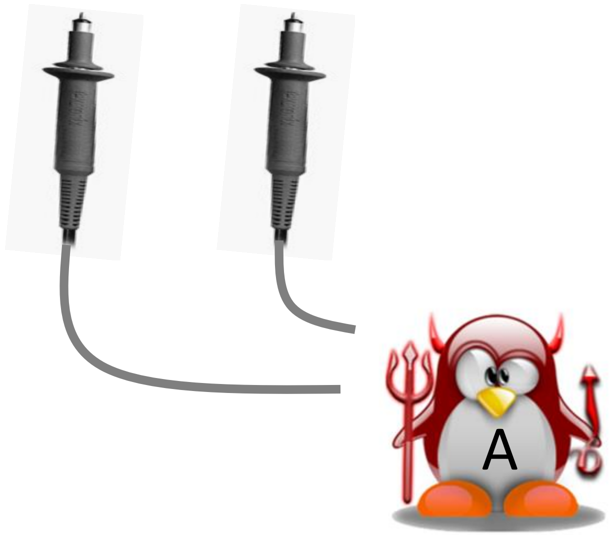
$$y = y_1 \oplus y_2 \oplus \cdots \oplus y_{d-1} \oplus y_d$$

- With $y_1, y_2, \ldots, y_{d-2}, y_{d-1} \leftarrow \{0,1\}^n$

Masking (abstract view)

- Private circuits / probing security [ISW03]

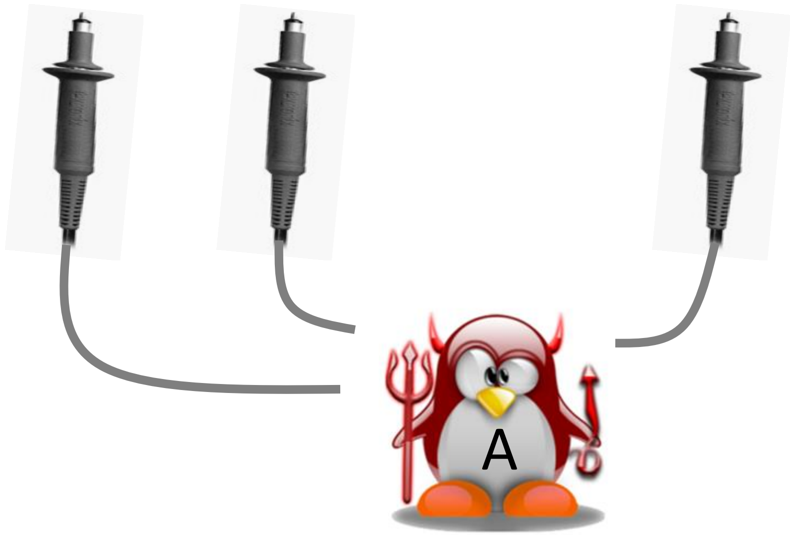
$$y = y_1 \oplus y_2 \oplus \cdots \oplus y_{d-1} \oplus y_d$$



Masking (abstract view)

- Private circuits / probing security [ISW03]

$$y = y_1 \oplus y_2 \oplus \cdots \oplus y_{d-1} \oplus y_d$$

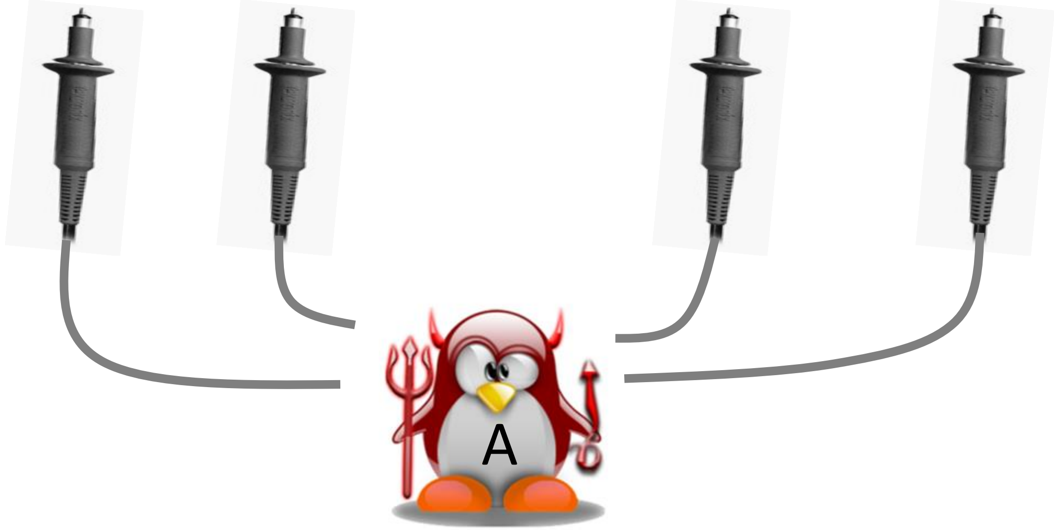


- $d - 1$ probes do not reveal anything on y

Masking (abstract view)

- Private circuits / probing security [ISW03]

$$y = y_1 \oplus y_2 \oplus \cdots \oplus y_{d-1} \oplus y_d$$

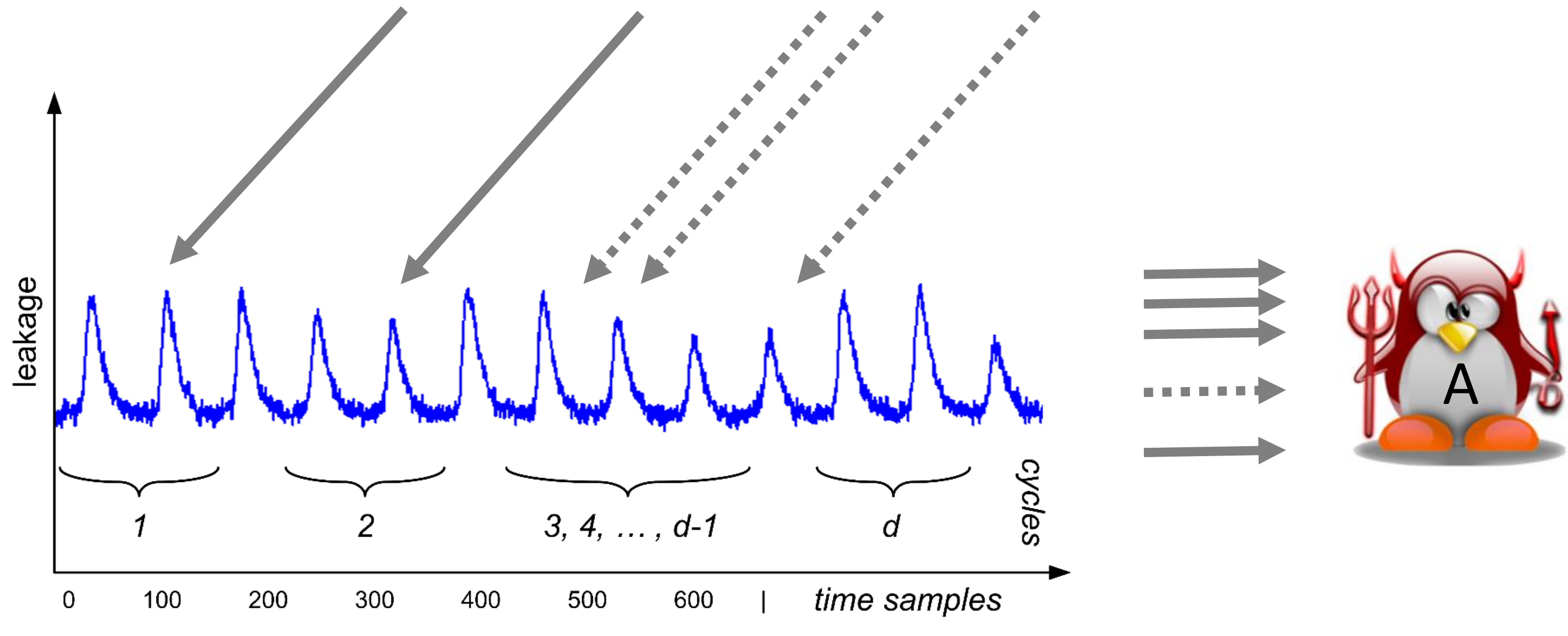


- But d probes completely reveal y

Masking (concrete view) 2.2

- Private circuits / probing security [ISW03]

$$y = y_1 \oplus y_2 \oplus \cdots \oplus y_{d-1} \oplus y_d$$



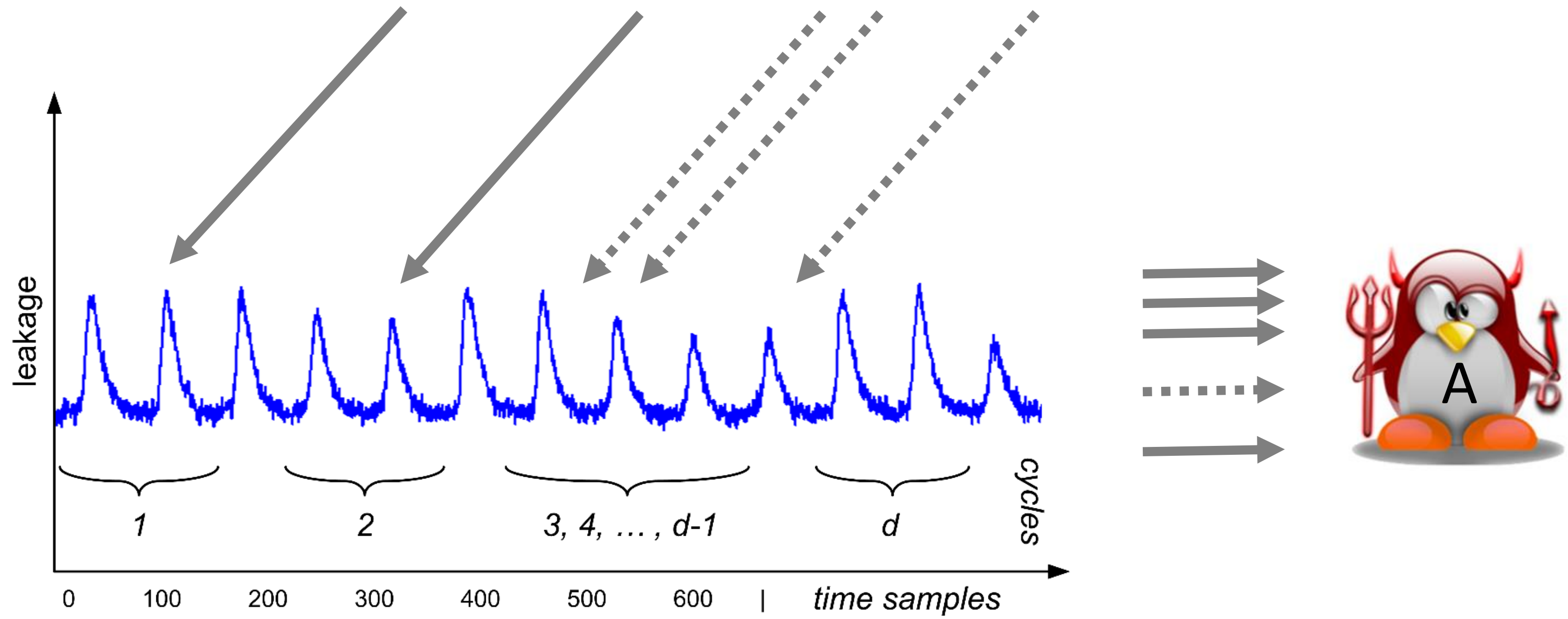
serial implementation.

- Noisy leakage security [PR13]

Masking (concrete view)

- Private circuits / probing security [ISW03]

$$y = y_1 \oplus y_2 \oplus \cdots \oplus y_{d-1} \oplus y_d$$

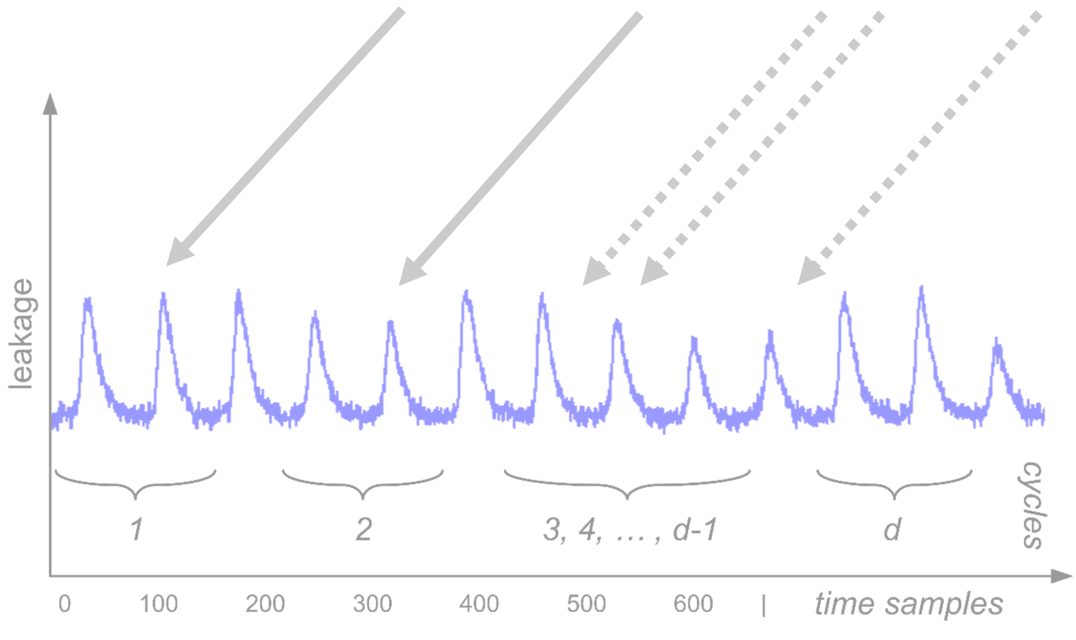


serial implementation.

- Bounded information $\mathrm{MI}(Y;\boldsymbol{L}) < \mathrm{MI}(Y_i;\boldsymbol{L}_{Y_i})^d$

- Private circuits / probing security [ISW03]

$$y = y_1 \oplus y_2 \oplus \cdots \oplus y_{d-1} \oplus y_d$$



serial implementation.

noise + independence
[DDF14]

- Bounded information $\mathrm{MI}(Y;\boldsymbol{L})<\mathrm{MI}(Y_i;\boldsymbol{L}_{Y_i})^d$

Masked operations [ISW03]

- Linear operations: $\mathrm{f(a)} = \mathrm{f}(a_1) \oplus \mathrm{f}(a_2) \oplus \cdots \oplus \mathrm{f}(a_d)$

Masked operations [ISW03]

- Linear operations: $\mathrm{f(a)} = \mathrm{f}(a_1) \oplus \mathrm{f}(a_2) \oplus \cdots \oplus \mathrm{f}(a_d)$
- Multiplications: $\mathrm{c} = a \times b$ in three steps

Masked operations [ISW03]

- Linear operations: $\mathrm{f(a)} = \mathrm{f}(a_1) \oplus \mathrm{f}(a_2) \oplus \cdots \oplus \mathrm{f}(a_d)$
- Multiplications: $\mathrm{c} = a \times b$ in three steps

$$\begin{bmatrix} a_1 b_1 & a_1 b_2 & a_1 b_3 \\ a_2 b_1 & a_2 b_2 & a_2 b_3 \\ a_3 b_1 & a_3 b_2 & a_3 b_3 \end{bmatrix}$$

partial products

Masked operations [ISW03]

- Linear operations: $\mathrm{f(a)} = \mathrm{f}(a_1) \oplus \mathrm{f}(a_2) \oplus \cdots \oplus \mathrm{f}(a_d)$

- Multiplications: $\mathrm{c} = a \times b$ in three steps

$$\begin{bmatrix} a_1 b_1 & a_1 b_2 & a_1 b_3 \\ a_2 b_1 & a_2 b_2 & a_2 b_3 \\ a_3 b_1 & a_3 b_2 & a_3 b_3 \end{bmatrix} + \begin{bmatrix} 0 & r_1 & r_2 \\ -r_1 & 0 & r_3 \\ -r_2 & -r_3 & 0 \end{bmatrix}$$

partial products   refreshing

Masked operations [ISW03]

- Linear operations: $f(a) = f(a_1) \oplus f(a_2) \oplus \cdots \oplus f(a_d)$
- Multiplications: $c = a \times b$ in three steps

$$\begin{bmatrix} a_1 b_1 & a_1 b_2 & a_1 b_3 \\ a_2 b_1 & a_2 b_2 & a_2 b_3 \\ a_3 b_1 & a_3 b_2 & a_3 b_3 \end{bmatrix} + \begin{bmatrix} 0 & r_1 & r_2 \\ -r_1 & 0 & r_3 \\ -r_2 & -r_3 & 0 \end{bmatrix} \Rightarrow \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

partial products refreshing compression

Masked operations [ISW03]

- Linear operations: $\mathrm{f(a)} = \mathrm{f}(a_1) \oplus \mathrm{f}(a_2) \oplus \cdots \oplus \mathrm{f}(a_d)$

- Multiplications: $\mathrm{c} = a \times b$ in three steps

$$\begin{bmatrix} a_1b_1 & a_1b_2 & a_1b_3 \\ a_2b_1 & a_2b_2 & a_2b_3 \\ a_3b_1 & a_3b_2 & a_3b_3 \end{bmatrix} + \begin{bmatrix} 0 & r_1 & r_2 \\ -r_1 & 0 & r_3 \\ -r_2 & -r_3 & 0 \end{bmatrix} \Rightarrow \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

partial products **refreshing** compression

$a_1b_1 \oplus a_1b_2 \oplus a_1b_3 = \boldsymbol{a_1 b}$ leaks on b

Masked operations [ISW03]

- Linear operations: $\mathrm{f(a)} = \mathrm{f}(a_1) \oplus \mathrm{f}(a_2) \oplus \cdots \oplus \mathrm{f}(a_d)$
- Multiplications: $\mathrm{c} = a \times b$ in three steps

$$\begin{bmatrix} a_1b_1 & a_1b_2 & a_1b_3 \\ a_2b_1 & a_2b_2 & a_2b_3 \\ a_3b_1 & a_3b_2 & a_3b_3 \end{bmatrix} + \begin{bmatrix} 0 & r_1 & r_2 \\ -r_1 & 0 & r_3 \\ -r_2 & -r_3 & 0 \end{bmatrix} \Rightarrow \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

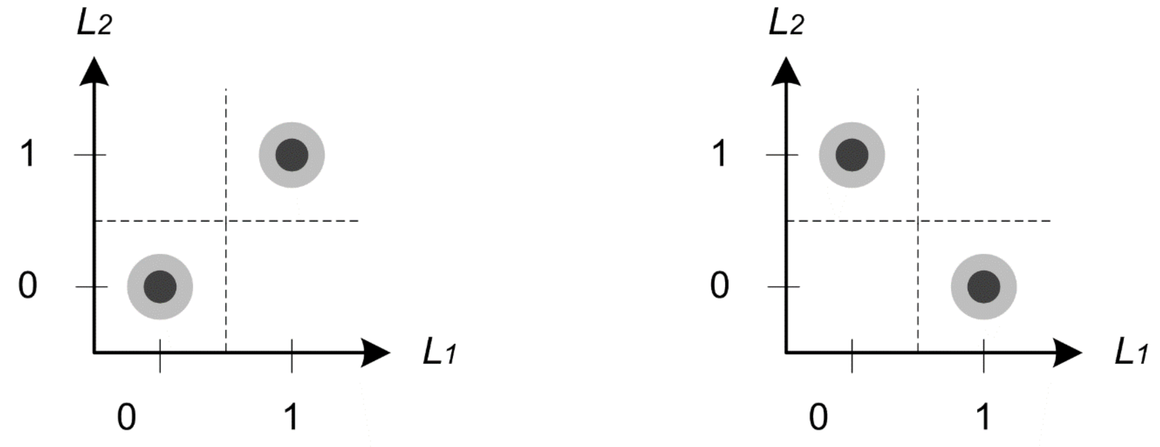
partial products **refreshing** compression

$a_1b_1 \oplus a_1b_2 \oplus a_1b_3 = \boldsymbol{a_1 b}$ leaks on b

⇒ Quadratic overheads & randomness

- (Many published optimizations [R+15,Be+16,GM18])

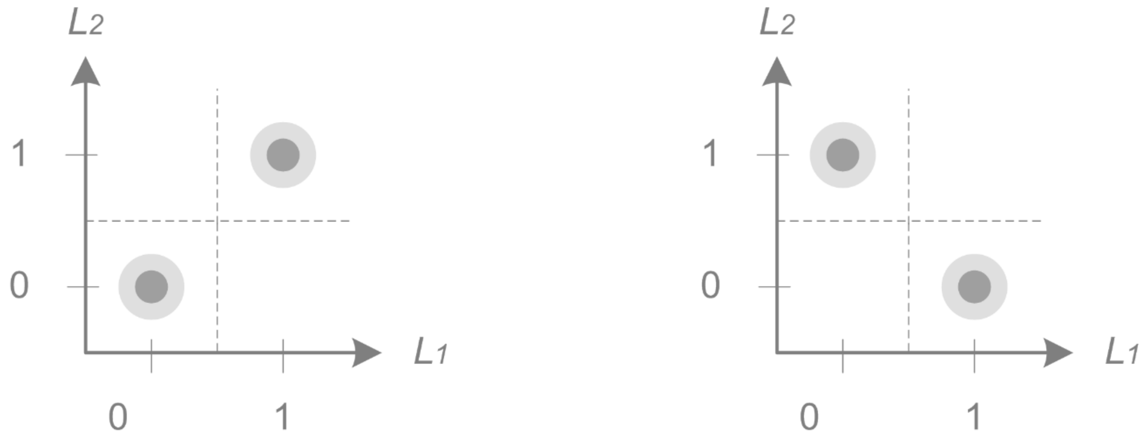
Statistical intuition (2 shares)



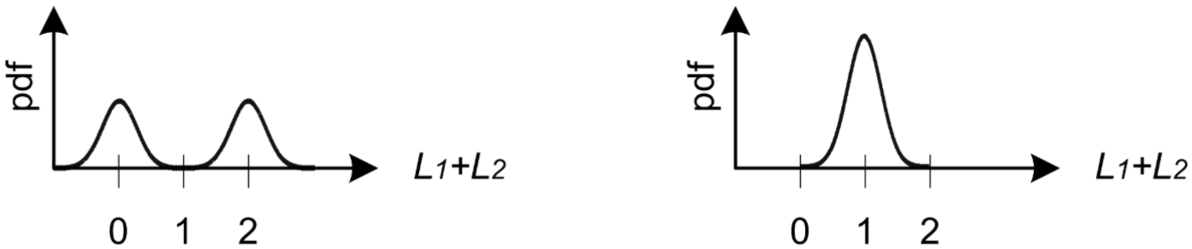
(a) $Y = 0$, serial. (b) $Y = 1$, serial.

