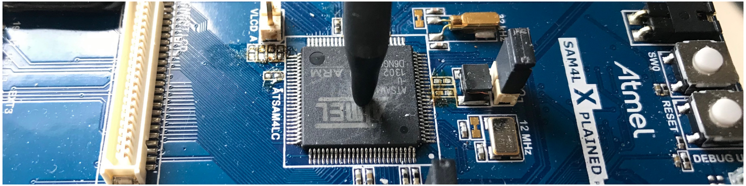


Secure and Efficient Masking of Lightweight Ciphers in Software and Hardware

Olivier Bronchain *Gaëtan Cassiers* François-Xavier Standaert

LWC NIST Workshop 2020

European Research Council
Established by the European Commission

GROUPE CRYPTO

UCLouvain

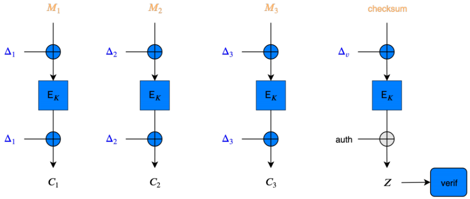
Content

Side-Channel Security at the Mode Level

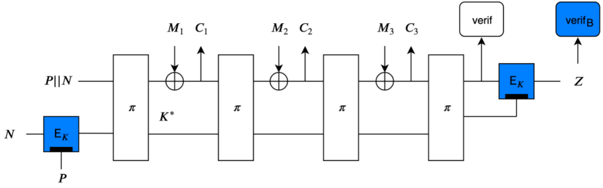
Integrity and confidentiality at the mode level with side-channel:

- Requires different protection levels for parts of an AEAD [Bel+20b].
- Some need DPA (many inputs attack) protection everywhere.
- Some allow a mix of DPA / SPA (few inputs) security and unbounded leakage.

Examples for integrity (qualitatively):



OCB-Pyjamask



Spook

How to Reach DPA Security ?

DPA security is required in many LWC candidates:

- Reach it by reducing DPA security to averaged-SPA security:
  - Isap and DryGascon
- Reach it through the use of masking:
  - Ascon, Spook, OCB-Pyjamask, ...

  Other implementation-level DPA countermeasures: less studied, part of this talk still applies.

In this talk we focus masking since it is well suited for many schemes:

1. How to implement safely and efficiently in software and hardware ?
2. How to compare candidates w.r.t. masking & SCA protections ?

Content

Masking: general principles

Idea: share variables and replace logic gates with “gadgets”.

$$x = x_1 \oplus x_2 \oplus \cdots \oplus x_d$$

$\perp\!\!\!\perp$ t probes

Masking enables “t-probing secure” implementations [ISW03].

Cost of secure gadgets:

- linear: $\mathcal{O}(d)$ (e.g. XOR gate)
- non-linear: $\mathcal{O}(d^2)$ (e.g. AND gate)
- refresh: $\mathcal{O}(d \log d)$ (sometimes required for secure *composition*)

Robust probing model for physical “imperfections” (i.e. glitches, transitions)

A Brief Timeline of Software Masking

Over the last decade:

CHES10 [RP10]
: Implementation of [ISW03] on MCU.

FSE13 [Cor+13]
: Attack on [RP10]: composition issue due to weak refreshing.

Eurocrypt17 [GR17]
: Efficient bitslice masking (proven secure in [CS20]).

Asiacrypt18 [BGR18]
: *Tight private circuits (TPC)*: improved efficiency (probing secure).

Eurocrypt20 [Bel+20a]
: *Tornado*: TPC with register-probing security & automated code generation.

A Brief Timeline of Hardware Masking

Some *glitch-robust probing* secure schemes from the last decade:

TI [NRS11]

Non-completeness + *uniformity* $\Longrightarrow$ first-order glitch-robust probing secure.

CMS [Rep+15] / DOM [GMK16] / UMA [GM18]

Higher-order glitch-robust optimized AND gadgets.

