



# CHES 2023 challenge

## SMAesH challenge

Gaëtan Cassiers   Charles Momin   François-Xavier Standaert

# SIMPLE-Crypto



# Content

---

## Challenge description and awards

Winners attack in short

# In the CHES2022 Rump session episode



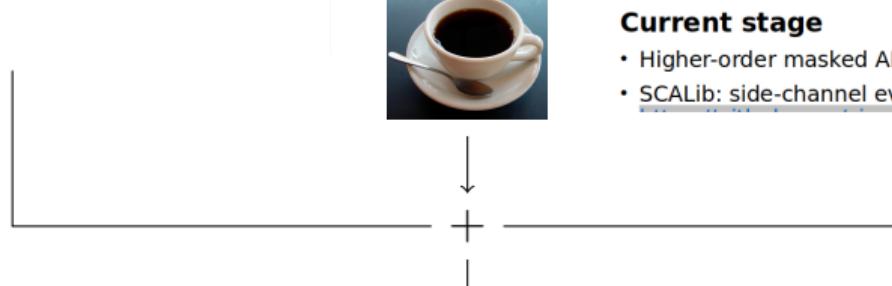
## SIMPLE-Crypto Association

Open Source Secure Implementation of Cryptographic Algorithms

## Concretely...

### Current stage

- Higher-order masked AES in hardware (soon a CTF?)
- SCALib: side-channel evaluation library,



SMAesH challenge

# SMAesH you said?

---

- ▶ Masked AES HW IP
- ▶ HPC2 (arbitrary order)
- ▶ Provably secure
- ▶ PRNG included

# SMAesH you said?

- ▶ Masked AES HW IP
- ▶ HPC2 (arbitrary order)
- ▶ Provably secure
- ▶ PRNG included

The screenshot shows the GitHub repository page for 'SMAesH' (Public). The repository has 2 branches and 0 tags. The main branch is selected. The commit history shows initial commits for various files by 'cmomin' on Jun 15, 2023. The files include: beh\_simu, docs, formal\_verif, hdl, ignore, COPYRIGHT.txt, LICENSE.txt, and README.md. The 'About' section describes it as 'Masked Hardware AES-128 Encryption with HPC2'. The 'Releases' section indicates 'No releases published'. The 'Packages' section shows 'No packages published'. The 'Contributors' section lists 'cmomin' (Merin Charles).

**SMAesH** (Public)

main · 2 branches · 0 tags

Go to file Add file `Code` About

**cmomin add fix revision history** committed on Jun 15 · 3 commits

File	Commit	Date
beh_simu	Initial commit	5 months ago
docs	add fix revision history	3 months ago
formal_verif	Initial commit	5 months ago
hdl	Initial commit	5 months ago
ignore	Initial commit	5 months ago
COPYRIGHT.txt	Initial commit	5 months ago
LICENSE.txt	Initial commit	5 months ago
README.md	Initial commit	5 months ago

**README.md**

**SIMPLE-Crypto's Masked AES in Hardware (SMAesH)**

An optimized masked hardware implementation of AES-128 Encryption using HPC2.

This repository contains the masked AES hardware implementation published by [SIMPLE-Crypto](#).

See PDF [technical documentation](#) and [preliminary evaluation report](#).

About · Releases · Packages · Contributors

cmomin · Merin Charles

# SMAesH you said?

- ▶ Masked AES HW IP
- ▶ HPC2 (arbitrary order)
- ▶ Provably secure
- ▶ PRNG included

The image shows two side-by-side screenshots. On the left is a GitHub repository page for 'SMAesH' with a public fork. It displays a file tree with 'beh\_simu', 'docs', 'formal\_verif', 'hdl', 'ignore', 'COPYRIGHT.txt', 'LICENSE.txt', and 'README.md'. On the right is a 'Contents' page for 'SMAesH: technical documentation' titled 'Masked Hardware AES-128 Encryption with HPC2'. The table of contents includes sections like Overview, History, Features, Core User Guide, Core Architecture, Core Performances, Core Verification, and Copyright, each with detailed sub-sections and page numbers.