- Leakage mean vector for $Y = 0,1 = [0.5\ 0.5]$

Statistical intuition (2 shares)



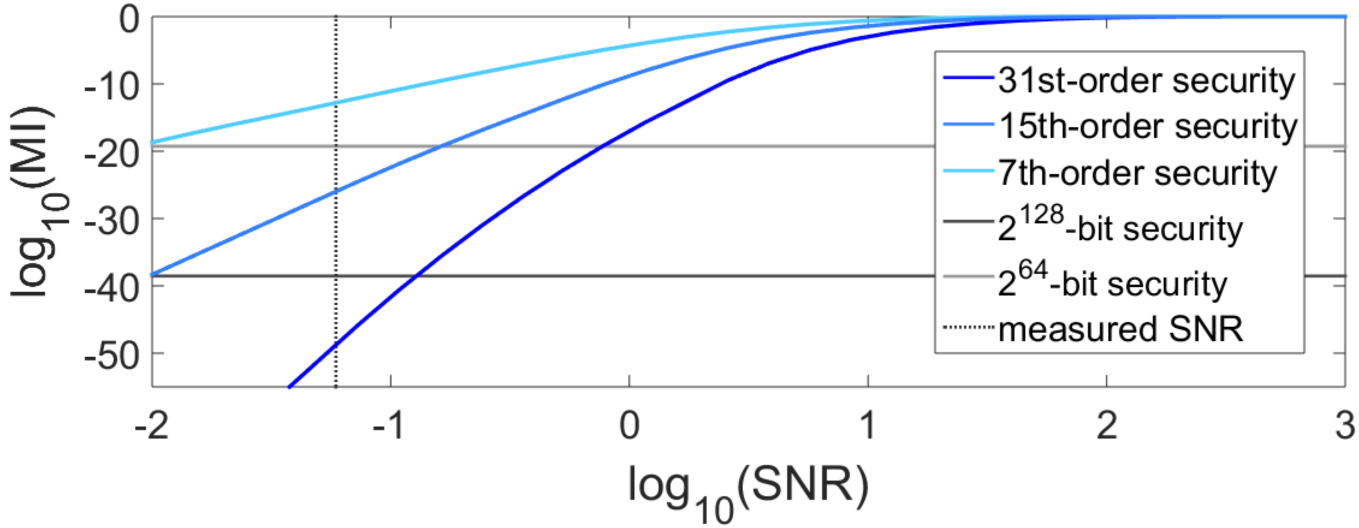
(a) $Y = 0$, serial. (b) $Y = 1$, serial.



(c) $Y = 0$, parallel. (d) $Y = 1$, parallel.

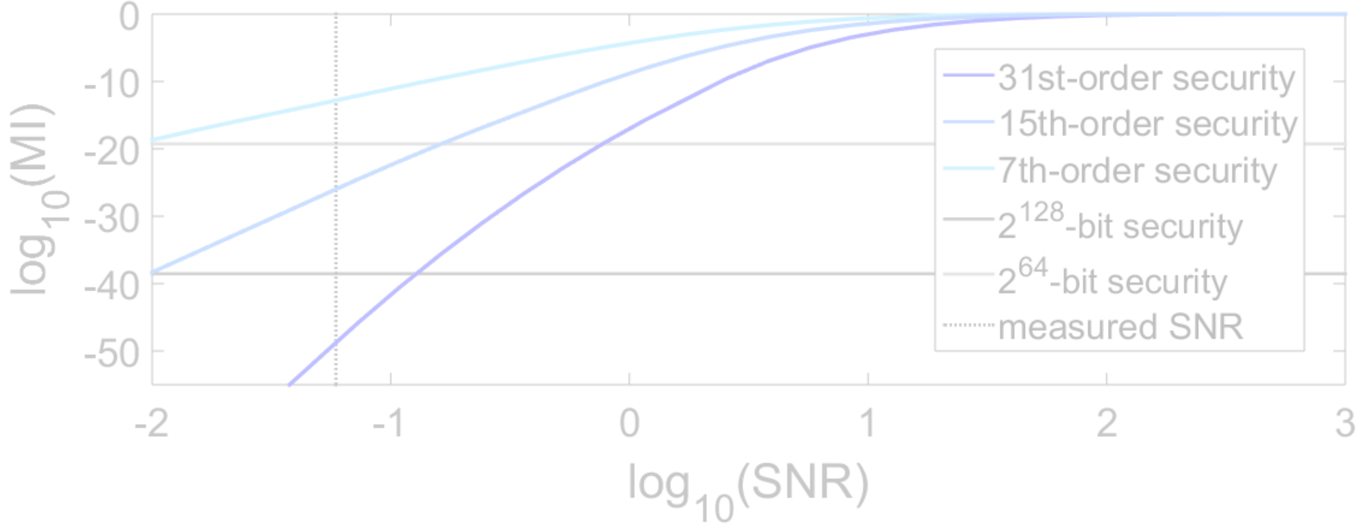
- Leakage mean value for $Y = 0,1 = 1$

Case study: ARM Cortex M4 [JS17]

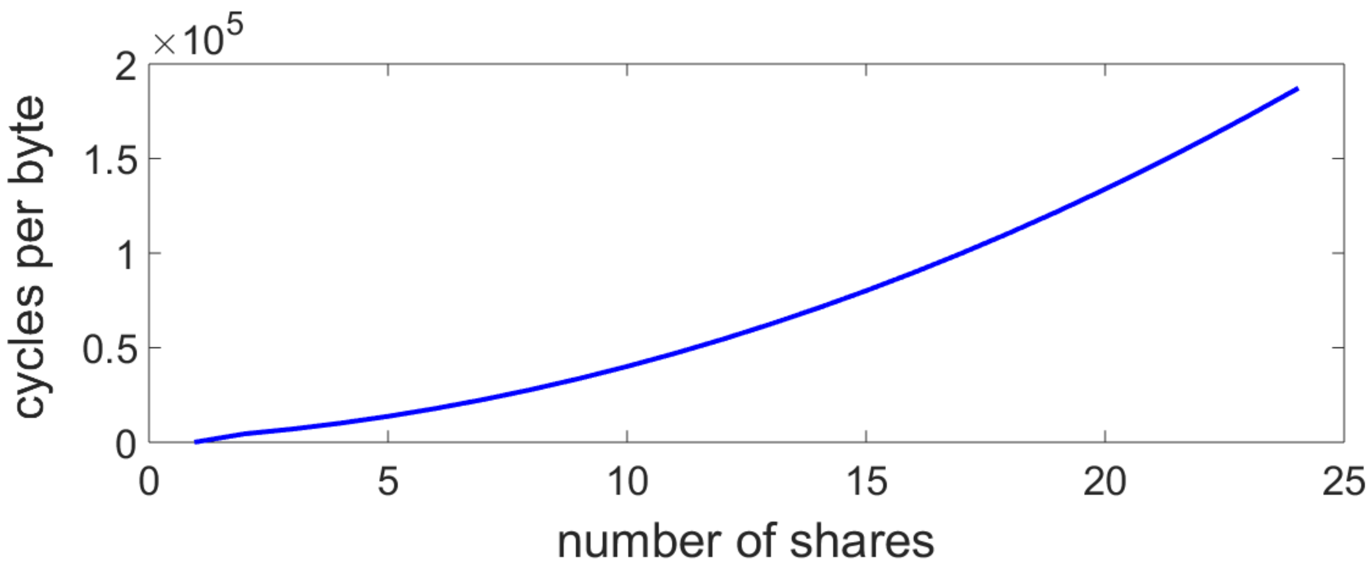


security

Case study: ARM Cortex M4 [JS17]

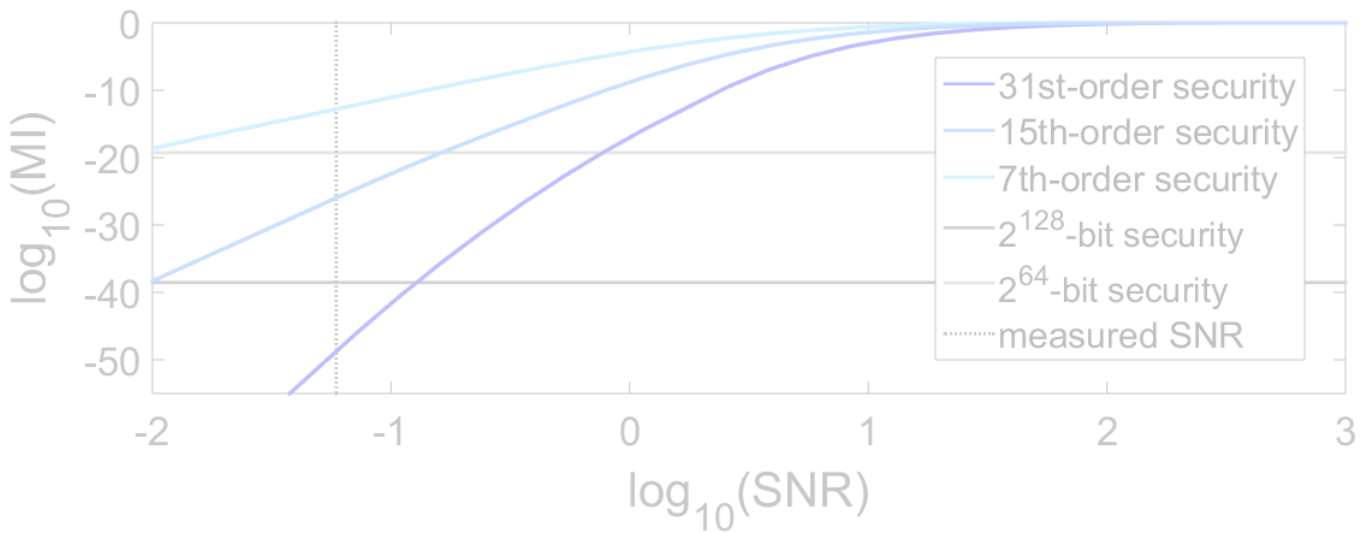


security

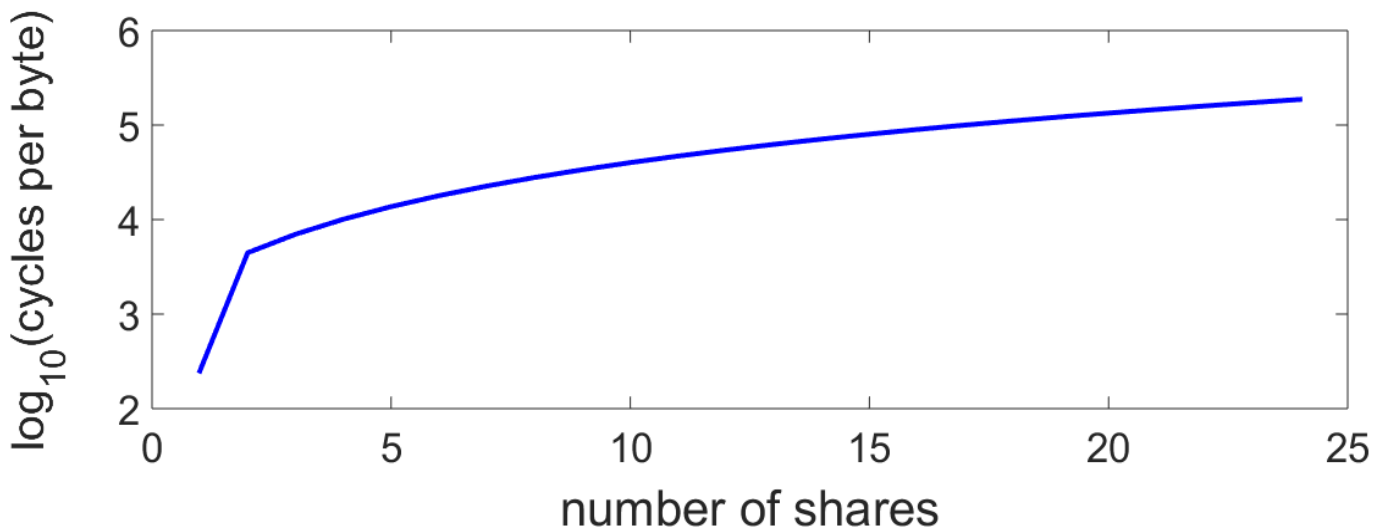


performance

Case study: ARM Cortex M4 [JS17]



security



performance

Summarizing

- Sounds easy but implementation is complex

Summarizing

- Sounds easy but implementation is complex
  - ***Independence issue***: physical defaults (e.g., glitches) can re-combine shares (e.g., [MPG05,NRS11,F+18])
  - Security against horizontal attacks require more ***noise/randomness*** as d increases [BCPZ16,CS19]
  - Scalability/***composition*** are challenging [Ba+15,Ba+16]

Summarizing

- Sounds easy but implementation is complex
  - ***Independence issue***: physical defaults (e.g., glitches) can re-combine shares (e.g., [MPG05,NRS11,F+18])
  - Security against horizontal attacks require more ***noise/randomness*** as d increases [BCPZ16,CS19]
  - Scalability/***composition*** are challenging [Ba+15,Ba+16]

⇒ High security against DPA can be reached but

- It implies large performance overheads
  - E.g., industry currently uses 2-4 shares (?)
- It « only » protects the key (plaintexts are not shared)

Summarizing

- Sounds easy but implementation is complex
  - ***Independence issue***: physical defaults (e.g., glitches) can re-combine shares (e.g., [MPG05,NRS11,F+18])
  - Security against horizontal attacks require more ***noise/randomness*** as d increases [BCPZ16,CS19]
  - Scalability/***composition*** are challenging [Ba+15,Ba+16]

⇒ High security against DPA can be reached but

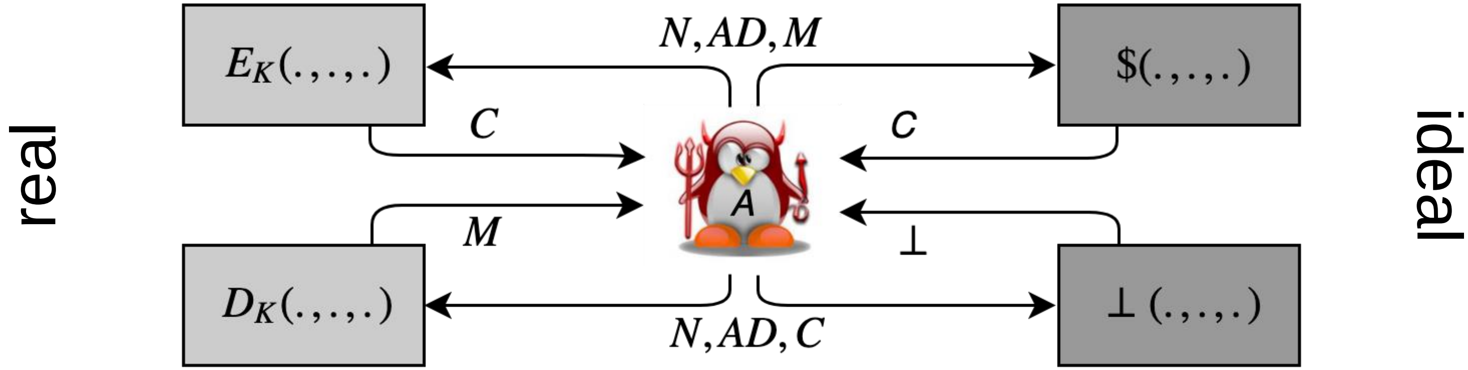
- It implies large performance overheads
  - E.g., industry currently uses 2-4 shares (?)
- It « only » protects the key (plaintexts are not shared)

- SPA security expected to be (much) cheaper

Outline

1. Side-channel **(crypt)analysis**: attacks taxonomy
2. Masking **countermeasure**: security vs. cost
3. Security **definitions** (authenticated encryption)
   a. Nonce-respecting setting (i.e., AEL)
   b. Nonce-misuse setting (i.e., AEmL)
1. Leakage-resistant AE **designs** (& implementations)
   - Level 0: no mode-level leakage-resistance
   - Level 1: re-keyed modes (including sponges)
   - Level 2: level 1 + strengthened init./final.
   - Level 3: level 2 + two-passes
2. **Conclusions** (& the need of open evaluations)

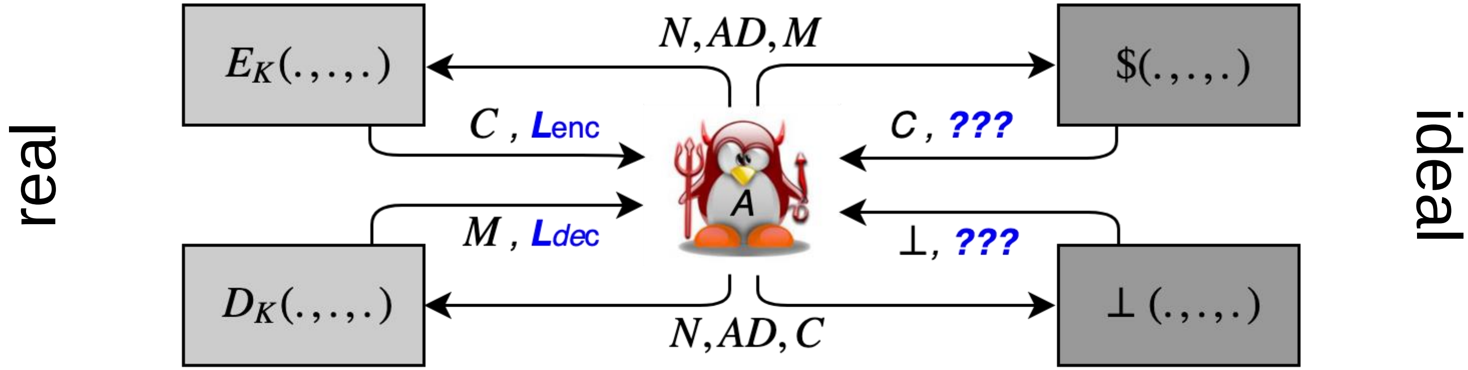
- Why not extending [RS06]'s *all in one* definition?