[Moo+19]

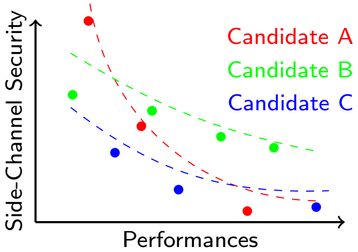
Probing attacks against CMS/DOM/UMA/...

HPC [Cas+20]

Provably secure AND gadgets & *fullVerif* composition verification tool.

Content

How to Compare Candidates ?



It should go in 3 steps:

1. Implement
2. Evaluate performance
3. Evaluate side-channel security

Challenges:

- Evaluate algorithms and not the masking schemes
  - Many optimized implementations for each candidate
- Accurate security evaluation

Given limited expert bandwidth

Side-Channel Security Evaluation

	Automated	Quantitative Worst-case
Probing security verification ▶ Algorithmic security order reductions	✓	✗
Robust probing security verification ▶ Alg. and some physical order reductions	✓	✗
Test Vector Leakage Assesment (TVLA) ▶ Detects order ▶ Based on measurements ▶ Limited to low order, low dimensionality verification ▶ Risk of false negative	✓	✗
Best attack ▶ Can spot multiple kinds of weaknesses ▶ Highly time consuming and skills required (e.g. Spook CTF)	✗	✓

Content

Proxy 1: Count masked AND gates

Starting point:

Masked AND gates make most of the cost of (high-order) implementations.

Software Implementation:

- Clock cycles
- Required randomness
- ...

Hardware Implementation:

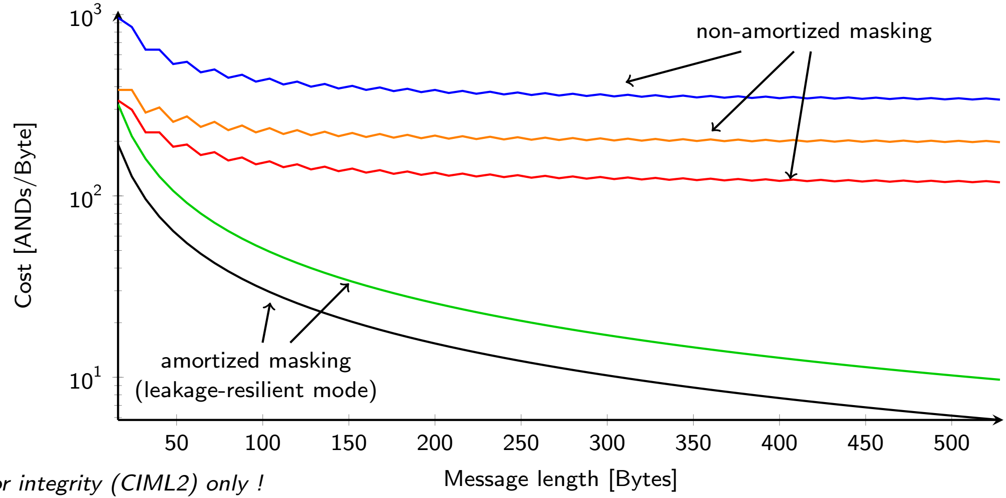
- Latency
- Required randomness
- Area
- ...

Limitations:

- Ignores the rest of the computation (not free!)
- Structure of the cipher also has an impact (e.g. depth)

Integrate counts from [Mey20] with mode-level requirements.

Proxy 1: AND gates per encrypte byte



For integrity (CIML2) only !

Proxy 2: Tornado

Tornado:

- Automated masked C code generation.
- +/-30% overhead w.r.t. hand-optimized.
- Ensure register-probing security.
- TPC+ masking scheme.

Not a magic tool:

- worst-case security (e.g. transitions)?
- optimal performance?
- other masking schemes?

$\Rightarrow$ Tornado implementations hardly comparable to hand-optimized ones.

Suggestion: *Compare Tornado implementations of candidates*

- More realistic than counting masked AND gates
- Easy/Fast implementation: high-level description of primitive
  - 11 candidate's primitives already done by the authors of Tornado

Content

Conclusion

Approaches to compare SCA robustness of candidates:

- Best implementations and best attacks:
  - **Both implementing and evaluating require expertise and time.**
  - **May evaluate the implementer's skills more than the candidates.**
  - Useful byproduct: good implementation of the winner(s) ?
- Proxies:
  - Counting masked AND gates,
  - Tornado: automated software masking,
  - Others ?