# SMAesH you said?

---

- ▶ Masked AES HW IP
- ▶ HPC2 (arbitrary order)
- ▶ Provably secure
- ▶ PRNG included

**SMAesH (Public)**

main · 2 branches · 0 tags

**csmomin add fix revision history**

- beh\_simu
- docs
- formal\_verif
- hdl
- ignore
- COPYRIGHT.txt
- LICENSE.txt
- README.md

**README.md**

**SIMPLE-Crypto'**

An optimized masked hardware implementation of the AES-128 encryption algorithm.

This repository contains the mask generation logic and the datapath module.

See PDF [technical documentation](#)

## SMAesH: technical documentation

Masked Hardware AES-128 Encryption with HPC2

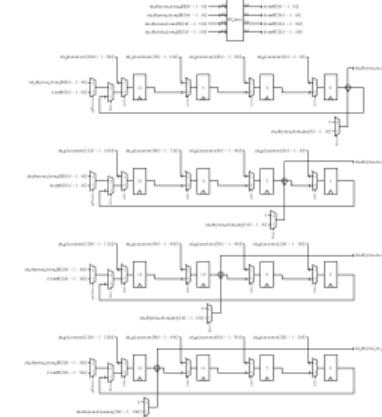
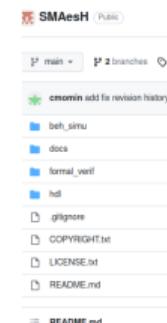


Figure 9: Global architecture of the `HS256bit_state.datapathmodule`. The value held by the DFF at index  $i$  is depicted by the signal `sh_reg[4:0]` in the HDL.

# SMAesH you said?

- ▶ Masked AES HW IP
- ▶ HPC2 (arbitrary order)
- ▶ Provably secure
- ▶ PRNG included



## SMAesH: technical documentation

Masked Hardware AES-128 Encryption with HPC2

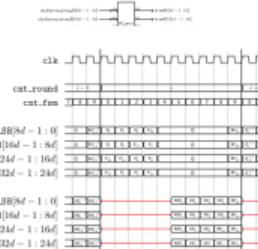


Figure 13: Data going into / coming from the S-boxes during a round.

begin. The round function and the key scheduling algorithm are executed in parallel by interleaving the S-boxes usage appropriately. In particular, the first cycle of the execution is used to start the key scheduling algorithm by asserting `feedLabKey` and `abx.valid.in`. During this cycle, both the module `MKaes_32bits.state.datapath` and `MKaes_32bits.key.datapath` are enabled.

Then, the core enters into a normal regime that computes a round in 10 cycles, as depicted in Figure 13. A typical round starts with 4 clock cycles during which data is read from the state registers, XORed with the subkey and fed to the S-boxes, which performs the `AddRoundkey`, `ShiftRows` and `SubBytes` layers for the full state (one column per cycle). During these cycles, `abx.valid.in` is asserted and data (`state` and `subkey`) loops over the shift registers. At the fifth cycle of a round (i.e., when `cnt.fsm = 4`), the module `MKaes_32bits.key.datapath` is disabled in order to wait one cycle for the S-

Figure 9: Global architecture of the `MKAes_32bits.state.datapath` module. The value held by the DFF at index  $i$  is depicted by the signal `sh.reg.out[1]` in the HDL.

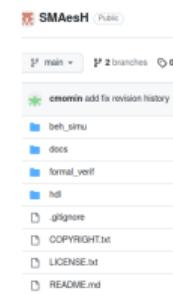


Figure 9: Global architecture of the `MKAes_32bits.state.datapath` module.

# SMAesH you said?

---

- ▶ Masked AES HW IP
- ▶ HPC2 (arbitrary order)
- ▶ Provably secure
- ▶ PRNG included



## SIMPLE-Crypto'

An optimized masked hardware IP

This repository contains the mask

See PDF [technical documentation](#)



## SMAesH: preliminary evaluation report

SIMPLE-Crypto

### Contents

<a href="#">1 Overview</a>	1
<a href="#">2 History</a>	1
<a href="#">3 Evaluation scope</a>	1
<a href="#">4 Measurement Setup and Traces Pre-processing</a>	2
<a href="#">5 Evaluation Methodology</a>	2
<a href="#">6 Results</a>	3
<a href="#">7 Conclusion</a>	5
<a href="#">8 Copyright</a>	5

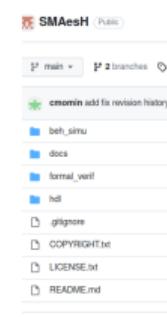
### 1 Overview

This document presents the findings of the preliminary evaluation of the resistance of the SMAesH (aes\_enc128\_32bita\_hpc2) hardware IP to power analysis attacks. The evaluation has been performed by the developers of SMAesH (SIMPLE-Crypto).