- A cannot ask a decryption query on (N, AD, C) after C is returned by an $(N, AD, .)$ encryption query

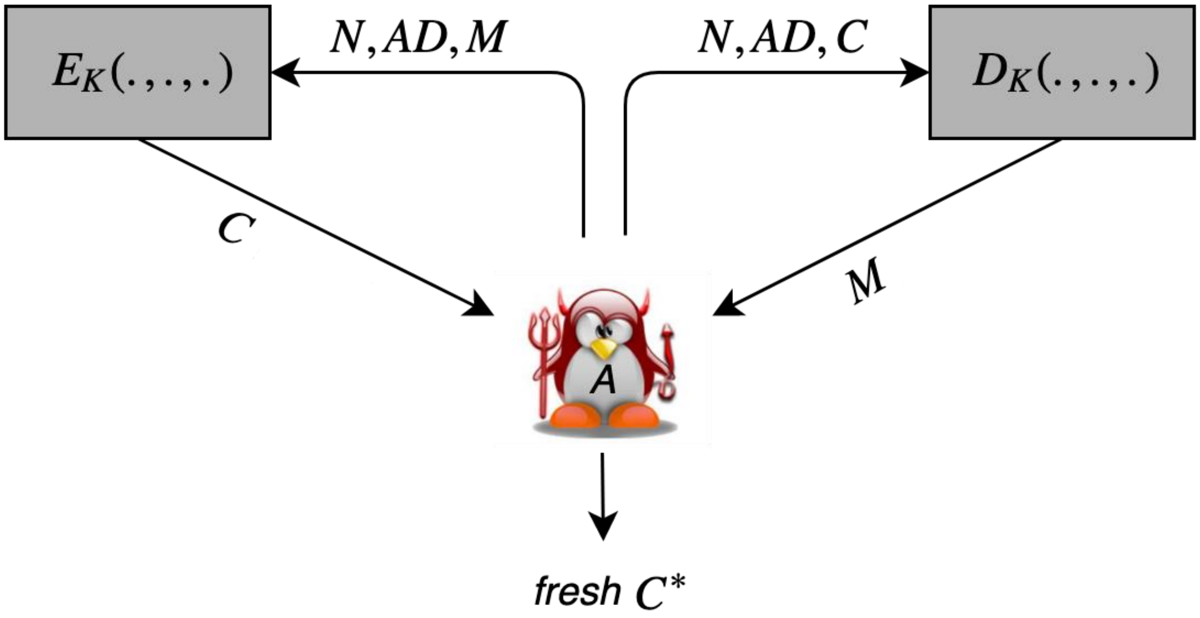
Authenticated Encryption (AEAD)

- Why not extending [RS06]'s *all in one* definition?

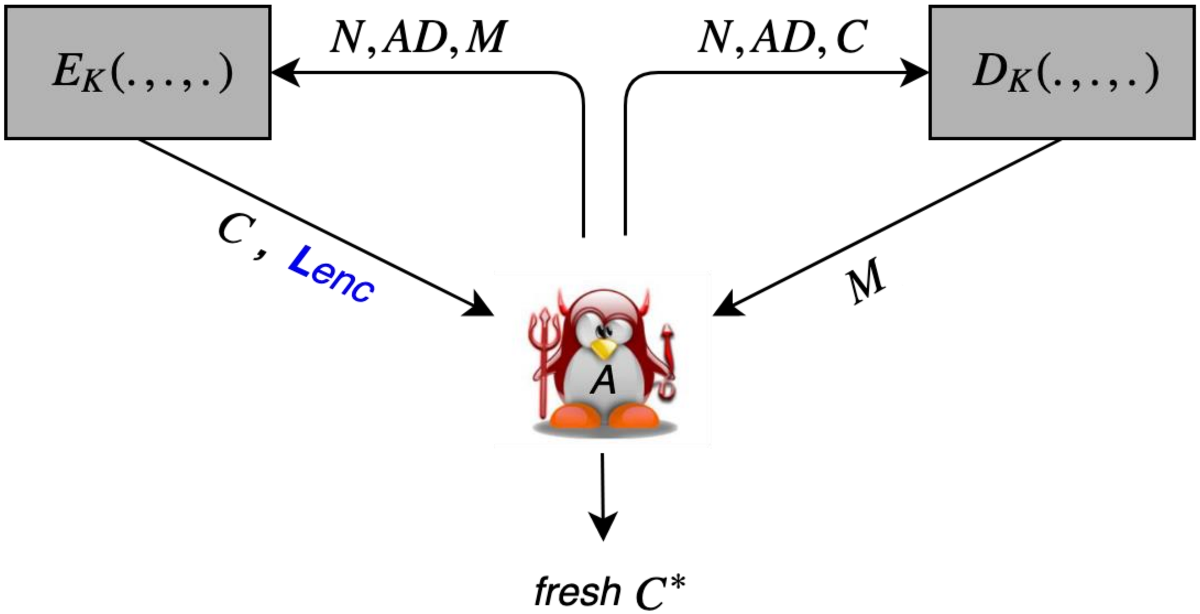


- A cannot ask a decryption query on (N, AD, C) after C is returned by an $(N, AD, .)$ encryption query
- **Problem: the leakage of ideal objects** (which do not have implementations) **seems difficult to define**

Ciphertext Integrity

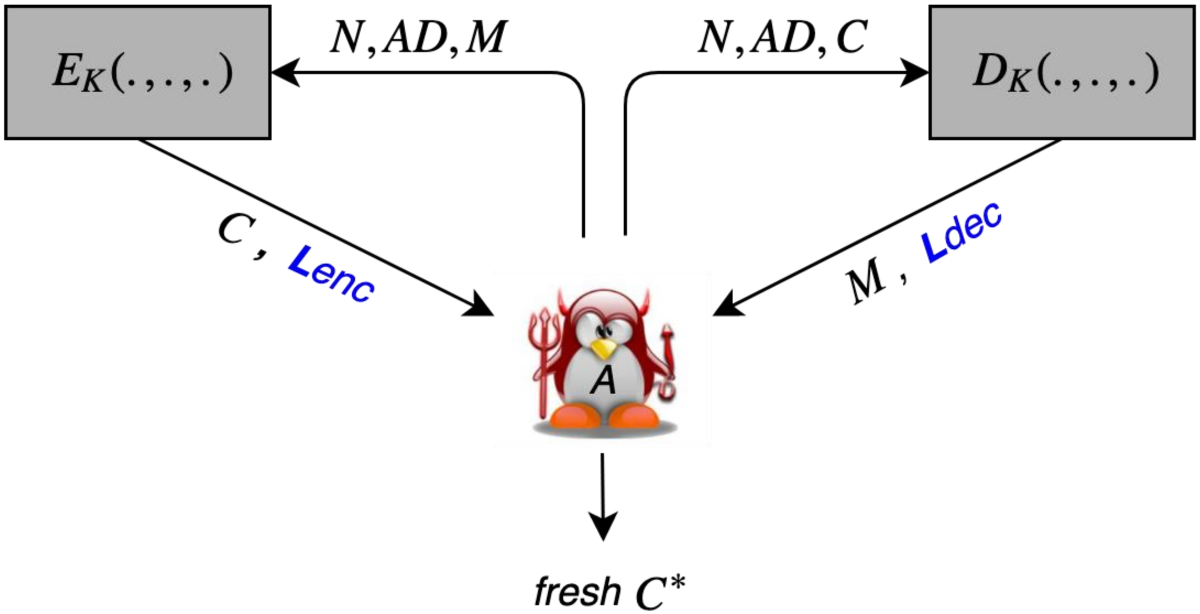


Ciphertext Integrity with Leakage



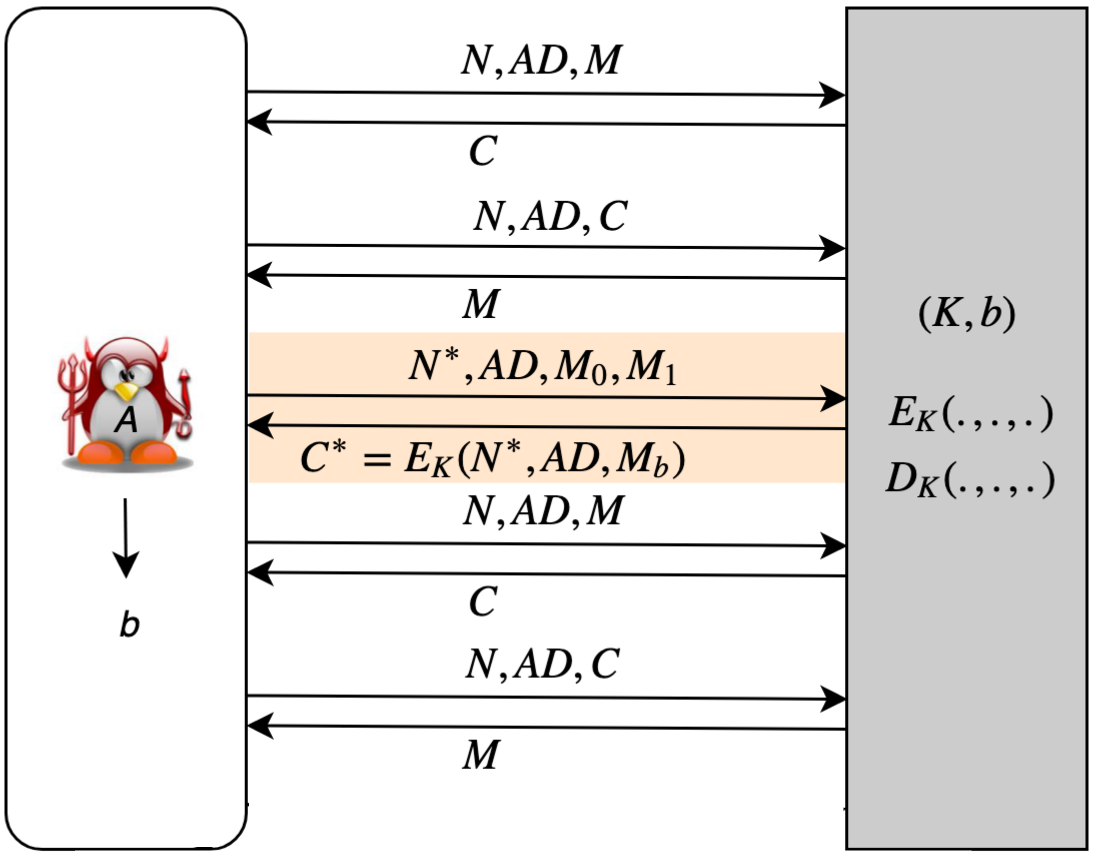
- CIL1: leagage in encryption only [Be+18]

Ciphertext Integrity with Leakage

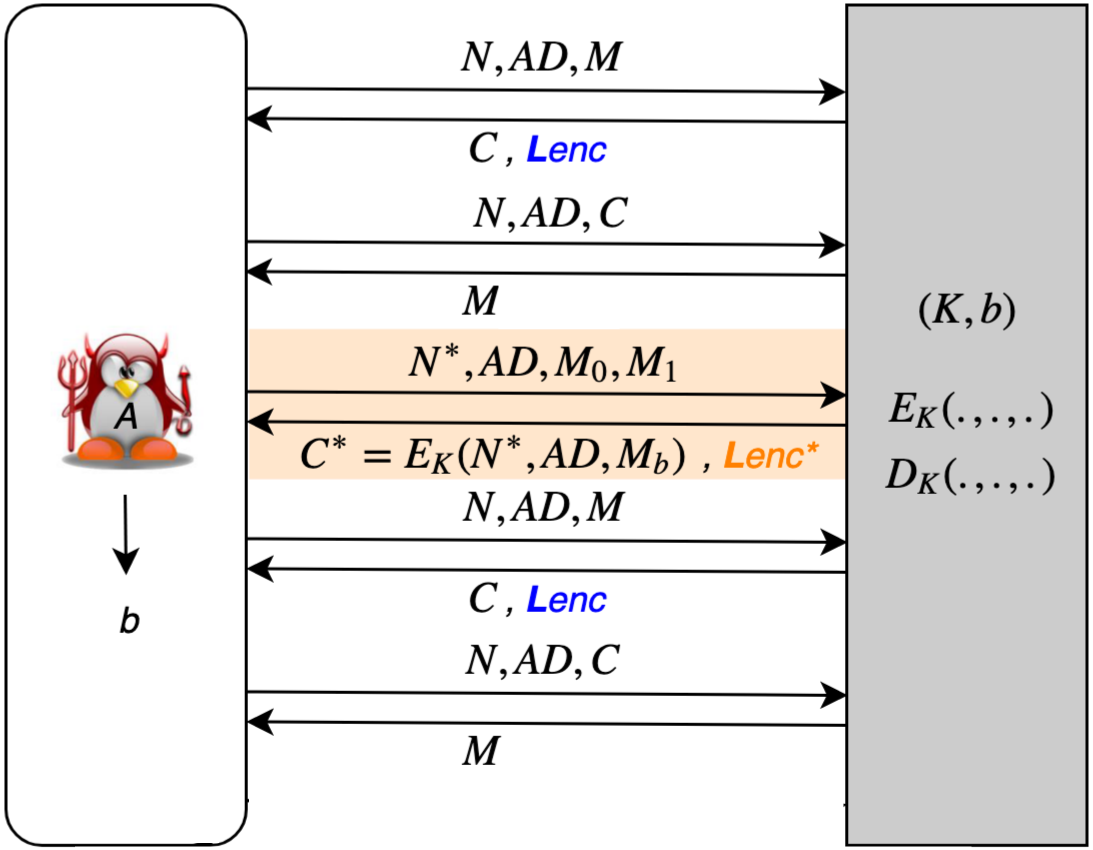


- CIL2: leagage in encryption and decryption [BPPS17]
- Natural extensions (no definitional challenges) with many applications (e.g., secure bootloading)

Chosen Ciphertext Security

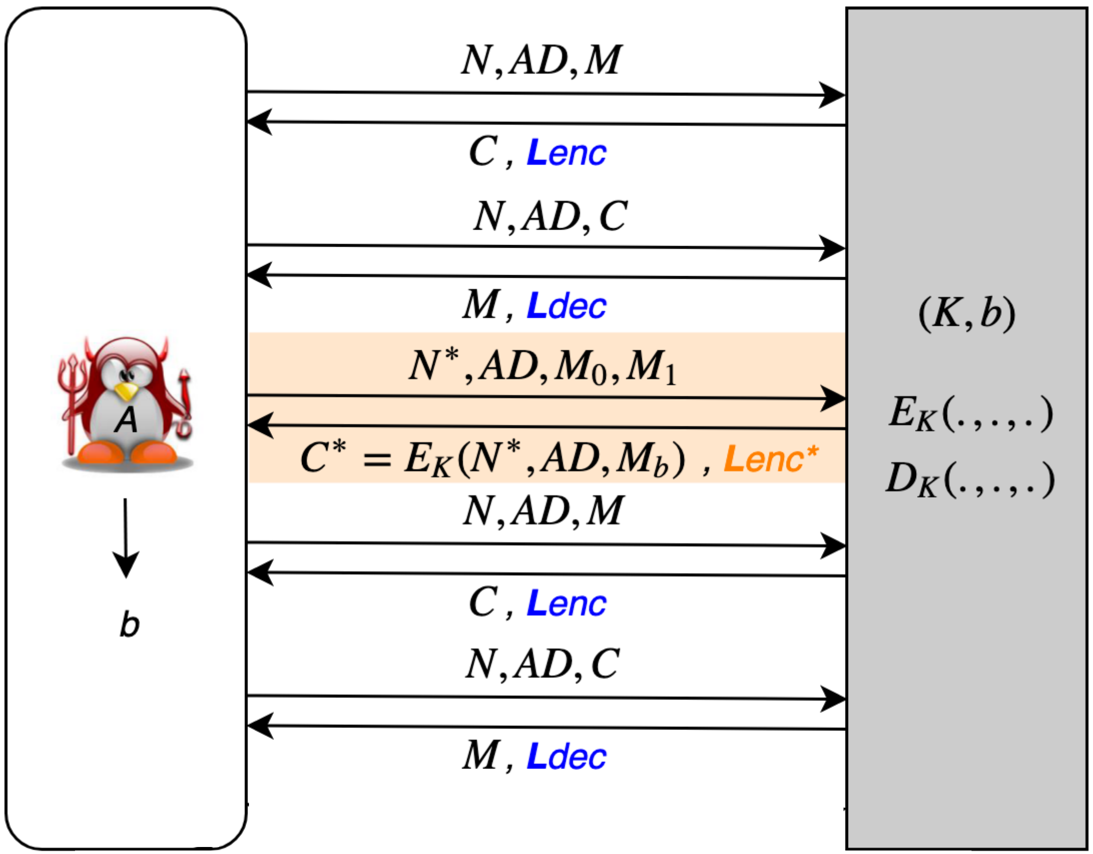


CCA Security with Leakage [GPPS18]



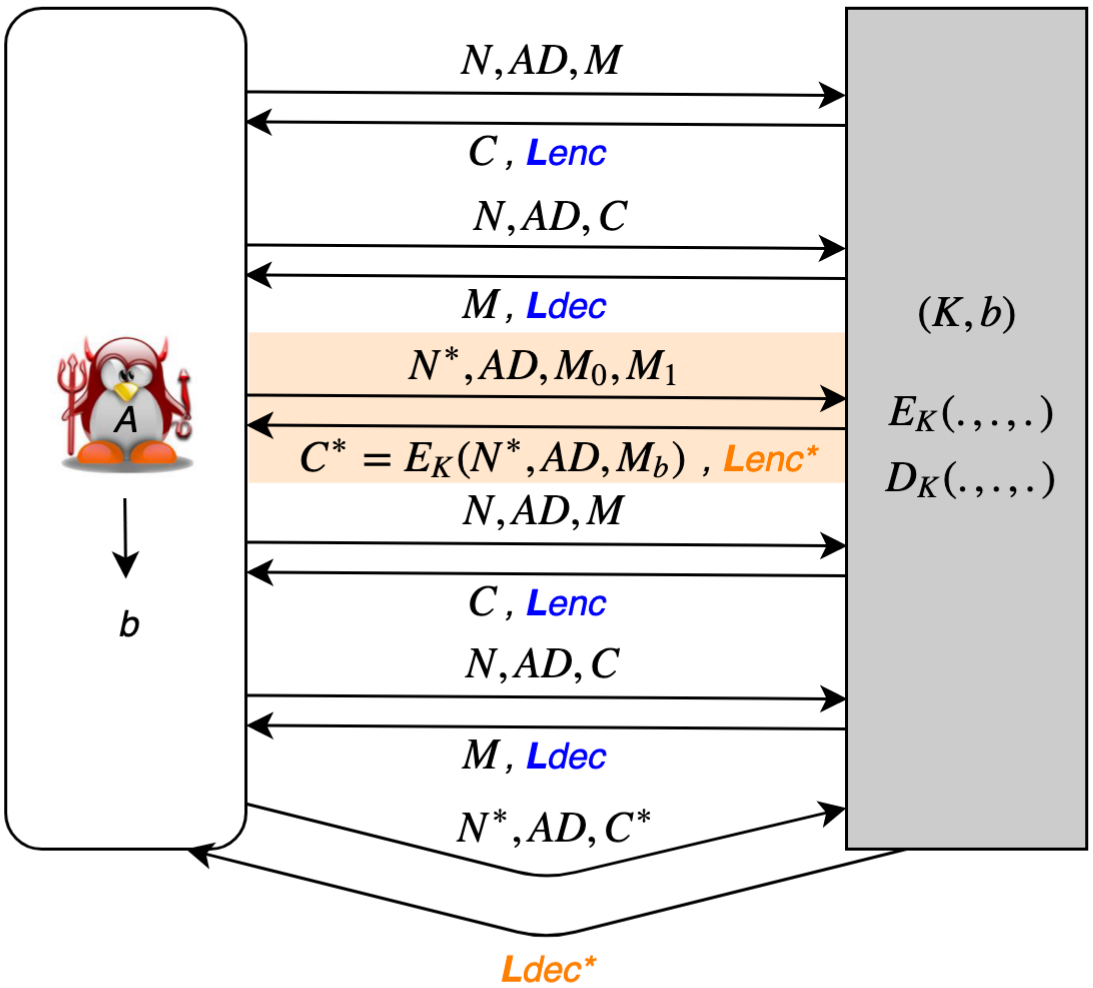
- CCAL1: leakage in encryption

CCA Security with Leakage [GPPS18]