Our opinion

- Proxies are more relevant than best implementation & attacks, esp. given resource constraints.
- The proposed proxies already have a good comparative value.

References I

[Bel+20a] Sonia Belaïd et al. "Tornado: Automatic Generation of Probing-Secure Masked Bitsliced Implementations". In: *EUROCRYPT (3)*. Vol. 12107. Lecture Notes in Computer Science. Springer, 2020, pp. 311–341.

[Bel+20b] Davide Bellizia et al. "Mode-Level vs. Implementation-Level Physical Security in Symmetric Cryptography - A Practical Guide Through the Leakage-Resistance Jungle". In: *CRYPTO (1)*. Vol. 12170. Lecture Notes in Computer Science. Springer, 2020, pp. 369–400.

[BGR18] Sonia Belaïd, Dahmun Goudarzi, and Matthieu Rivain. "Tight Private Circuits: Achieving Probing Security with the Least Refreshing". In: *ASIACRYPT (2)*. Vol. 11273. Lecture Notes in Computer Science. Springer, 2018, pp. 343–372.

[Cas+20] Gaëtan Cassiers et al. *Hardware Private Circuits: From Trivial Composition to Full Verification*. Cryptology ePrint Archive, Report 2020/185. `https://eprint.iacr.org/2020/185`. 2020.

[Cor+13] Jean-Sébastien Coron et al. "Higher-Order Side Channel Security and Mask Refreshing". In: *FSE*. Vol. 8424. Lecture Notes in Computer Science. Springer, 2013, pp. 410–424.

References II

[CS20] Gaëtan Cassiers and François-Xavier Standaert. "Trivially and Efficiently Composing Masked Gadgets With Probe Isolating Non-Interference". In: *IEEE Trans. Inf. Forensics Secur.* 15 (2020), pp. 2542–2555.

[GM18] Hannes Groß and Stefan Mangard. "A unified masking approach". In: *J. Cryptographic Engineering* 8.2 (2018), pp. 109–124.

[GMK16] Hannes Groß, Stefan Mangard, and Thomas Korak. "Domain-Oriented Masking: Compact Masked Hardware Implementations with Arbitrary Protection Order". In: *TISCCS*. ACM, 2016, p. 3.

[GR17] Dahmun Goudarzi and Matthieu Rivain. "How Fast Can Higher-Order Masking Be in Software?" In: *EUROCRYPT (1)*. Vol. 10210. Lecture Notes in Computer Science. 2017, pp. 567–597.

[ISW03] Yuval Ishai, Amit Sahai, and David A. Wagner. "Private Circuits: Securing Hardware against Probing Attacks". In: *CRYPTO*. Vol. 2729. Lecture Notes in Computer Science. Springer, 2003, pp. 463–481.

References III

[Mey20] Lauren De Meyer. “Looking at the NIST Lightweight Candidates from a Masking Point-of-View”. In: *IACR Cryptol. ePrint Arch.* 2020 (2020), p. 699.

[Moo+19] Thorben Moos et al. “Glitch-Resistant Masking Revisited or Why Proofs in the Robust Probing Model are Needed”. In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2019.2 (2019), pp. 256–292.

[NRS11] Svetla Nikova, Vincent Rijmen, and Martin Schläffer. “Secure Hardware Implementation of Nonlinear Functions in the Presence of Glitches”. In: *J. Cryptology* 24.2 (2011), pp. 292–321.

[Rep+15] Oscar Reparaz et al. “Consolidating Masking Schemes”. In: *CRYPTO (1)*. Vol. 9215. Lecture Notes in Computer Science. Springer, 2015, pp. 764–783.

[RP10] Matthieu Rivain and Emmanuel Prouff. “Provably Secure Higher-Order Masking of AES”. In: *CHES*. Vol. 6225. Lecture Notes in Computer Science. Springer, 2010, pp. 413–427.