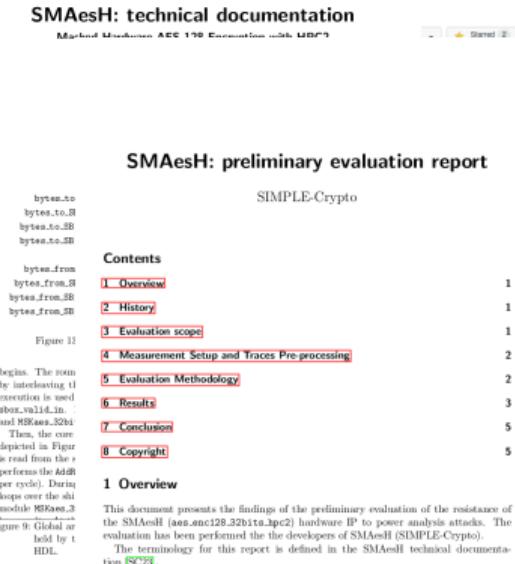
The terminology for this report is defined in the SMAesH technical documentation [\[SC2\]](#).

SMAesH you said?

- ▶ Masked AES HW IP
  - ▶ HPC2 (arbitrary order)
  - ▶ Provably secure
  - ▶ PRNG included



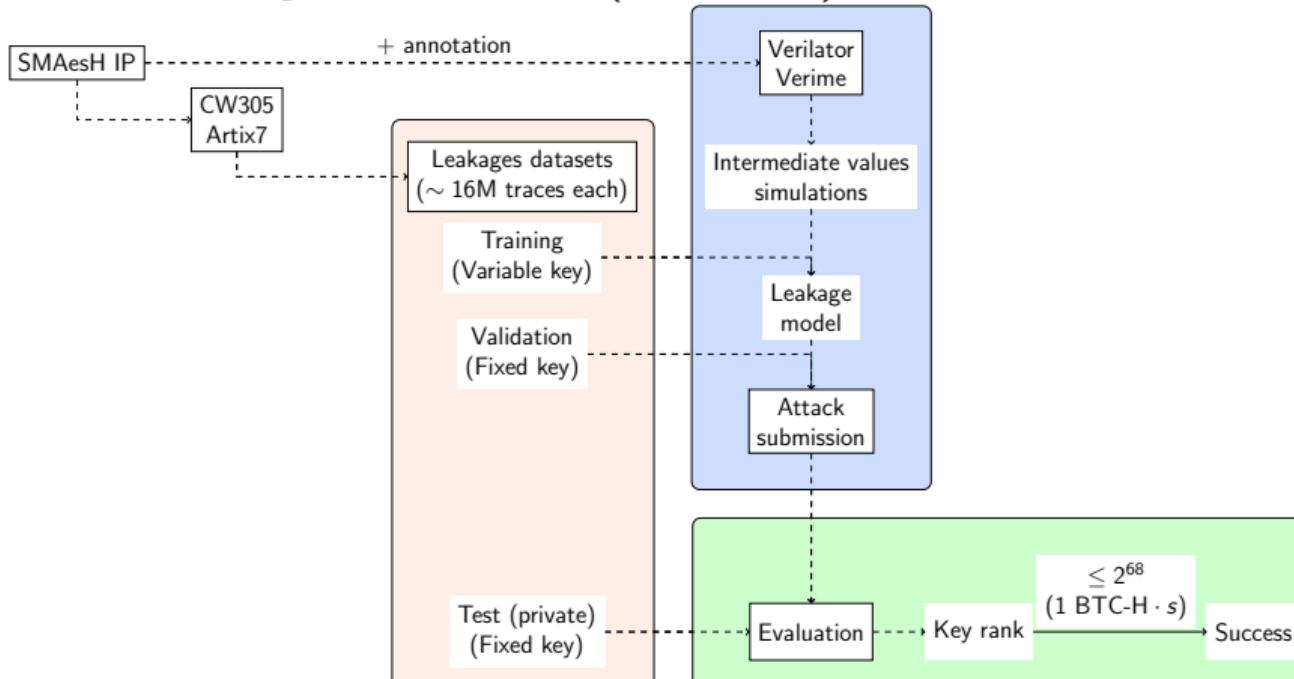
# SIMPLE-Cryp



→ See [simple-crypto.org/activities/smaesh/](http://simple-crypto.org/activities/smaesh/)