- CCAL2: leakage in encryption and decryption

CCA Security with Leakage [GPPS18]



- + challenge ***Ldec**** (applications: IP protection, …)

The challenge leakage controversy (I)

- [MR04] (and [NS09,BG10,…]): indistinguishability with *Lenc** is hard (one bit breaks it with $p = 1$)
  - So it is quite tempting to ignore it 
  - Which can make sense (e.g., if you tolerate « local attacks » but not « global » security degradations)
    - **Leakage-resilience** vs. **leakage-resistance**

The challenge leakage controversy (I)

- [MR04] (and [NS09,BG10,…]): indistinguishability with *Lenc** is hard (one bit breaks it with $p = 1$)
  - So it is quite tempting to ignore it 
  - Which can make sense (e.g., if you tolerate « local attacks » but not « global » security degradations)
    - **Leakage-resilience** vs. **leakage-resistance**

- Ignoring challenge leakages means that an implementation leaking messages in full is OK
  - This is not what we want in general / theory
  - It can have big impact (e.g., TLS [CHV03],[AP13], …)
    - *Different attacks but they show plaintext leakage matters*

The challenge leakage controversy (II) 3.5

- If we do not make it part of the definition it will never be a goal for cryptographers & engineers
  - Cryptographers: minimize the message manipulation
  - Engineers: minimize message leakage, e.g., with special encodings (which is not much studied yet)

The challenge leakage controversy (II)

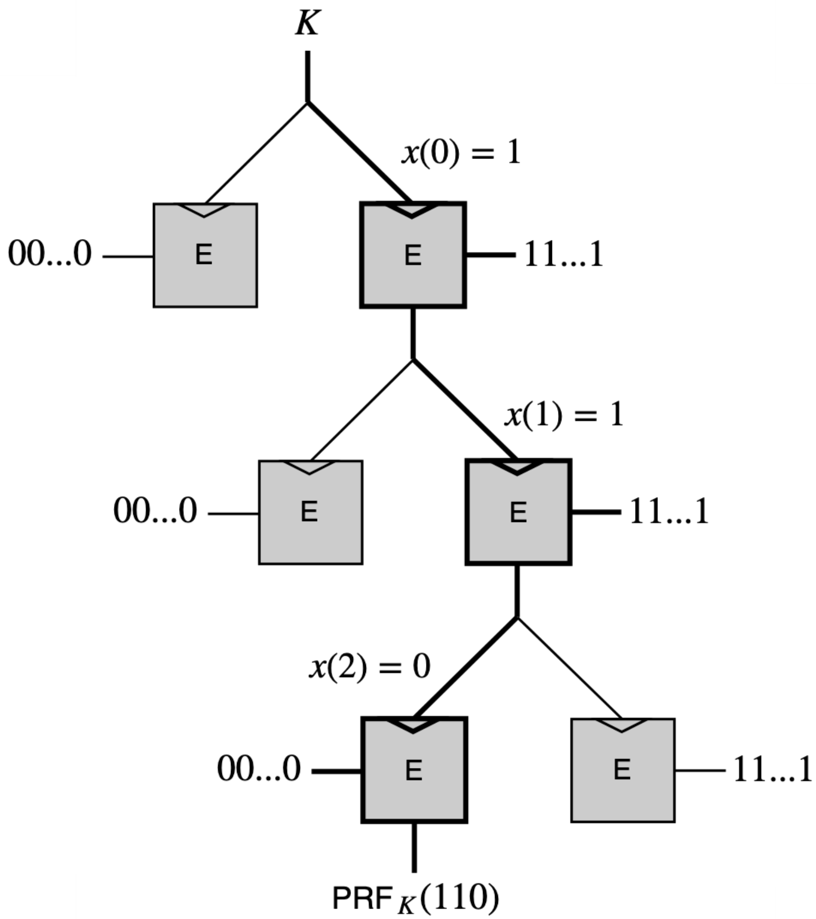
- If we do not make it part of the definition it will never be a goal for cryptographers & engineers
  - Cryptographers: minimize the message manipulation
  - Engineers: minimize message leakage, e.g., with special encodings (which is not much studied yet)
- We need to understand what can be achieved
  - *Even if results are not ideal* (e.g., no negl. Adv.)

The challenge leakage controversy (II)

- If we do not make it part of the definition it will never be a goal for cryptographers & engineers
  - Cryptographers: minimize the message manipulation
  - Engineers: minimize message leakage, e.g., with special encodings (which is not much studied yet)
- We need to understand what can be achieved
  - *Even if results are not ideal* (e.g., no negl. Adv.)
- Technically: more greyscale view than [MR04]
  - Challenge leakages allow Message Comparison (MC) attacks which are not always tivial, e.g.,
    - Remote timing attacks: scalar leakages (vs. vectors)
    - Proxy re-encryption: messages are not chosen

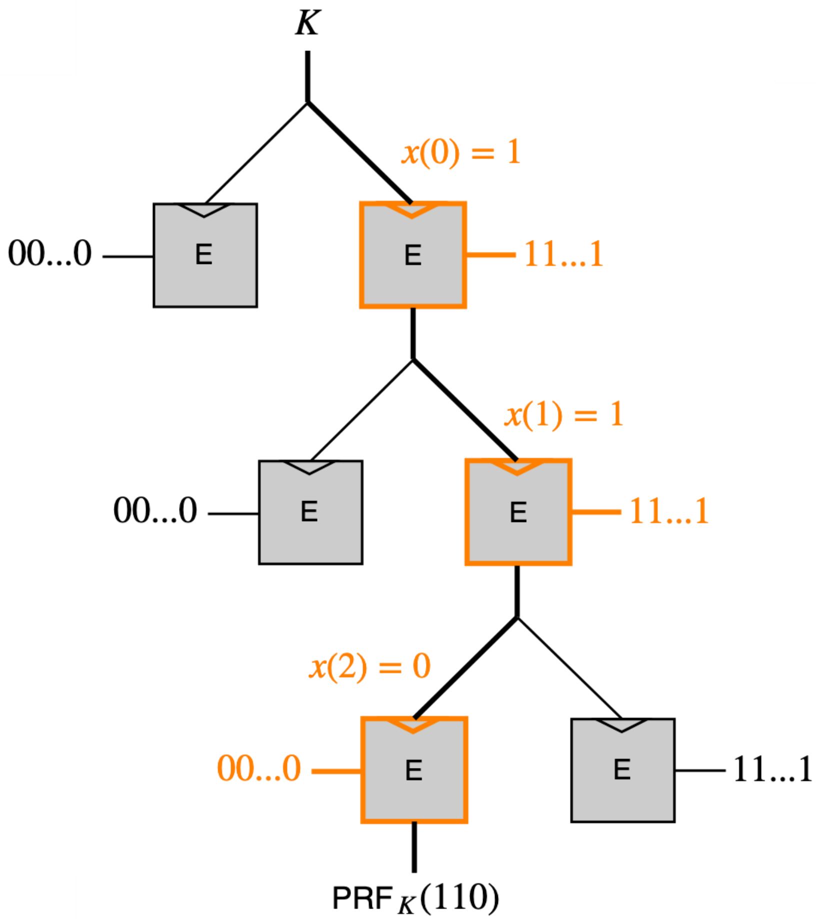
An motivating example

- Tree-based leakage-resilient PRF [GGM84,FPS12]



An motivating example

- Tree-based leakage-resilient PRF [GGM84,FPS12]



- Leads to simple MC attacks
  - Message encrypted bit per bit ⇒ no algorithmic noise
  - Constant block cipher inputs « all zeros » and « all ones » easy to distinguish with HWs [B12]
- (Yet is quite good against KR)

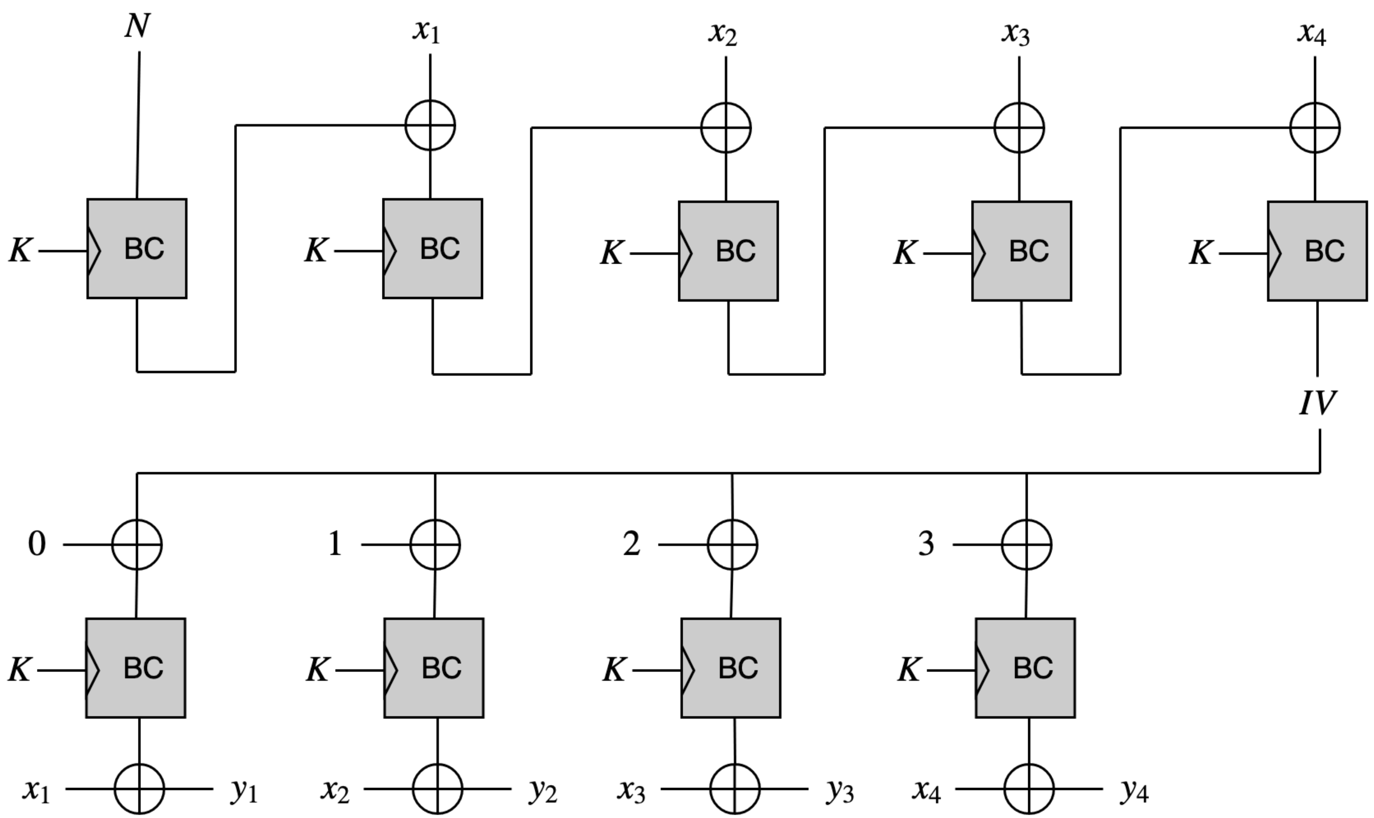
Outline

1. Side-channel **(crypt)analysis**: attacks taxonomy
2. Masking **countermeasure**: security vs. cost
3. Security **definitions** (authenticated encryption)
   a. Nonce-respecting setting (i.e., AEL)
   b. Nonce-misuse setting (i.e., AEmL)

1. Leakage-resistant AE **designs** (& implementations)
   - Level 0: no mode-level leakage-resistance
   - Level 1: re-keyed modes (including sponges)
   - Level 2: level 1 + strengthened init./final.
   - Level 3: level 2 + two-passes

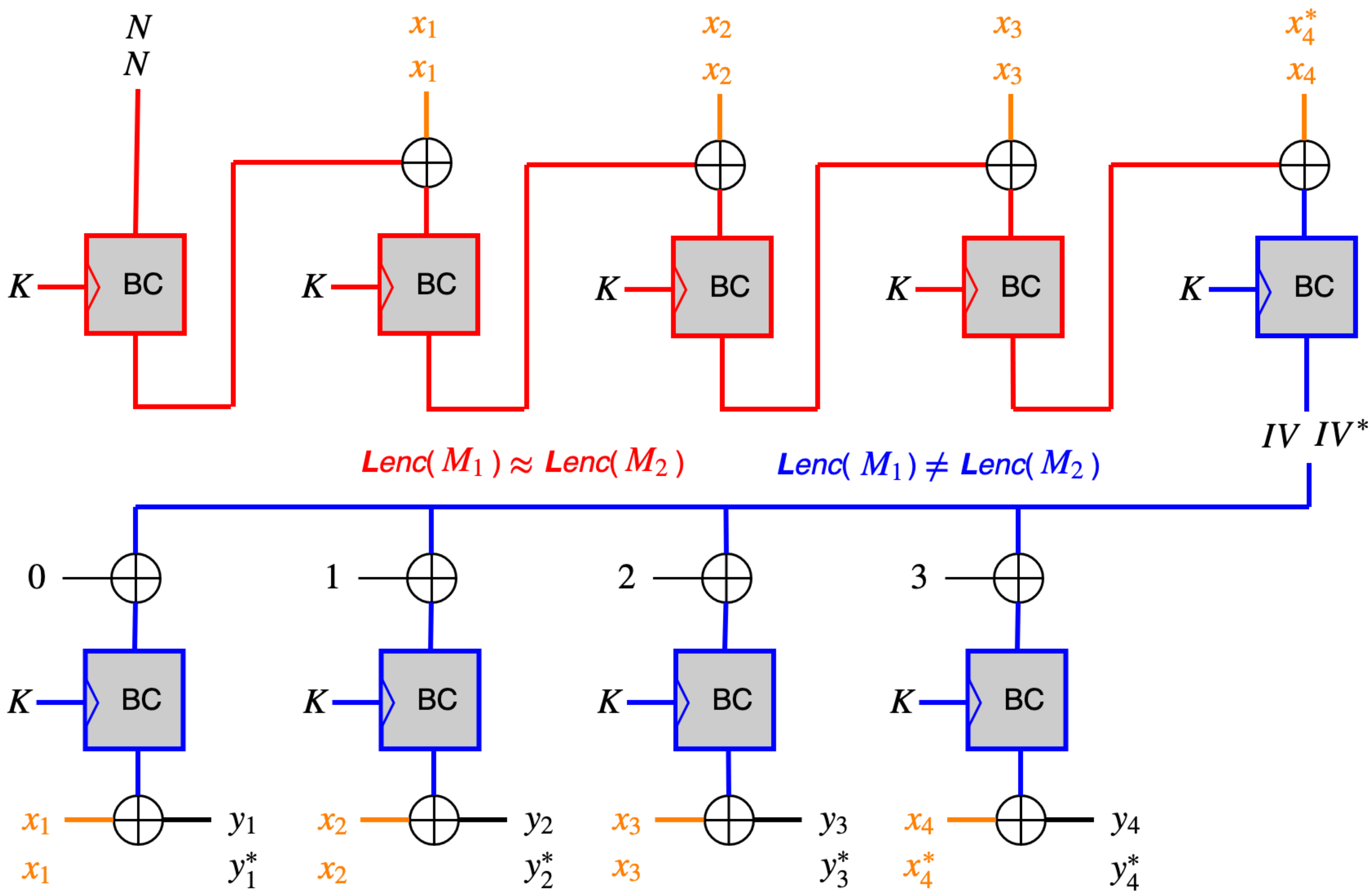
2. **Conclusions** (& the need of open evaluations)

Misuse-Resistance (MR) [RS06]



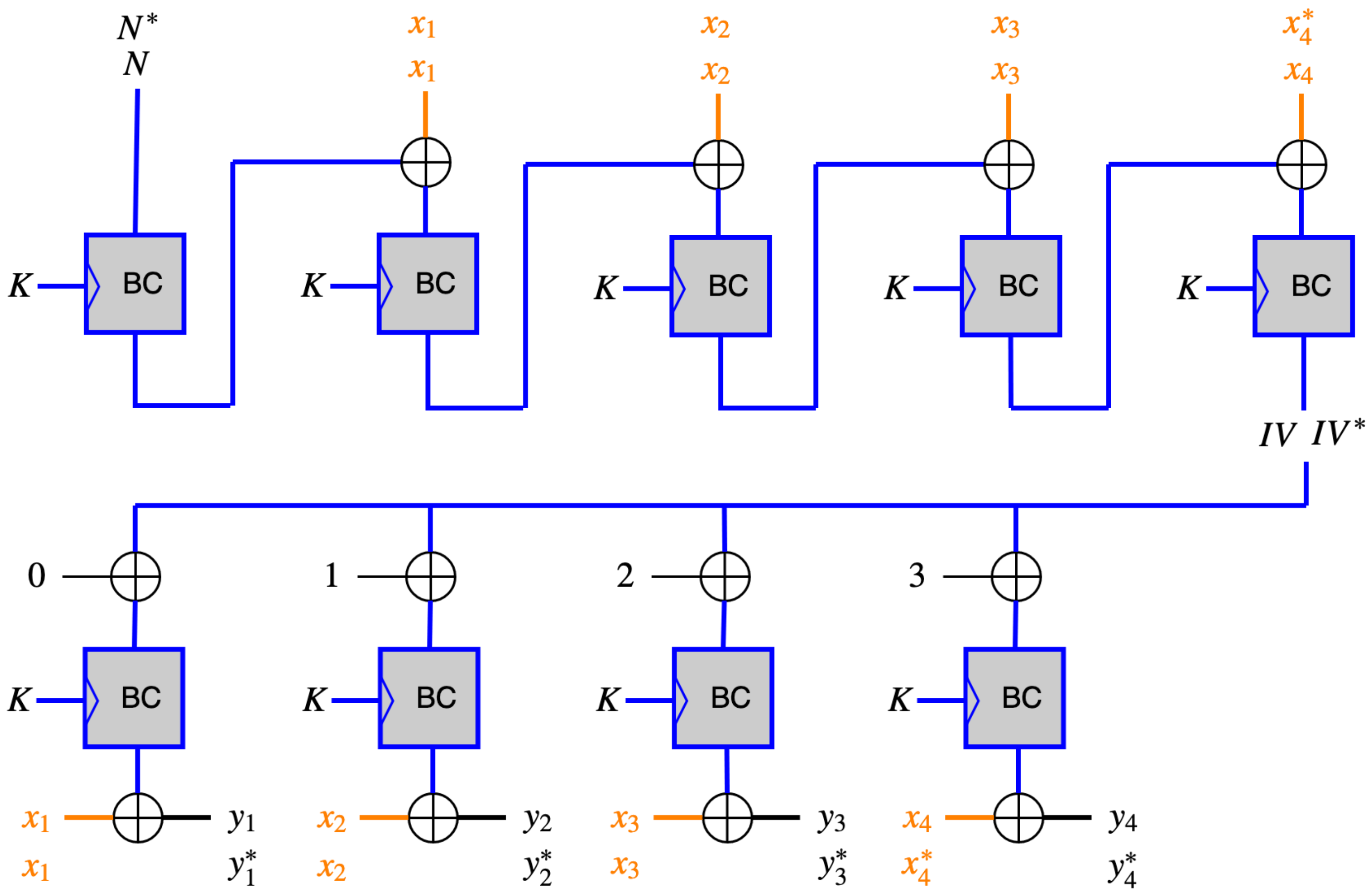
- Black box: only identical (N, M) pairs should be at risk
- Typically achieved by having a 2-pass mode (e.g., SIV)

Misuse-Resistance (MR) [RS06]



- With leakage: a SC attack against $M_1 = \{x_1, x_2, x_3, x_4\}$ and $M_2 = \{x_1, x_2, x_3, x_4^*\}$ leaks that they first blocks are equal

Misuse-Resilience (mR) [ADL17]



- Fresh challenge nonce circumvent this impossibility
  - Intuition: leaves mostly MC attacks and DPAs

Summarizing

- For confidentiality, no meaningful encryption scheme seem to ensure leakage-resistance and (nonce) misuse-resistance (excluding trivial / leak-free solutions)

Summarizing

- For confidentiality, no meaningful encryption scheme seem to ensure leakage-resistance and (nonce) misuse-resistance (excluding trivial / leak-free solutions)
- Natural combinations include:
  a. Misuse-resilience/leakage-resistance: CCAmL [GPPS18]
  b. Misuse-resistance/leakage-resilience: CCAMl [BMOS17]

Summarizing

- For confidentiality, no meaningful encryption scheme seem to ensure leakage-resistance and (nonce) misuse-resistance (excluding trivial / leak-free solutions)
- Natural combinations include:
  a. Misuse-resilience/leakage-resistance: CCAmL [GPPS18]
  b. Misuse-resistance/leakage-resilience: CCAMl [BMOS17]
- ≈ a choice between the need for applications to limit the leakage or for implementers to control nonces

Summarizing

- For confidentiality, no meaningful encryption scheme seem to ensure leakage-resistance and (nonce) misuse-resistance (excluding trivial / leak-free solutions)
- Natural combinations include:
  a. Misuse-resilience/leakage-resistance: CCAmL [GPPS18]
  b. Misuse-resistance/leakage-resilience: CCAMl [BMOS17]
- $\approx$ a choice between the need for applications to limit the leakage or for implementers to control nonces
- Strongest def.: AEML=CIML2+CCAmL2+CCAMl2

Summarizing

- For confidentiality, no meaningful encryption scheme seem to ensure leakage-resistance and (nonce) misuse-resistance (excluding trivial / leak-free solutions)
- Natural combinations include:
  a. Misuse-resilience/leakage-resistance: CCAmL [GPPS18]
  b. Misuse-resistance/leakage-resilience: CCAMl [BMOS17]
- ≈ a choice between the need for applications to limit the leakage or for implementers to control nonces
- Strongest def.: AEML=CIML2+CCAmL2+CCAMl2
- Weaker variants can be meaningful: for instance AEmL=CIML2+CCAmL2 [Be+19], CPAl1 [DM19], …

Outline

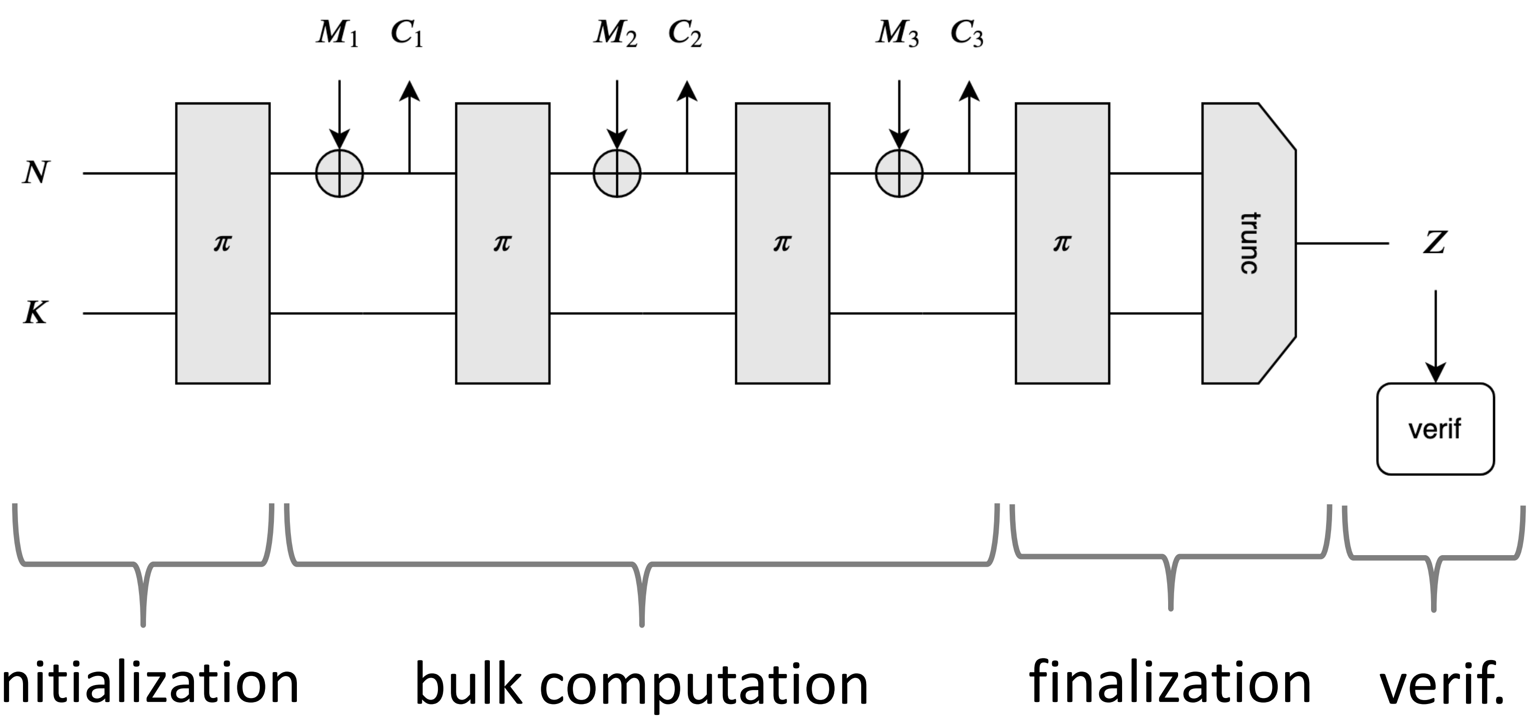
1. Side-channel **(crypt)analysis**: attacks taxonomy
2. Masking **countermeasure**: security vs. cost
3. Security **definitions** (authenticated encryption)
   a. Nonce-respecting setting (i.e., AEL)
   b. Nonce-misuse setting (i.e., AEmL)

1. Leakage-resistant AE **designs** (& implementations)
   - Level 0: no mode-level leakage-resistance
   - Level 1: re-keyed modes (including sponges)
   - Level 2: level 1 + strengthened init./final.
   - Level 3: level 2 + two-passes

2. **Conclusions** (& the need of open evaluations)

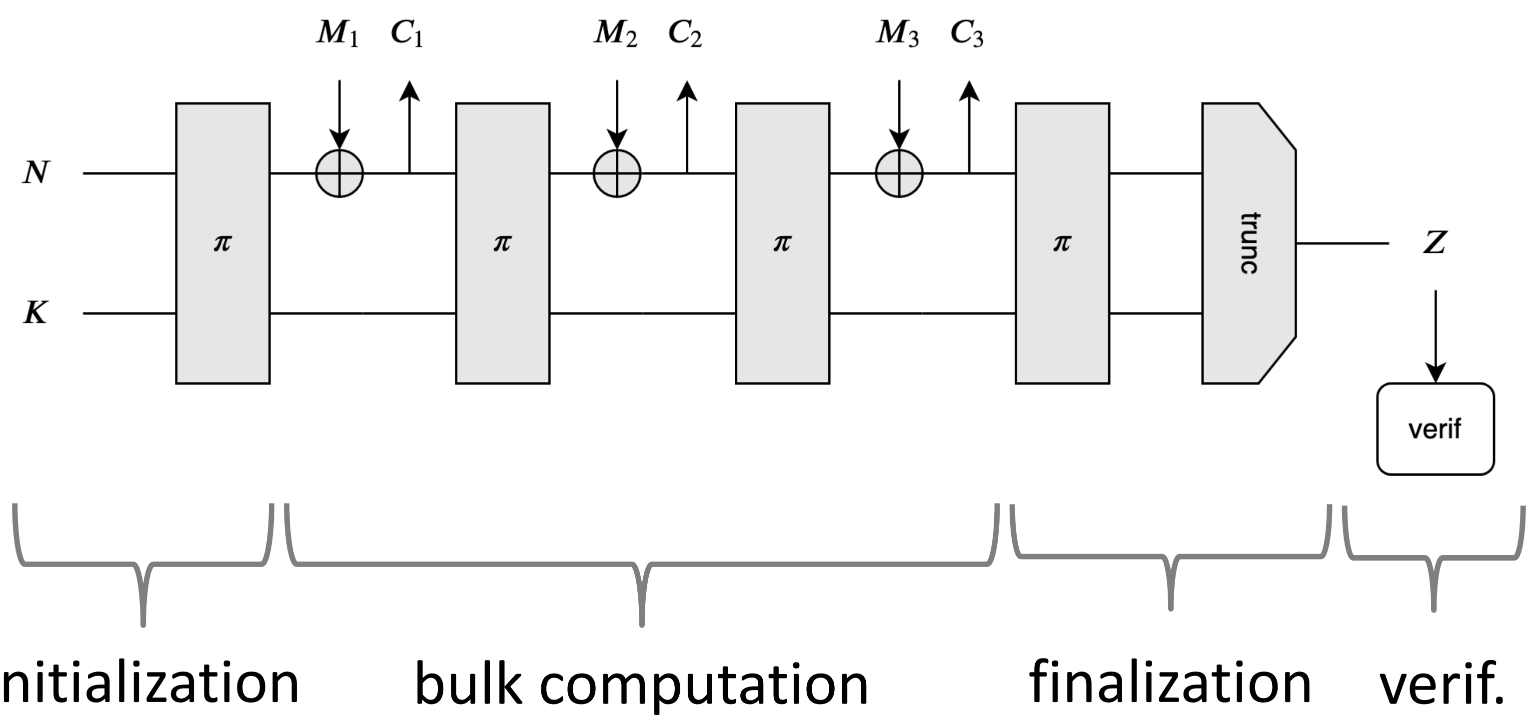
Mode analysis (I)

- Identify main steps, e.g., inner keyed sponge



Mode analysis (I)

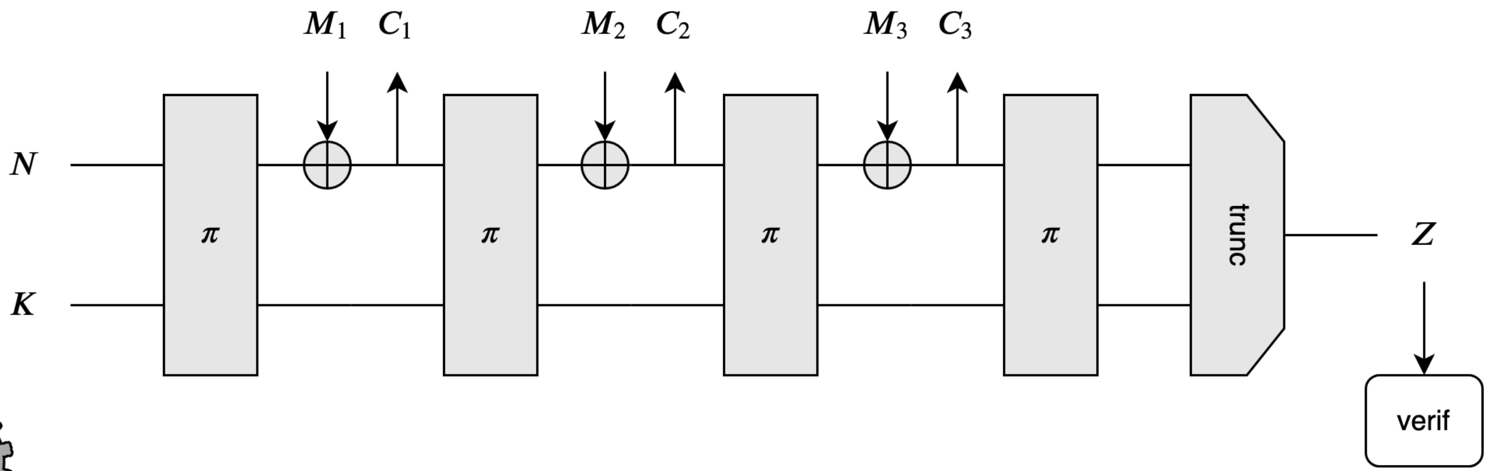
- Identify main steps, e.g., inner keyed sponge

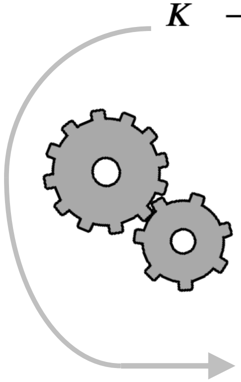


- Choose the target for confidentiality & integrity

Mode analysis (II)

- Reduce the mode to (weak) assumptions (tightly)





only computation leaks

leak-free components   bounded leakage

strong unpredictability with leakage

simulatable leakages   hard-to-invert leakages

oracle-free leakages   [...]

Practical evaluation (I)

- Translate assumptions into necessary design goals

	init./final.	bulk comp.	tag verif.
conf.	DPA (key recovery)	DPA (key recovery) SPA (key recovery) MC	∅
int.	DPA (key recovery)	DPA (key recovery) SPA (key recovery) unbounded leakages	DPA (tag recovery) unbounded leakages

- *Set the target security level (2^m leakages, 2^t time)*
- *Evaluate implementation cost & performances*

Practical evaluation (II)

- Approximate performance overheads

	init./final.	bulk comp.	tag verif.
conf.	x 5 – 10 – >100	x 5 – 10 – >100 x 1 – 5 ???	Ø
int.	x 5 – 10 – >100	x 5 – 10 – >100 x 1 – 5 x 1	x 5 – 10 – >100 x 1 – 5

- DPA security: high-order masking, shuffling, …
- SPA security: parallel implementations, noise, …

Outline

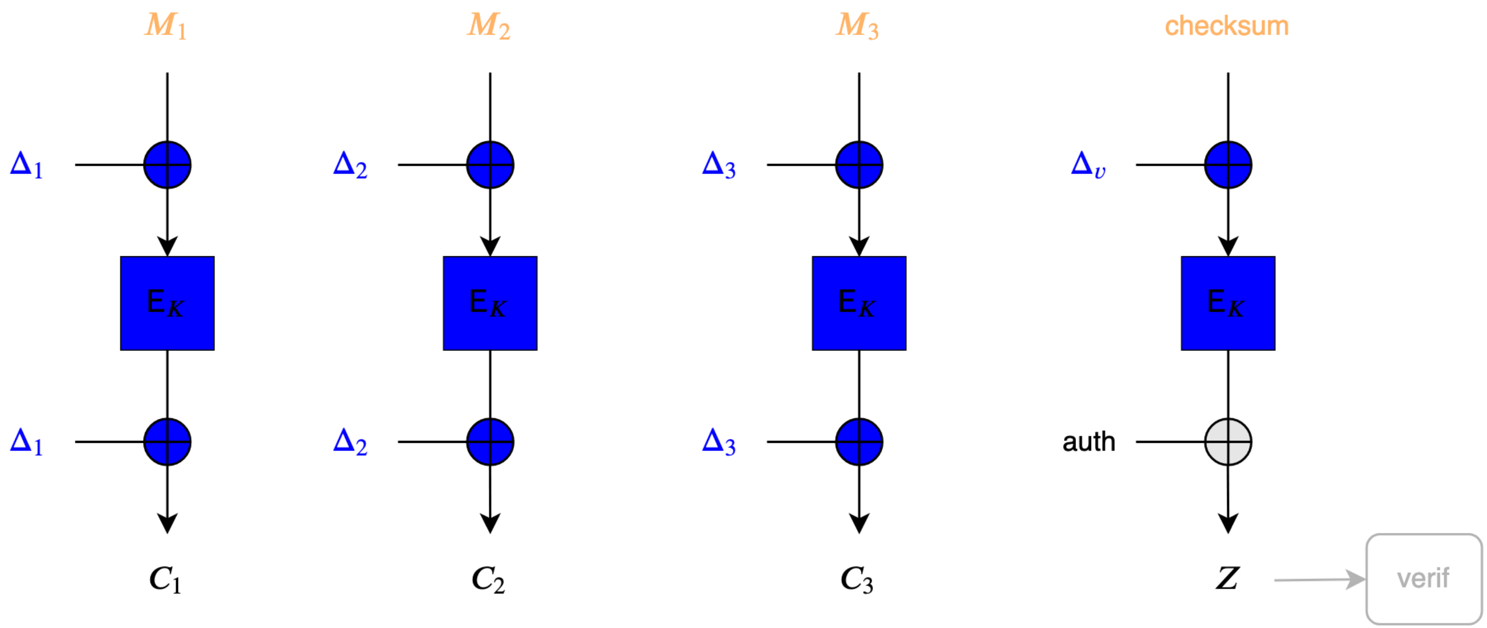
1. Side-channel **(crypt)analysis**: attacks taxonomy
2. Masking **countermeasure**: security vs. cost
3. Security **definitions** (authenticated encryption)
   a. Nonce-respecting setting (i.e., AEL)
   b. Nonce-misuse setting (i.e., AEmL)

1. Leakage-resistant AE **designs** (& implementations)
   - Level 0: no mode-level leakage-resistance
   - Level 1: re-keyed modes (including sponges)
   - Level 2: level 1 + strengthened init./final.
   - Level 3: level 2 + two-passes

2. Conclusions (& the need of open evaluations)

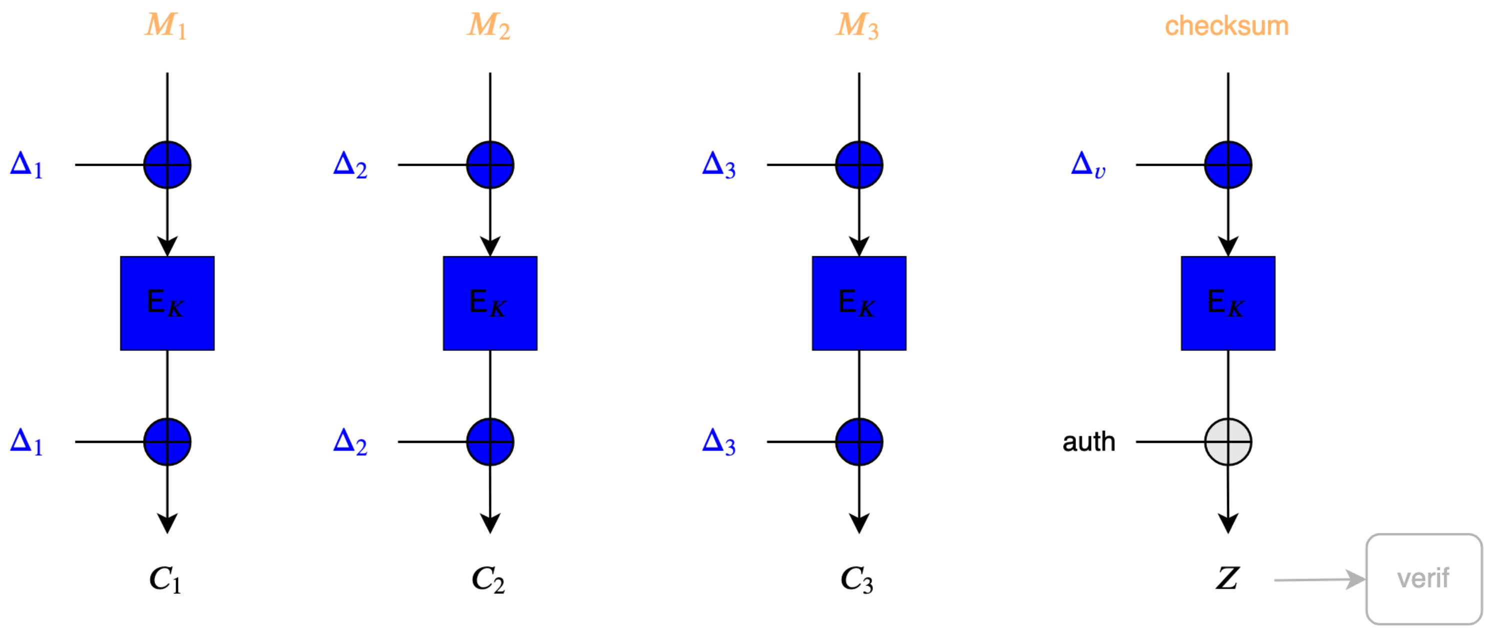
OCB-Pyjamask [G+19]

- Target: CCAL1, CIL1 (L in enc only, no misuse)



- Needs DPA resistance for all E_K blocks
  - Primitive/implementation SCA security only

- Target: CCAL1, CIL1 (L in enc only, no misuse)