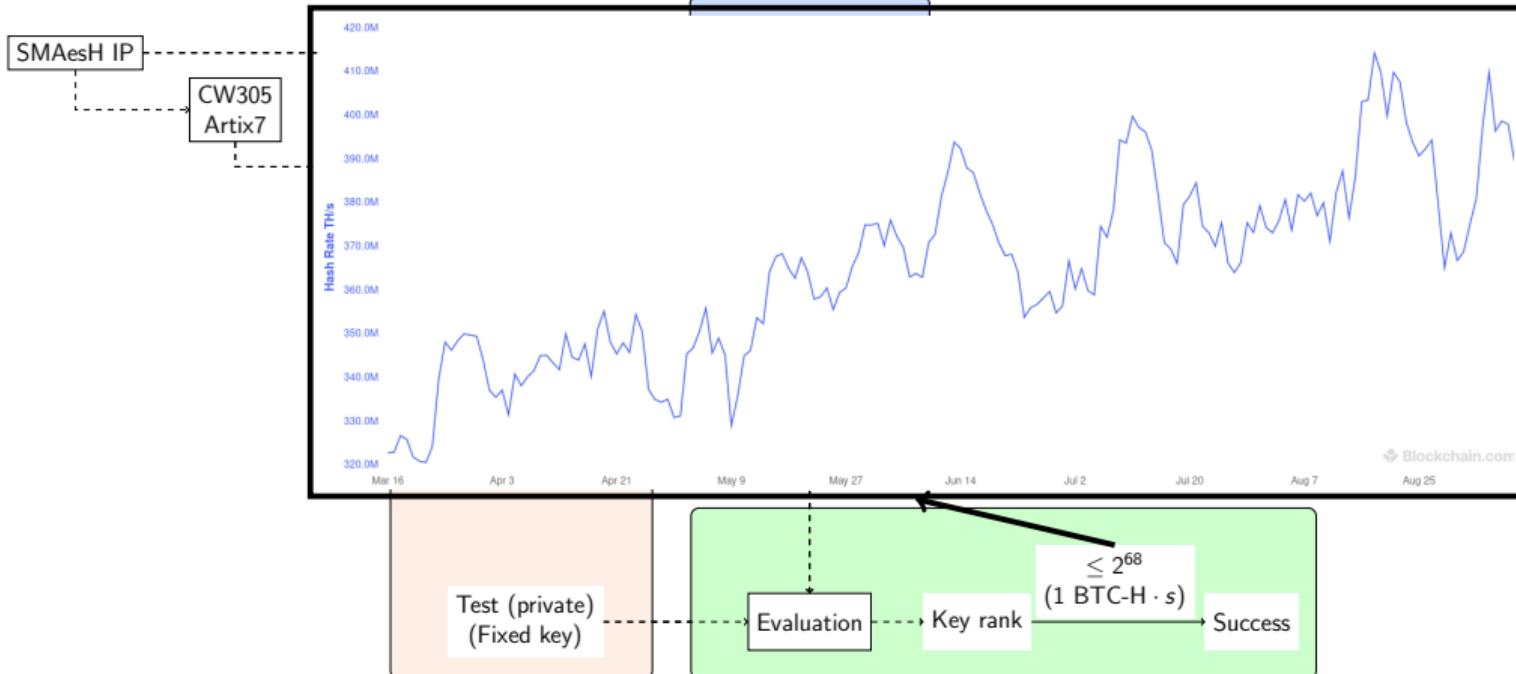
# Challenge description

Goal: SCA attack against SMAesH (first order)