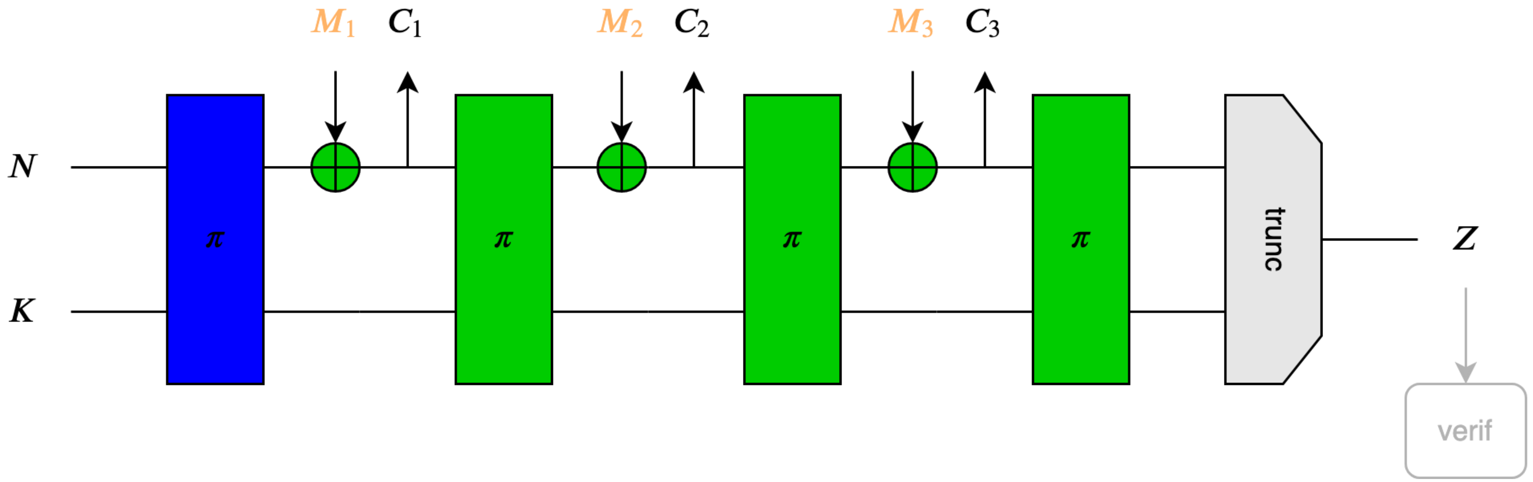
- Needs DPA resistance for all E_K blocks
  - Primitive/implementation SCA security only
- Others: SKINNY-AEAD, SUNDAE-GIFT, OCB-AES, …

Outline

1. Side-channel **(crypt)analysis**: attacks taxonomy
2. Masking **countermeasure**: security vs. cost
3. Security **definitions** (authenticated encryption)
   a. Nonce-respecting setting (i.e., AEL)
   b. Nonce-misuse setting (i.e., AEmL)

1. Leakage-resistant AE **designs** (& implementations)
   - Level 0: no mode-level leakage-resistance
   - Level 1: re-keyed modes (including sponges)
   - Level 2: level 1 + strengthened init./final.
   - Level 3: level 2 + two-passes
2. **Conclusions** (& the need of open evaluations)

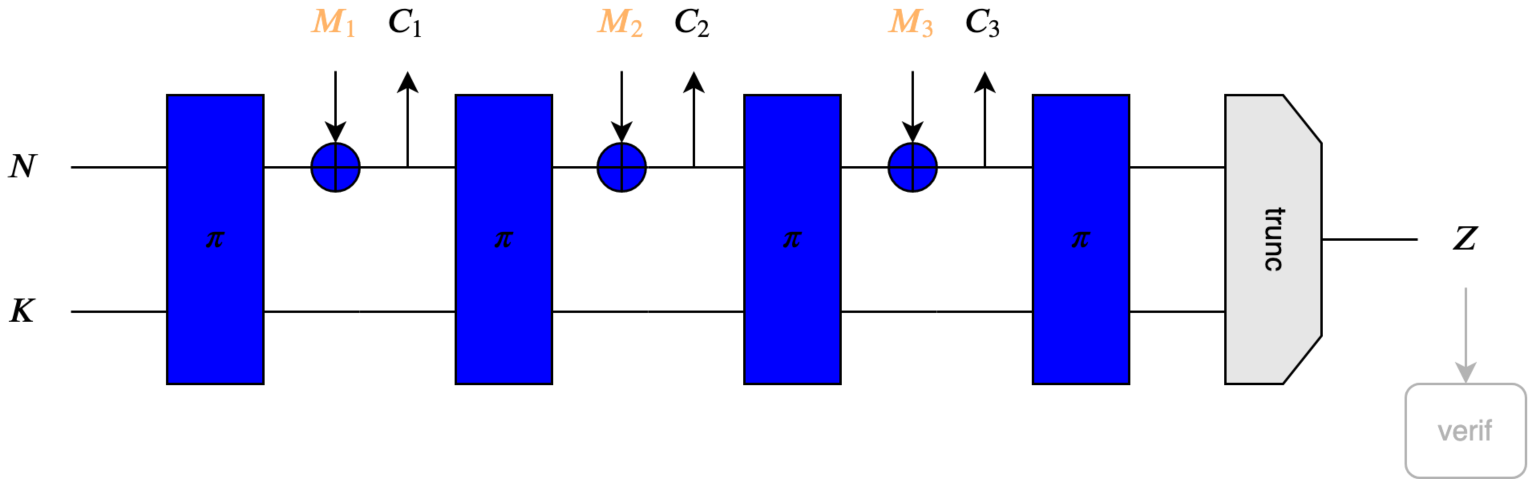
- Target: CCAL1, CIL1 (L in enc only, no misuse)



- Bulk computation only requires SPA security
  - Light green: no averaging is possible (fresh states)
- Calling for so-called "levelled" implementations
  - Energy gains thanks to 2 different implementations

PHOTON-Beetle [B+19]

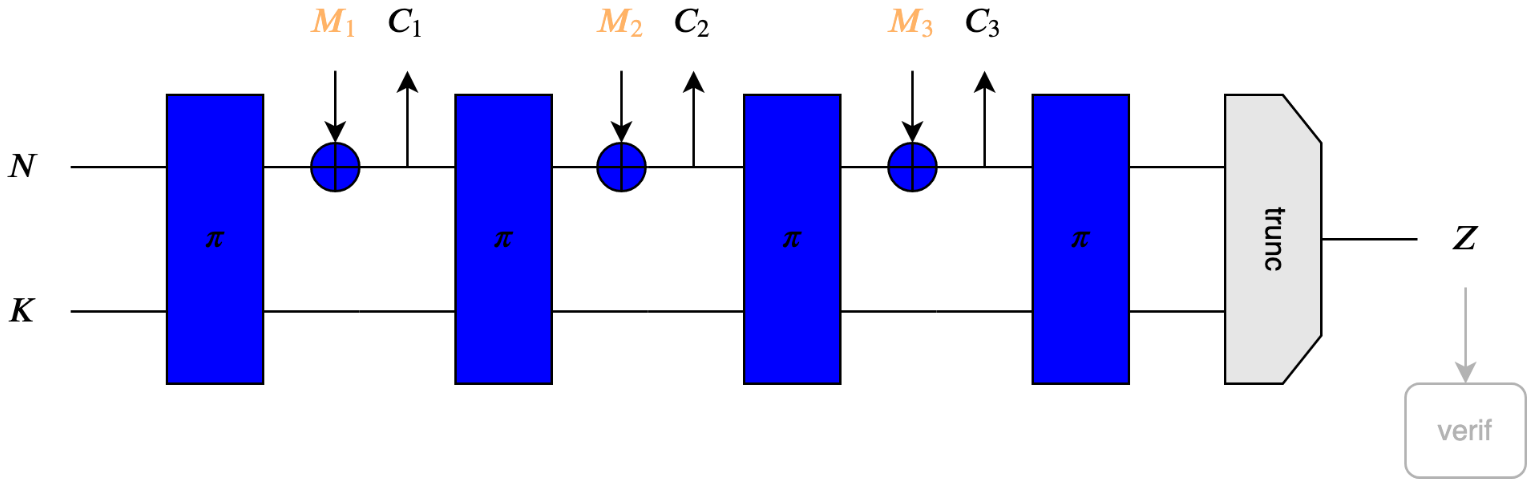
- Target: CCAmL1, CIML1 (L in enc only, misuse)



- DPA security needed everywhere with nonce misuse (idem with decryption leakages)

PHOTON-Beetle [B+19]

- Target: CCAmL1, CIML1 (L in enc only, misuse)



- DPA security needed everywhere with nonce misuse (idem with decryption leakages)
- Others: Gimli, Ketje, Oribatida, …
  - (Roughly applies to all inner-keyed sponges)

Outline

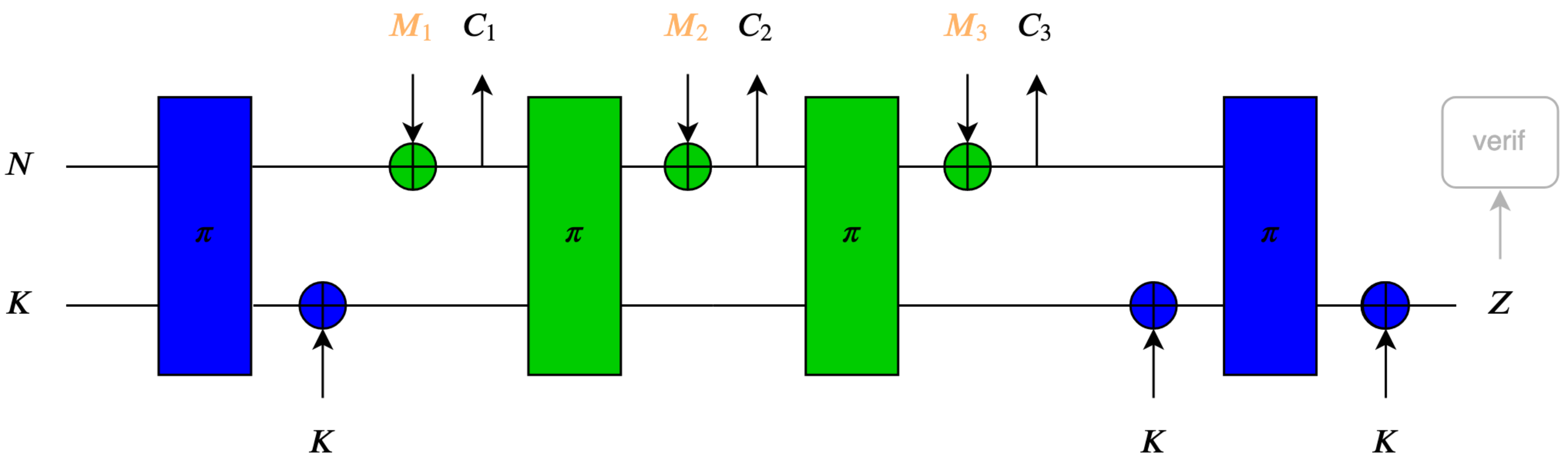
1. Side-channel **(crypt)analysis**: attacks taxonomy
2. Masking **countermeasure**: security vs. cost
3. Security **definitions** (authenticated encryption)
   a. Nonce-respecting setting (i.e., AEL)
   b. Nonce-misuse setting (i.e., AEmL)

1. Leakage-resistant AE **designs** (& implementations)
   - Level 0: no mode-level leakage-resistance
   - Level 1: re-keyed modes (including sponges)
   - Level 2: level 1 + strengthened init./final.
   - Level 3: level 2 + two-passes

2. Conclusions (& the need of open evaluations)

Ascon [DEMS19] (confidentiality)

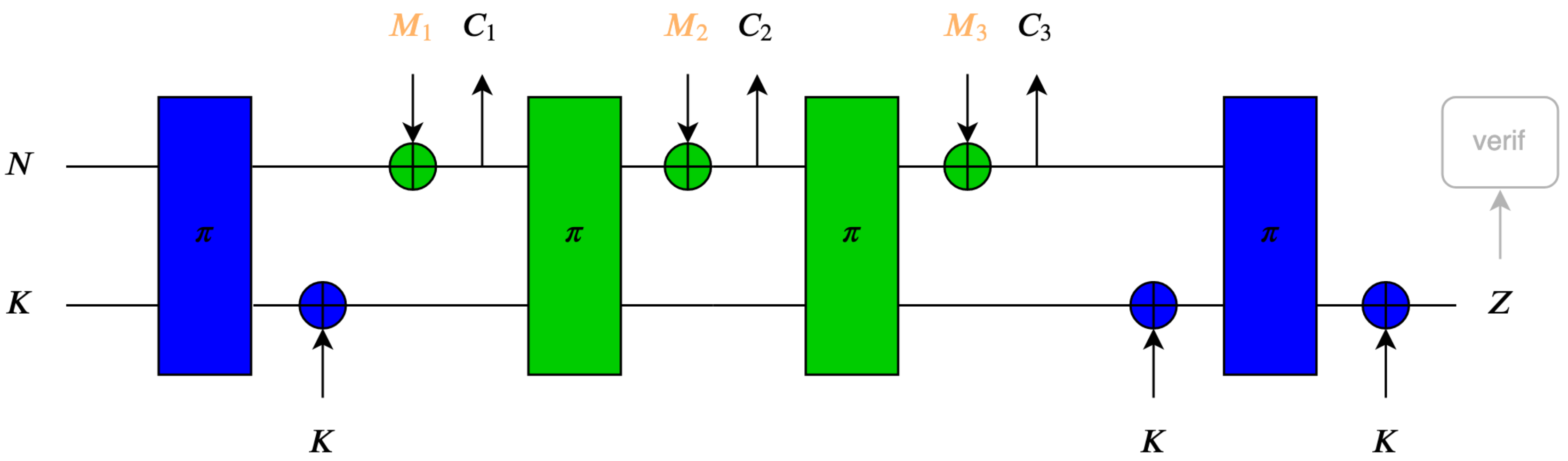
- Target: CCAL1 (L in enc only, no misuse)



- Similar to inner-keyed sponges

Ascon [DEMS19] (confidentiality)

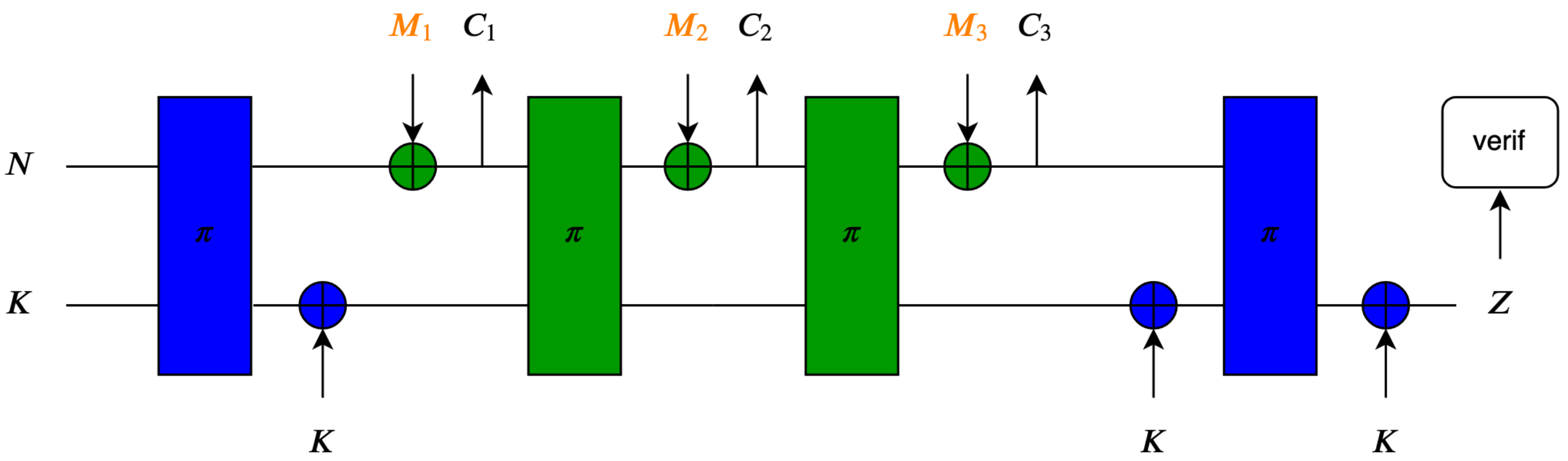
- Target: CCAmL1 (L in enc only, misuse-resilience)



- Strengthened init./final. steps maintain the SPA resistance requirement for the bulk computation with nonce misuse and encryption leakages

Ascon [DEMS19] (confidentiality)

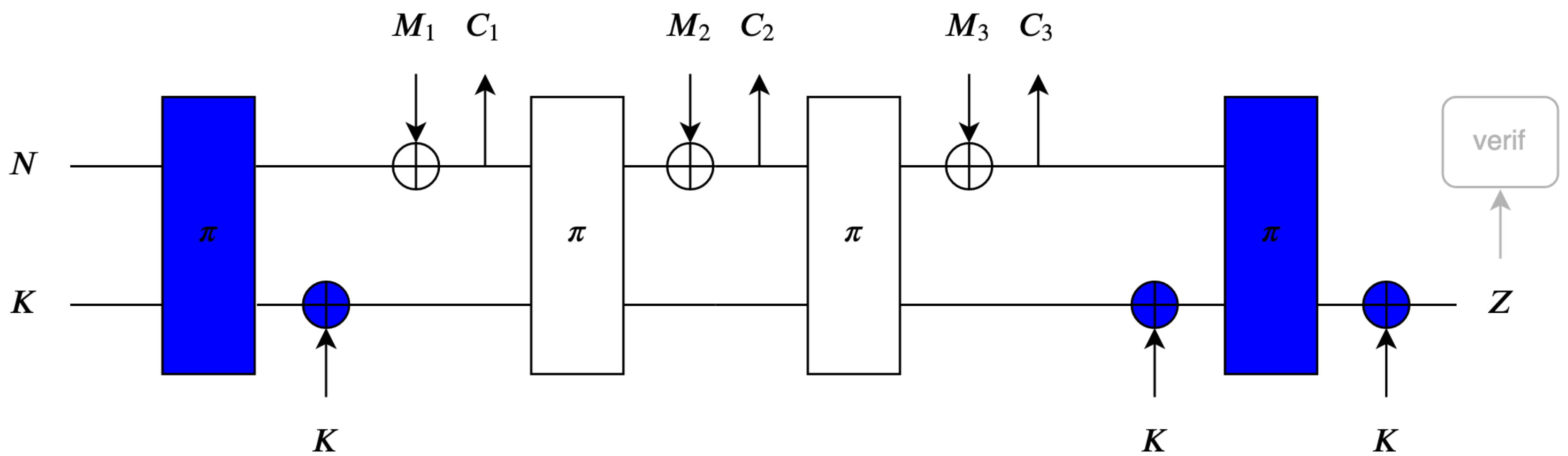
- Target: CCAmL2 (L in enc/dec, misuse-resilience)



- Limited confidentiality with decryption leakages
- Dark orange/green: message decrypted before verification ⇒ the same state can be repeatedly measured, allowing SPA with averaged leakage

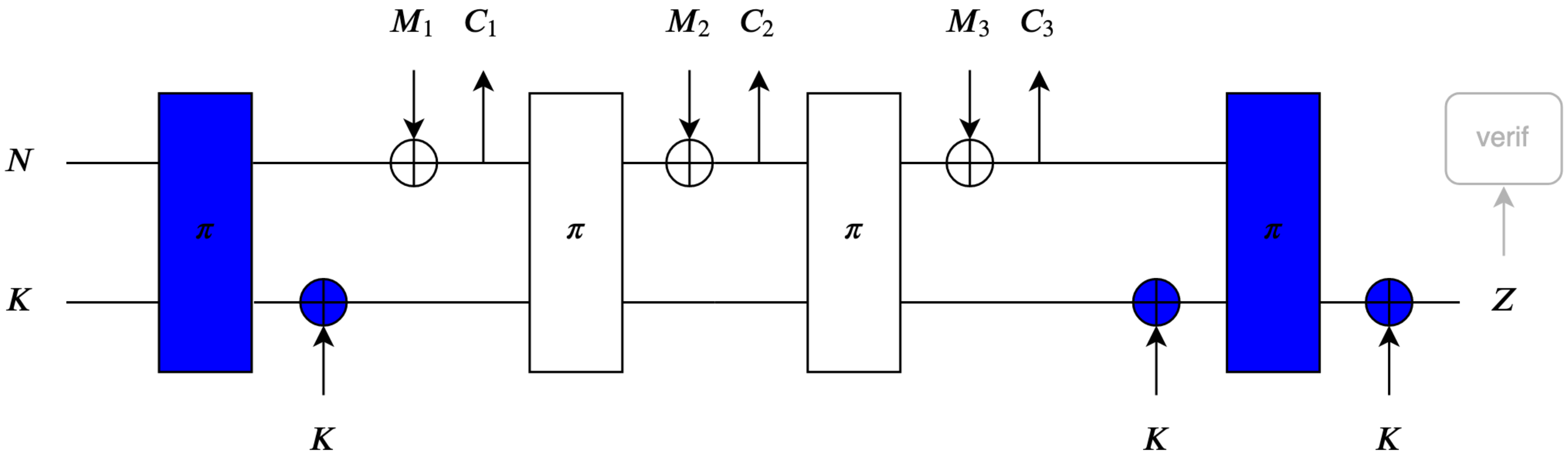
Ascon [DEMS19] (integrity)

- Target: CIL1 (L in enc only, no misuse)



- Bulk computation leakage can be unbounded
- Shows interest of composite definitions!

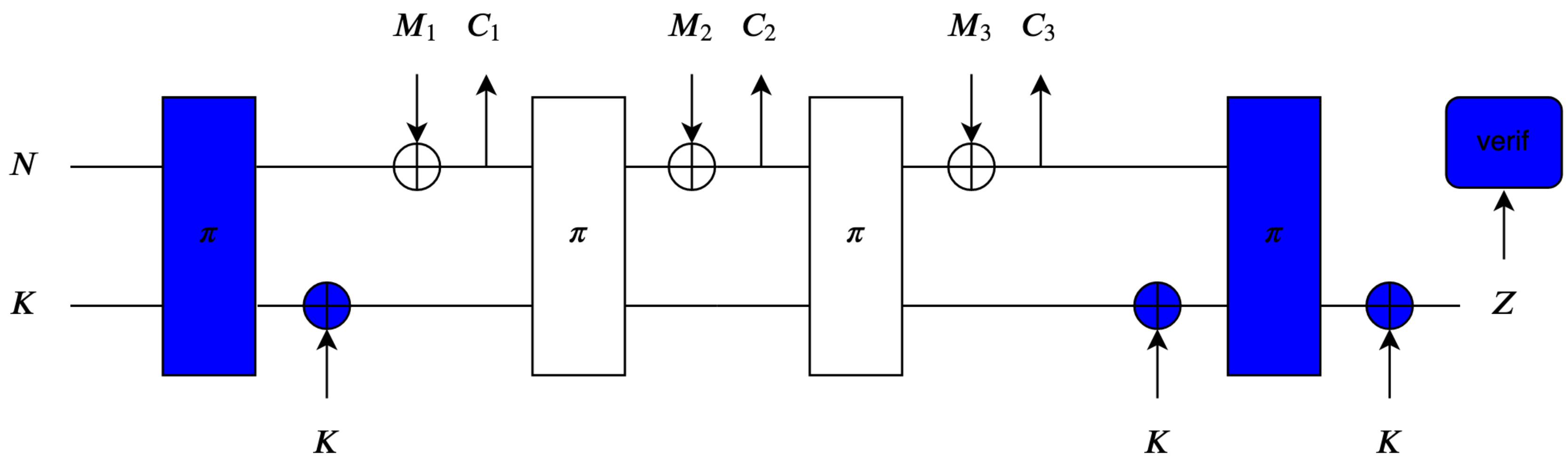
- Target: CIML1 (L in enc only, misuse-resistance)



- Same feature (unbounded leakages for the bulk)

Ascon [DEMS19] (integrity)

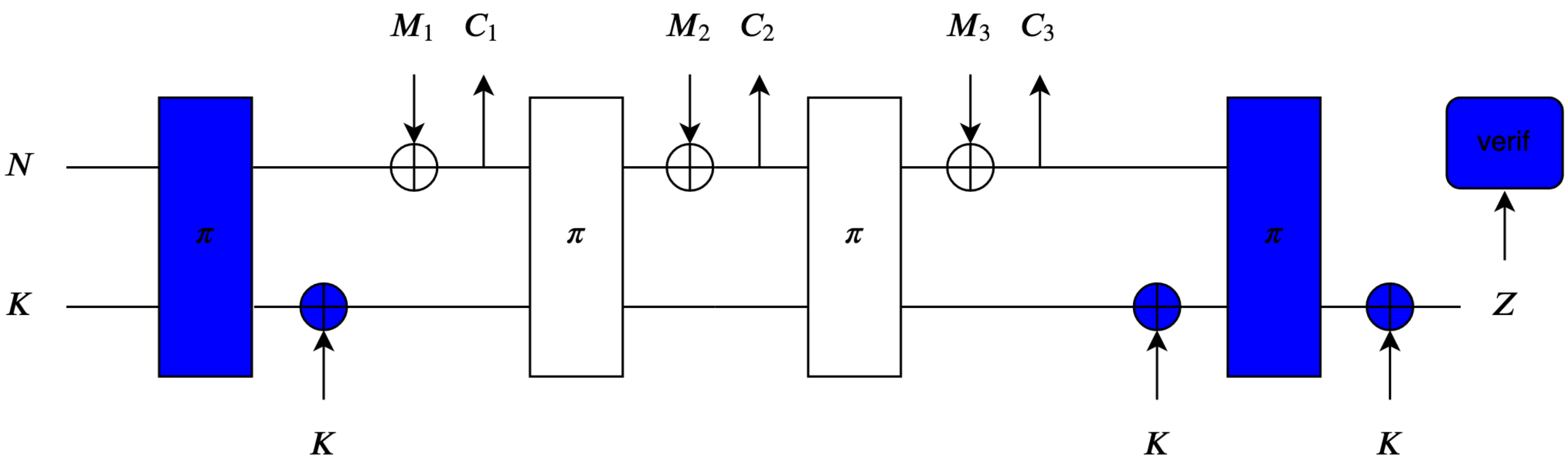
- Target: CIML2 (L in enc/dec, misuse-resistance)



- Tag verification must be protected against DPA
- Shows key recovery security is not enough!

Ascon [DEMS19] (integrity)

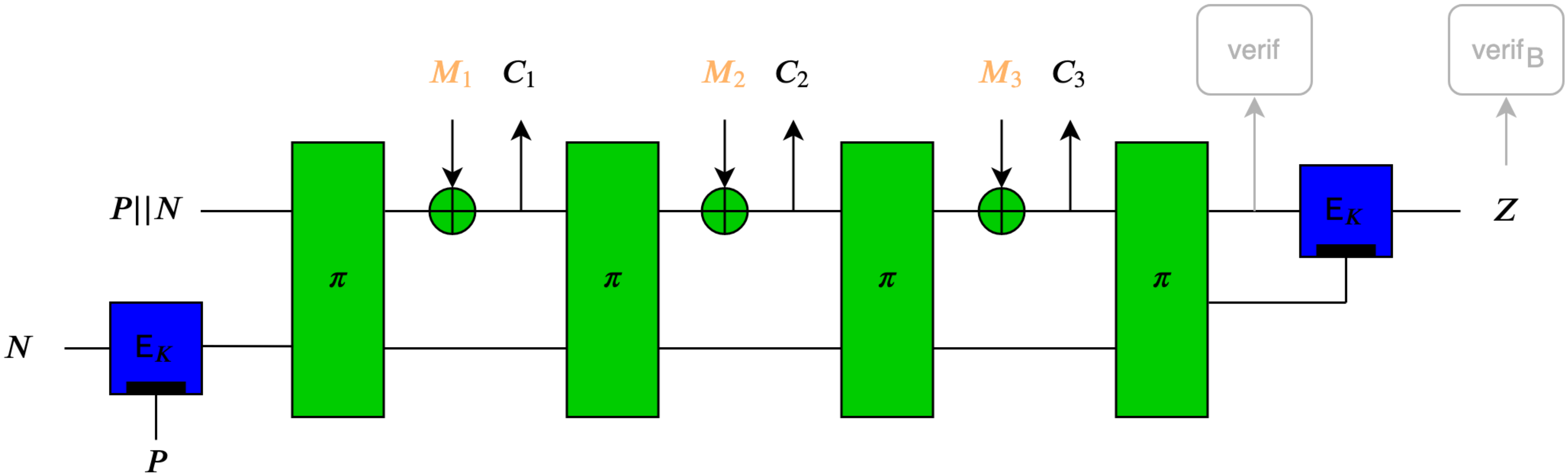
- Target: CIML2 (L in enc/dec, misuse-resistance)



- Tag verification must be protected against DPA
- Shows key recovery security is not enough!

- Others: ACE, GIBBON, Spix, WAGE, …

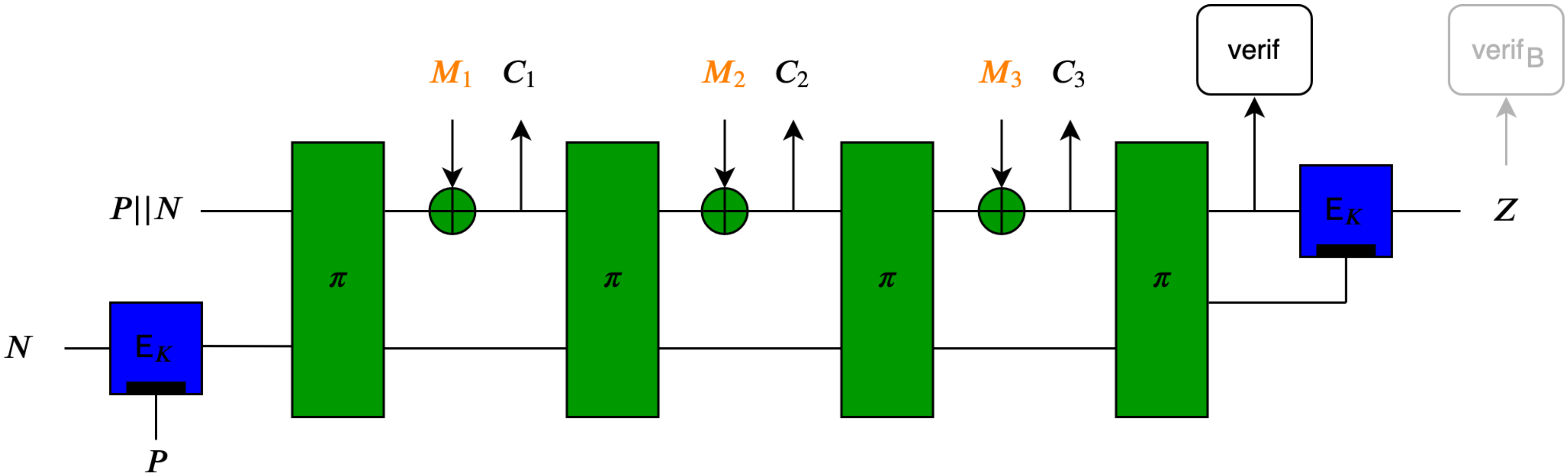
- CCAL1, CCAmL1



≈ further exploiting the leveled implementation concept

- **Similar to ASCON** (but smaller masked state)

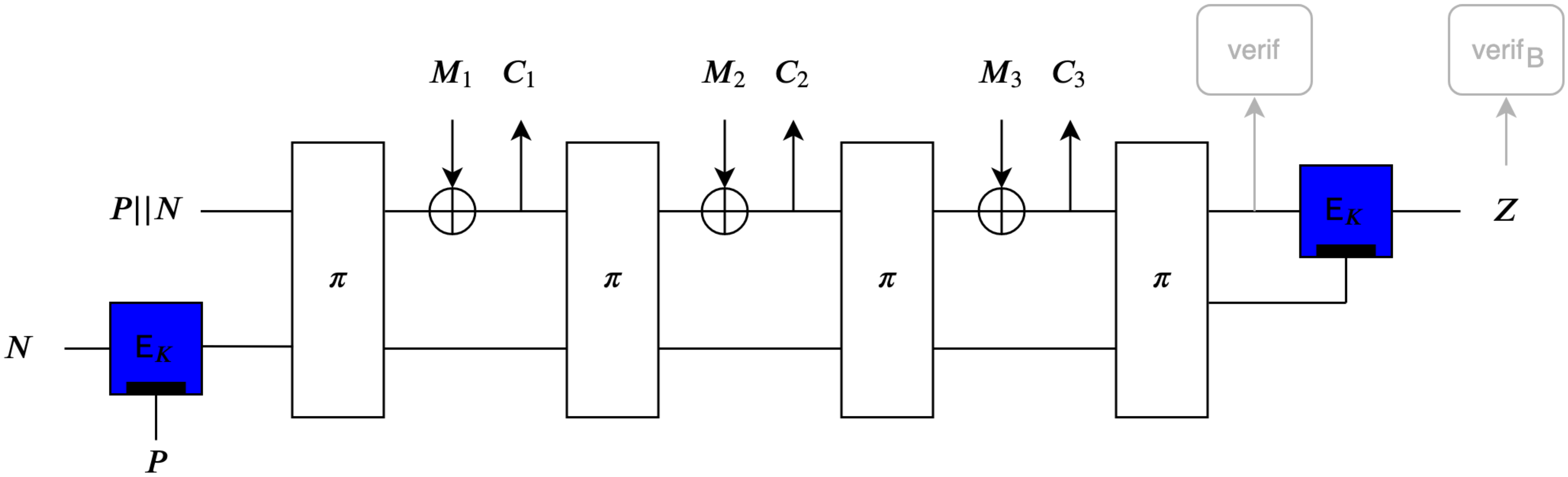
- CCAmL2



≈ further exploiting the leveled implementation concept

- **Similar to ASCON** (but smaller masked state)

- CIL1, CIML1

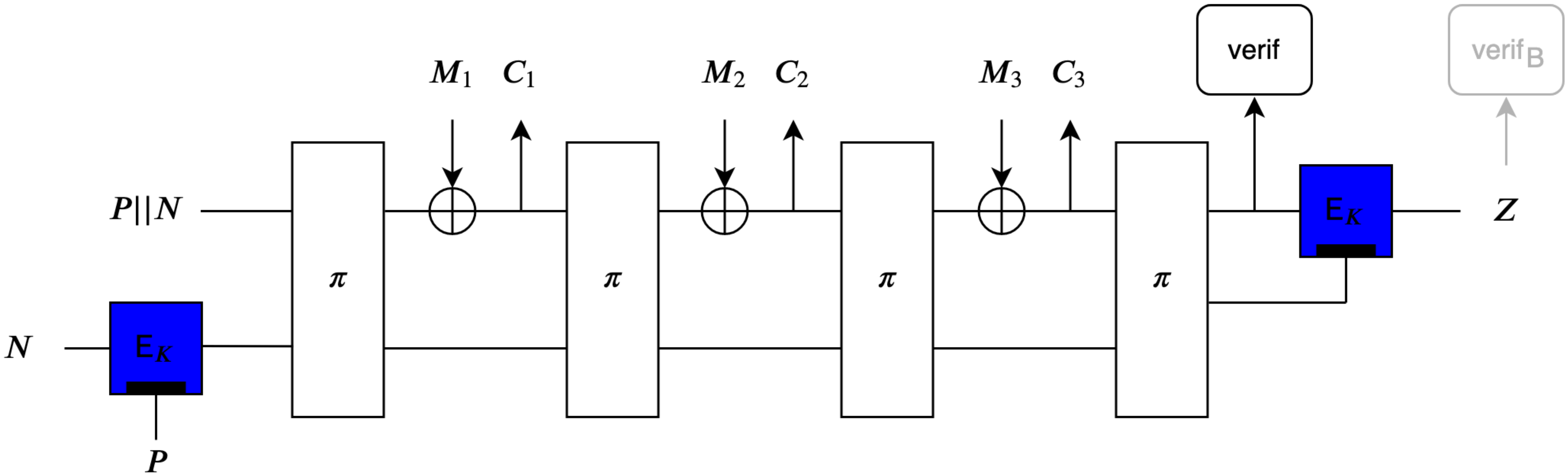


≈ further exploiting the leveled implementation concept

- **Similar to ASCON** (but smaller masked state)

Spook [B+19] (integrity)

- CIML2 (L in enc/dec, misuse-resistance)



- Tag verification tolerates unbounded leakages
- (Inverse-free DPA resistant tag verif. also possible)
- Others: TBC-only variant (TET)

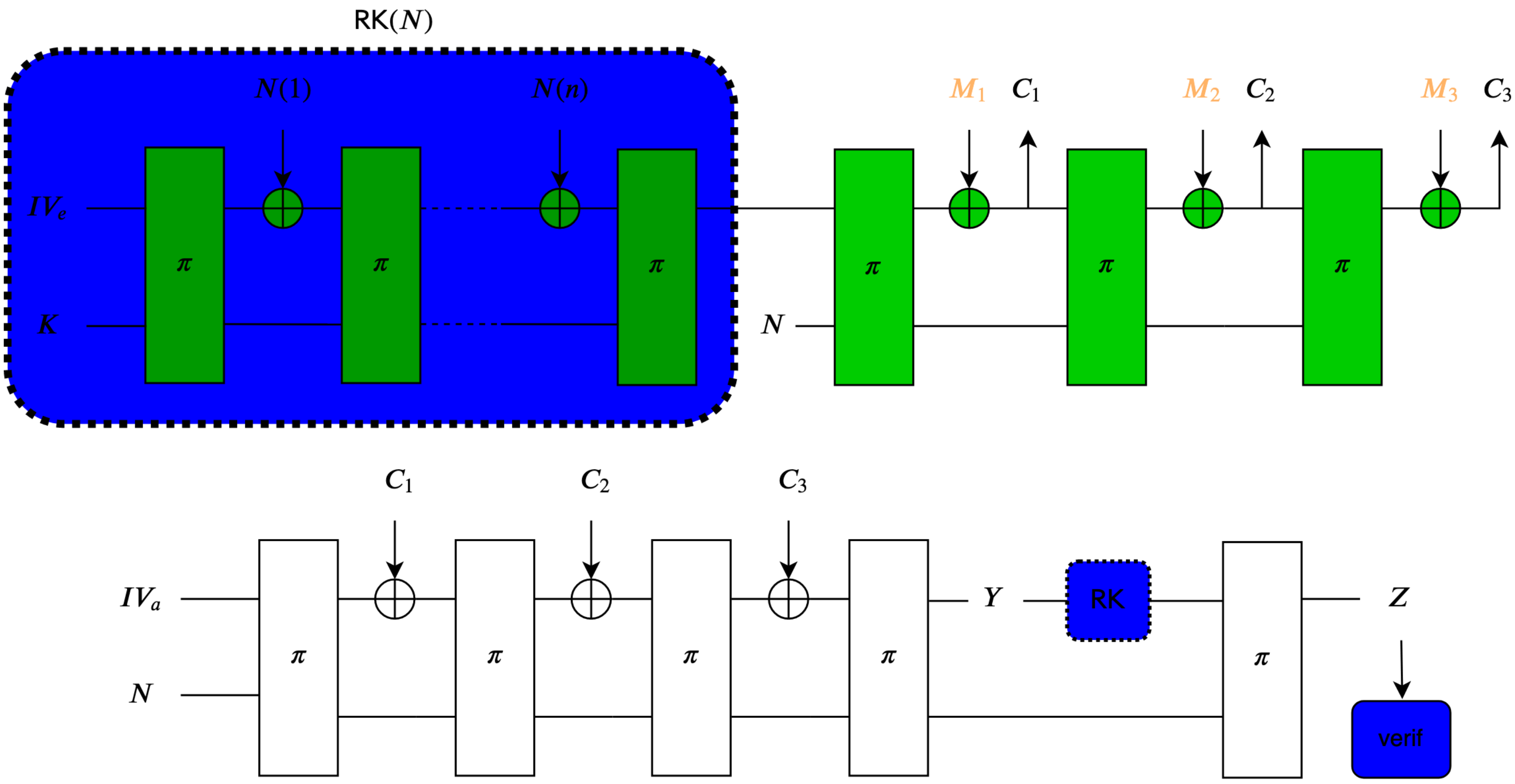
Outline

1. Side-channel **(crypt)analysis**: attacks taxonomy
2. Masking **countermeasure**: security vs. cost
3. Security **definitions** (authenticated encryption)
   a. Nonce-respecting setting (i.e., AEL)
   b. Nonce-misuse setting (i.e., AEmL)

1. Leakage-resistant AE **designs** (& implementations)
   - Level 0: no mode-level leakage-resistance
   - Level 1: re-keyed modes (including sponges)
   - Level 2: level 1 + strengthened init./final.
   - Level 3: level 2 + two-passes