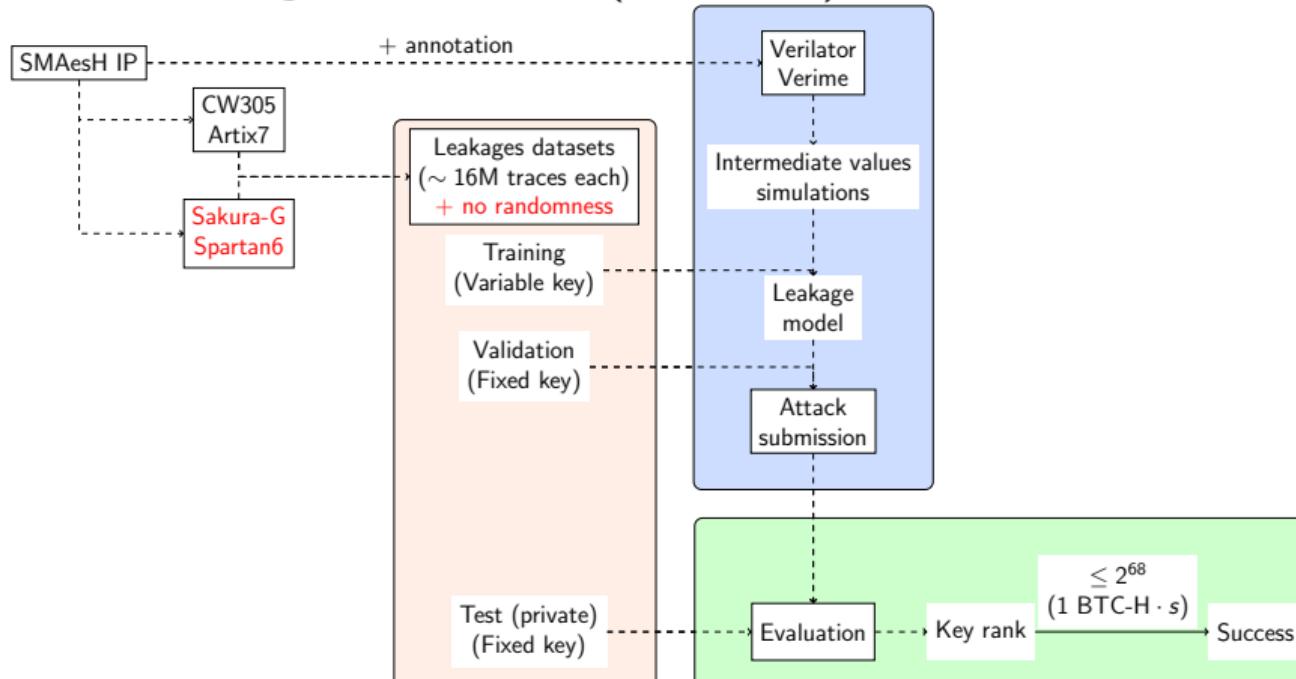
# Challenge description

Goal: SCA attack against SMAesH (first order)



# Challenge description

Goal: SCA attack against SMAesH (first order)