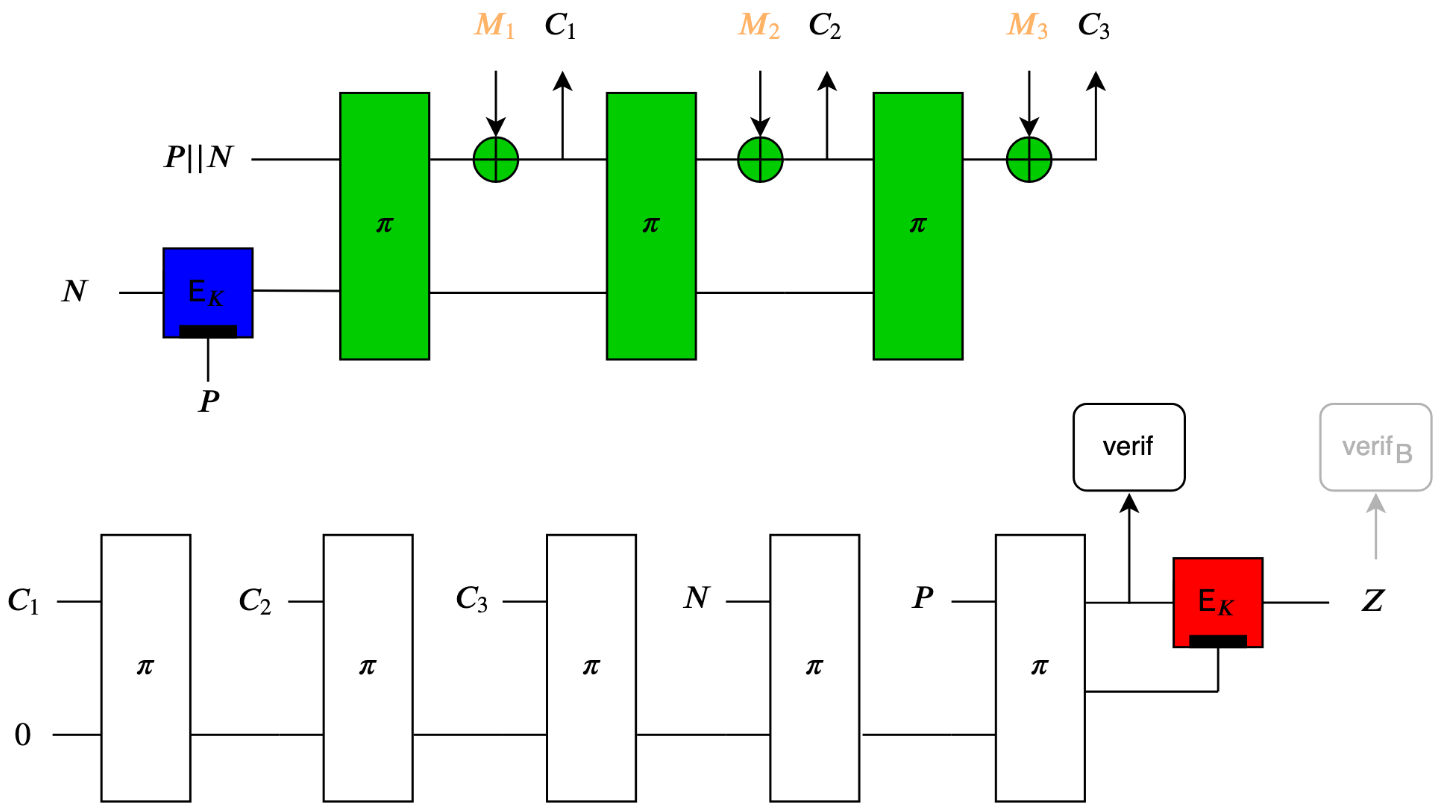
2. Conclusions (& the need of open evaluations)

- CCAmL2 (L in enc/dec, misuse-resilience)



- 2 pass ⇒ confidentiality in dec. if DPA-resistant verif.

- CCAmL2 (L in enc/dec, misuse-resilience)



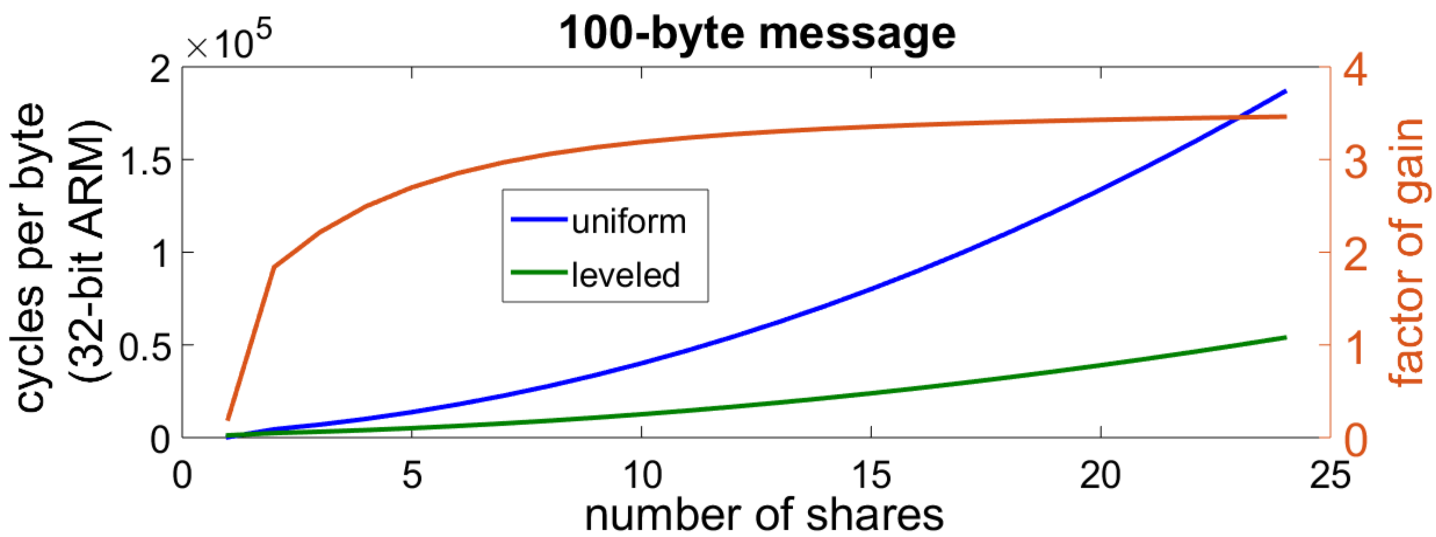
- Tag verification with unbounded leakages

Summarizing

- $\exists$ a tradeoff between mode-level and implementation leakage-resistance
- As the security target and level increase, mode-level leakage-resistance gains more interest

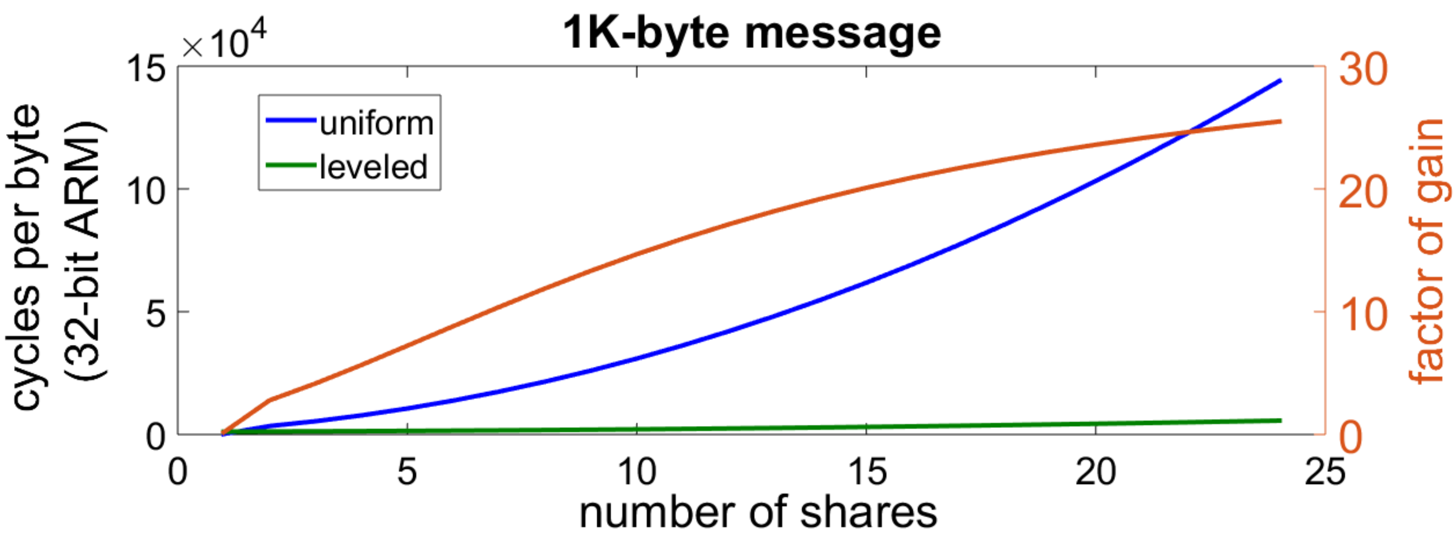
- ∃ a tradeoff between mode-level and implementation leakage-resistance
- As the security target and level increase, mode-level leakage-resistance gains more interest

- Performance gains of levelled implementations



- ∃ a tradeoff between mode-level and implementation leakage-resistance
- As the security target and level increase, mode-level leakage-resistance gains more interest

- Performance gains of levelled implementations



Outline

1. Side-channel **(crypt)analysis**: attacks taxonomy
2. Masking **countermeasure**: security vs. cost
3. Security **definitions** (authenticated encryption)
   a. Nonce-respecting setting (i.e., AEL)
   b. Nonce-misuse setting (i.e., AEmL)

1. Leakage-resistant AE **designs** (& implementations)
   - Level 0: no mode-level leakage-resistance
   - Level 1: re-keyed modes (including sponges)
   - Level 2: level 1 + strengthened init./final.
   - Level 3: level 2 + two-passes

2. Conclusions (& the need of open evaluations)

A theory to guide practice?

- Overall, ∃ a wide zoo of definitions including
  - Leakage-resilience vs. leakage-resistance
  - Misuse-resilience vs. misuse-resistance
  - Leakage in encryption and decryption
  - For integrity and confidentiality

A theory to guide practice?

- Overall, ∃ a wide zoo of definitions including
  - Leakage-resilience vs. leakage-resistance
  - Misuse-resilience vs. misuse-resistance
  - Leakage in encryption and decryption
  - For integrity and confidentiality
- Not black & white notions: all security notions can be reached using more demanding physical assumptions
  - Best solutions to reach each target have to be evaluated
    - *Which requires (tight) bounds and concrete (primitive-dependent) security evaluations*

A theory to guide practice?

- Overall, ∃ a wide zoo of definitions including
  - Leakage-resilience vs. leakage-resistance
  - Misuse-resilience vs. misuse-resistance
  - Leakage in encryption and decryption
  - For integrity and confidentiality
- Not black & white notions: all security notions can be reached using more demanding physical assumptions
  - Best solutions to reach each target have to be evaluated
    - *Which requires (tight) bounds and concrete (primitive-dependent) security evaluations*

⇒ Hope: strong assumptions in the proofs/analyzes indicate where implementers must put most efforts

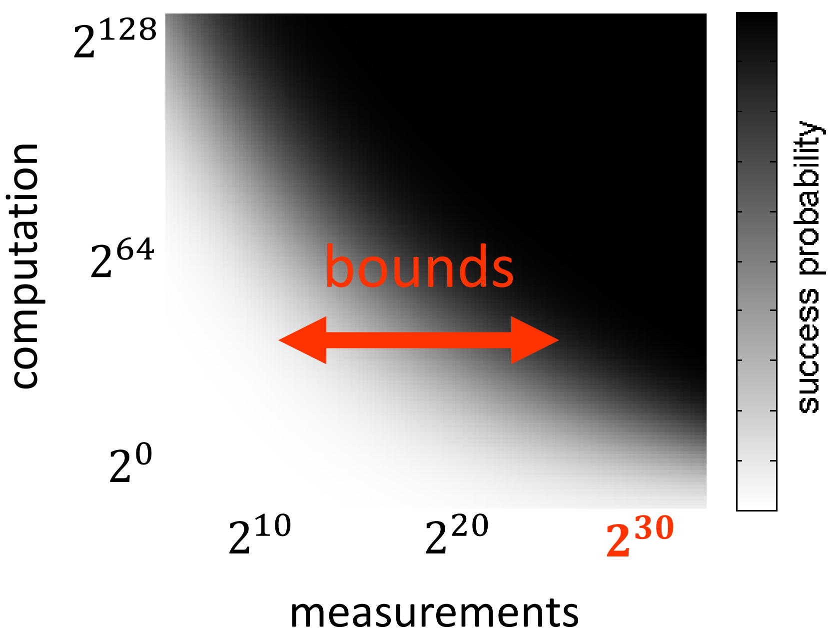
Open problems

- We have good ingredients ⇒ how to mix them?
- Evaluation of AE schemes for various security targets
- Links between the different security notions
- Graceful degradations (for CIML2, CCAmL2)
- Proofs under weaker physical assumptions
- Application to signatures/PKE?
- Cipher designs / key-homomorphic primitives
- Masking (physical defaults, composition, …)
- Improved confidentiality for 1-block messages
- Prototype (open source) implementations
- Anything leading to simple(r) hardware guidelines…

Evaluation challenge 5.3

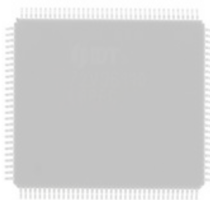
standard practice

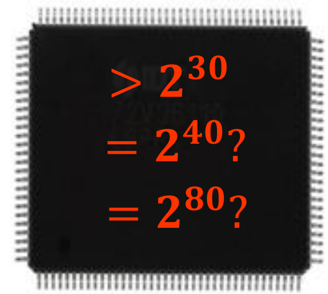
evidence-based evaluations
(*assumptions tested per device!*)



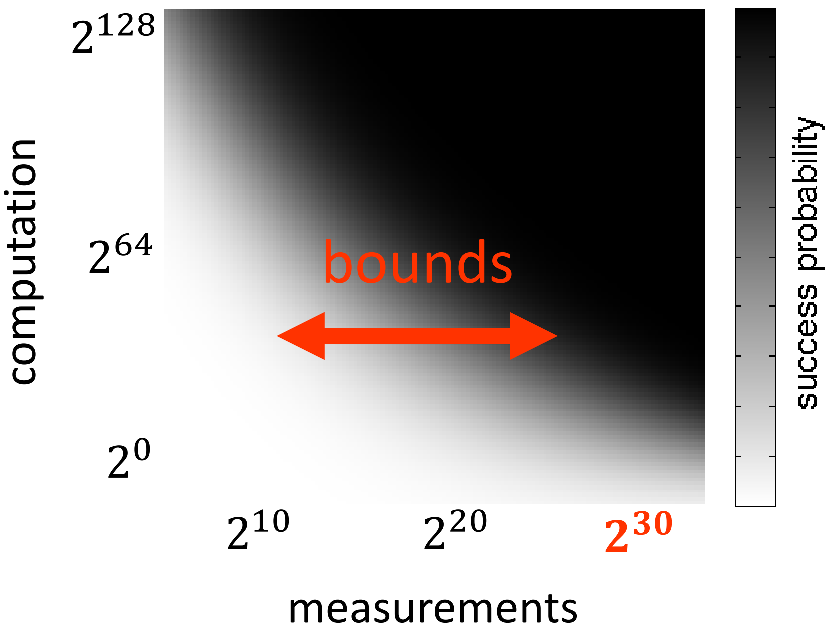
Evaluation challenge

standard practice





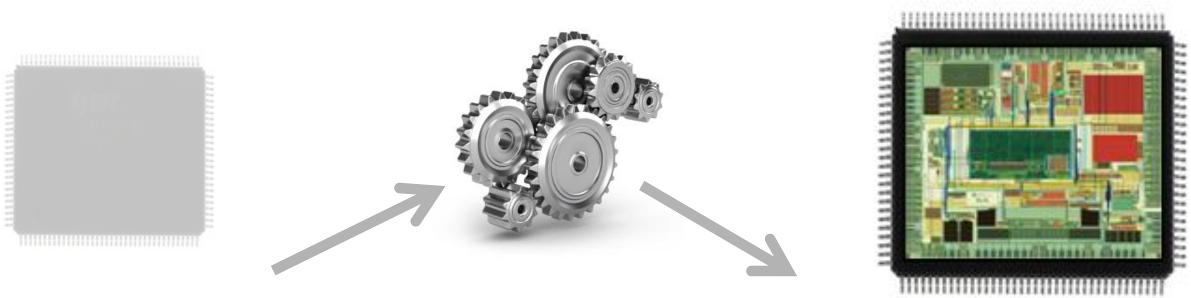
evidence-based evaluations
(*assumptions tested per device!*)



Evaluation challenge

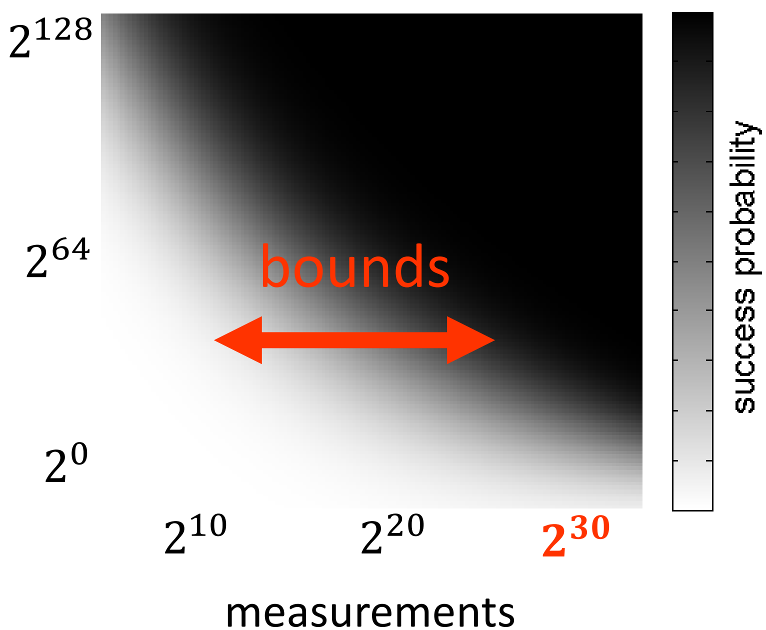
standard practice

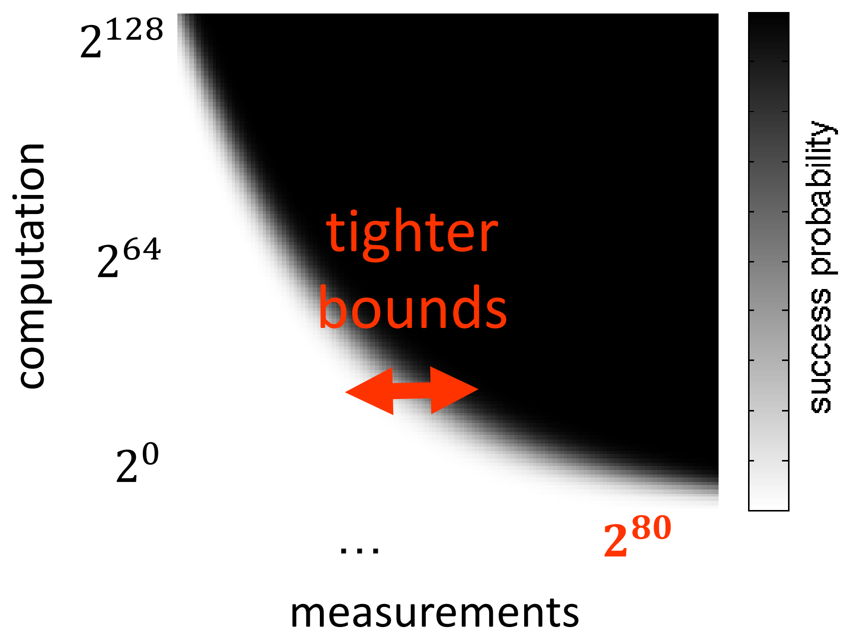
open design & evaluation



evidence-based evaluations
on reduced versions

proof-based evaluations [DFS15,GS18]





THANKS

http://perso.uclouvain.be/fstandae/