# Challenge Stats

---

- ▶ 5 teams:  $\infty \times$  '20 CTF :D

# Challenge Stats

---

- ▶ 5 teams:  $\infty \times '20$  CTF :D



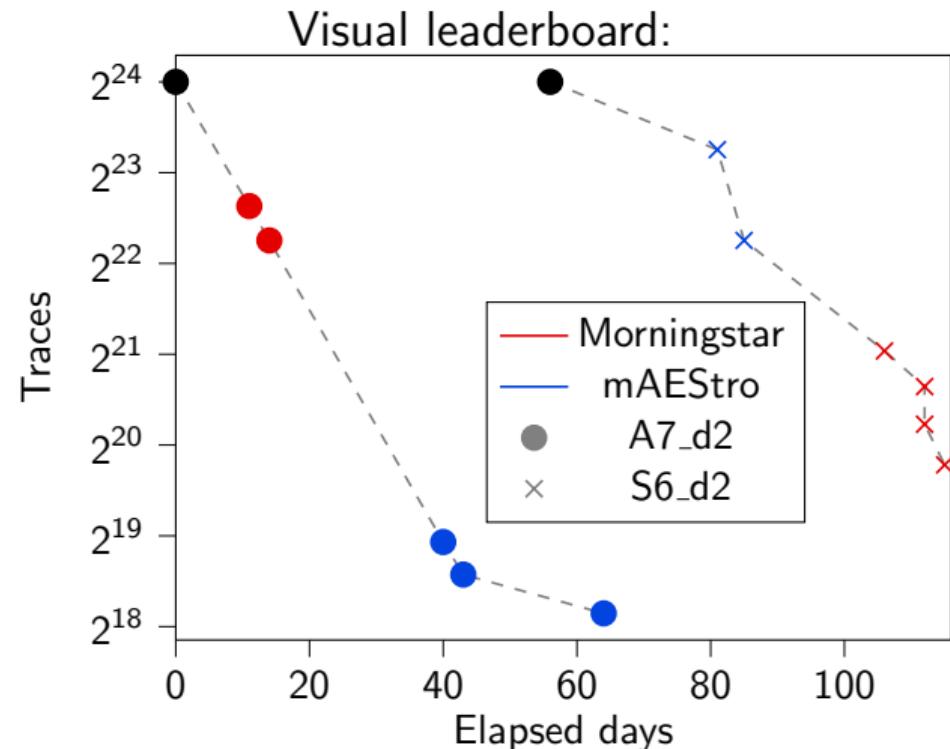
# Challenge Stats

---

- ▶ 5 teams:  $\infty \times$  '20 CTF :D
- ▶ 112 submissions

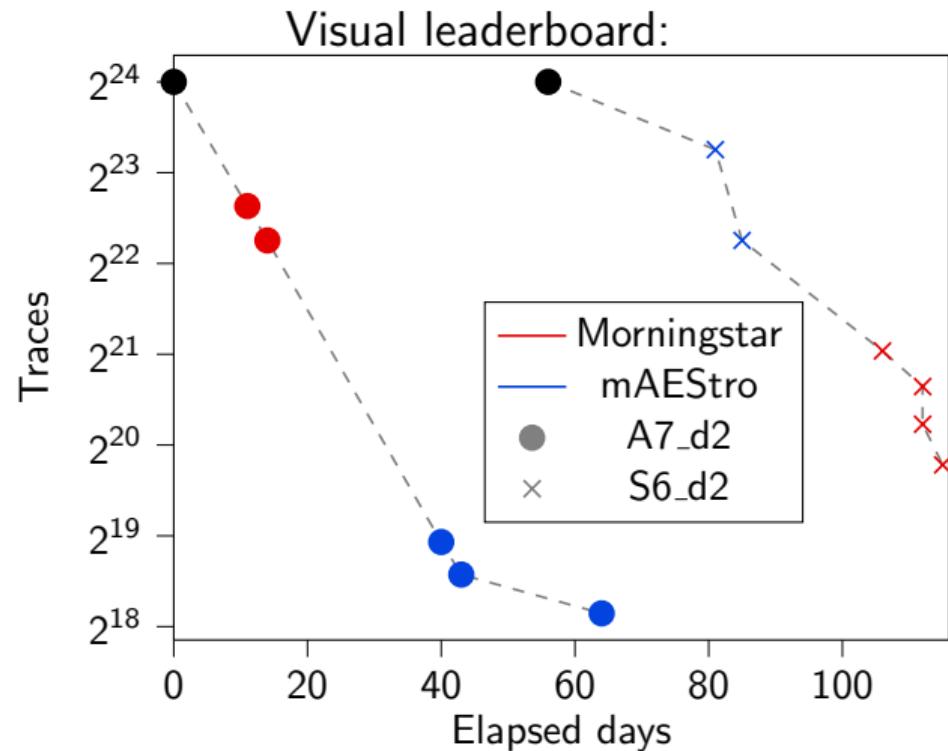
# Challenge Stats

- ▶ 5 teams:  $\infty \times '20$  CTF :D
- ▶ 112 submissions
- ▶ A7\_d2:
  - 77 submissions
  - 5 successful attacks
- ▶ S6\_d2:
  - 35 submissions
  - 6 successful attacks



# Challenge Stats

- ▶ 5 teams:  $\infty \times$  '20 CTF :D
- ▶ 112 submissions
- ▶ A7\_d2:
  - 77 submissions
  - 5 successful attacks
- ▶ S6\_d2:
  - 35 submissions
  - 6 successful attacks
- ▶ Peak rates:
  - ▶ 2 submissions/h/team
  - ▶ 12 submissions/day



# These damn ninjas cutting onions...

**NinjaLab** **NinjaLab**  
270 followers  
2d

The **NinjaLab** team will be present in Prague for CHES 2023 with surprises 🎁

Spoiler: a ninja did what ninjas do best: sneak into the top of CHES 2023 challenge (codename "team Sec-artorez") with a "single trace" attack 😊

It lasted for few hours before organizers updated the rules and rejected the submission 😞

**My SMAesH Attacks**

**Valid attacks**

Submission name	Target	Traces	Successful	log2 rank	Challenge Status
One_Shot	A7_d2	1	<input checked="" type="checkbox"/>	61.9	Current challenger!
Hawai	A7_d2	200000	<input type="checkbox"/>	128.0	
Everest	A7_d2	210000	<input type="checkbox"/>	126.7	

cement 🔍  
ently detected a team mounting a side-channel attack system rather than against the hardware targets' t forbidden by the rules, this strategy is not in line w etition, which leads us to consider it differently than  
e took the following actions (justified by the final ren les):  
the following rule:

# Winners

---

- ▶ Prizes<sup>a</sup> for most points and best attack!

# Winners

---

- ▶ Prizes<sup>a</sup> for most points and best attack!
  - ▶ A7\_d2:
    - ▶ Most points: Valence Cristiani (team mAEStro)
    - ▶ Best attack: Valence Cristiani (team mAEStro)
  - ▶ S6\_d2:
    - ▶ Most points: Valence Cristiani (team mAEStro)
    - ▶ Best attack: Thomas Marquet (team Morningstar)
- <sup>a</sup>Teams cannot win more than one prize...
- ▶ Valence Cristiani (NinjaLab) is awarded 1000 USD
  - ▶ Thomas Marquet (AAU) is awarded 500 USD

Congratulations!

# Content

---

Challenge description and awards

Winners attack in short



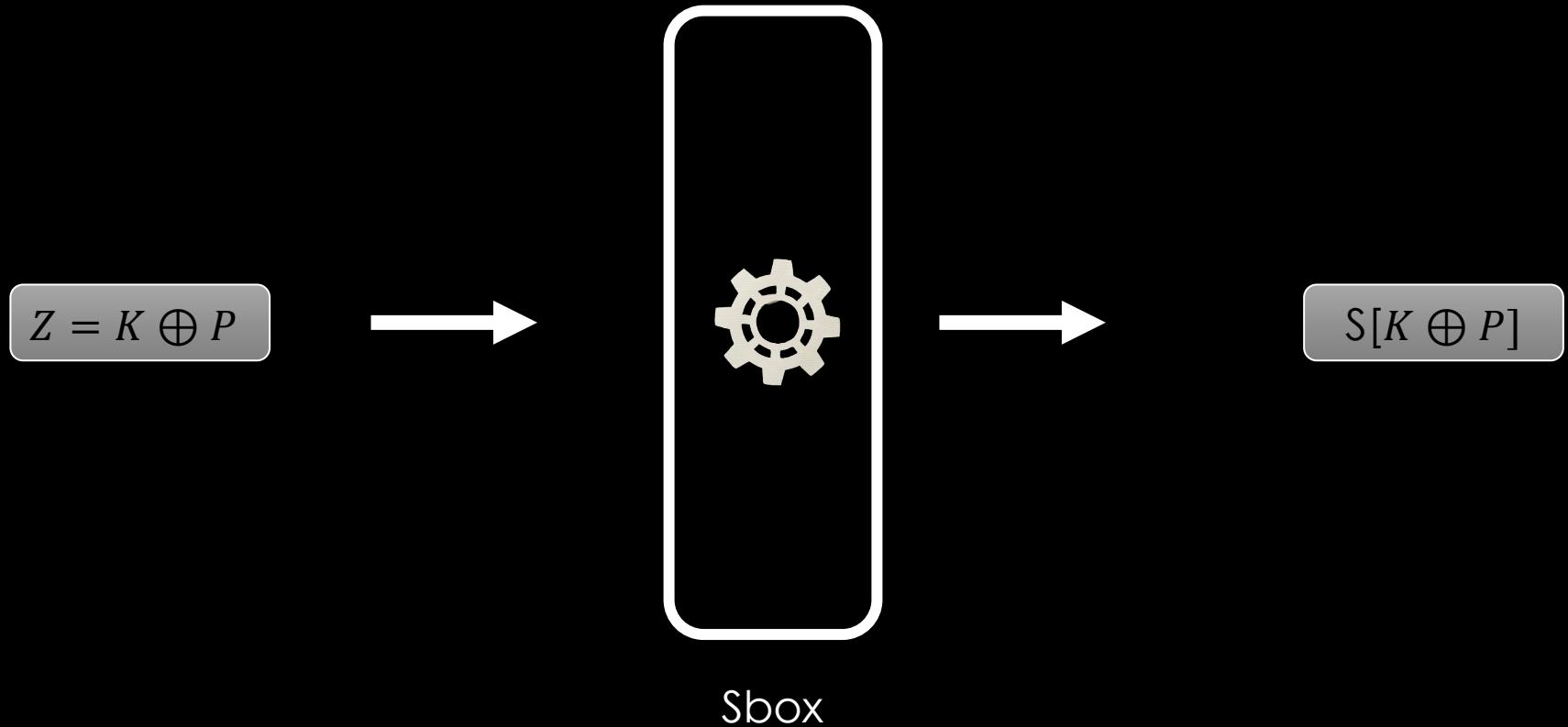
# HOW TO SMASH THE SMAESH CHES CHALLENGE ?

BEING HONEST OR EVIL...



Valence Cristiani | Ches 2023  
NinjaLab

# BEING AN HONEST GUY



# BEING AN HONEST GUY

T1 = U0 + U3	T8 = U7 + T6	T15 = T5 + T11	T22 = T7 + T21
T2 = U0 + U5	T9 = U7 + T7	T16 = T5 + T12	T23 = T2 + T22
T3 = U0 + U6	T10 = T6 + T7	T17 = T9 + T16	T24 = T2 + T10
T4 = U3 + U5	T11 = U1 + U5	T18 = U3 + U7	T25 = T20 + T17
T5 = U4 + U6	T12 = U2 + U5	T19 = T7 + T18	T26 = T3 + T16
T6 = T1 + T5	T13 = T3 + T4	T20 = T1 + T19	T27 = T1 + T12
T7 = U1 + U2	T14 = T6 + T11	T21 = U6 + U7	

Figure 5: Top linear transform in forward direction.

T23 = U0 + U3	T19 = T22 + R5	T17 = U2 # T19	T6 = T22 + R17
T22 = U1 # U3	T9 = U7 # T1	T20 = T24 + R13	T16 = R13 + R19
T2 = U0 # U1	T10 = T2 + T24	T4 = U4 + T8	T27 = T1 + R18
T1 = U3 + U4	T13 = T2 + R5	R17 = U2 # U5	T15 = T10 + T27
T24 = U4 # U7	T3 = T1 + R5	R18 = U5 # U6	T14 = T10 + R18
R5 = U6 + U7	T25 = U2 # T1	R19 = U2 # U4	T26 = T3 + T16
T8 = U1 # T23	R13 = U1 + U6	Y5 = U0 + R17	

Figure 6: Top linear transform in reverse direction.

$Z = K \oplus P$  

  $S[K \oplus P]$

M1 = T13 x T6	M17 = M5 + T24	M33 = M27 + M25	M49 = M43 x T16
M2 = T23 x T8	M18 = M8 + M7	M34 = M21 x M22	M50 = M38 x T9
M3 = T14 + M1	M19 = M10 + M15	M35 = M24 x M34	M51 = M37 x T17
M4 = T19 x D	M20 = M16 + M13	M36 = M24 + M25	M52 = M42 x T15
M5 = M4 + M1	M21 = M17 + M15	M37 = M21 + M29	M53 = M45 x T27
M6 = T3 x T16	M22 = M18 + M13	M38 = M32 + M33	M54 = M41 x T10
M7 = T22 x T9	M23 = M19 + T25	M39 = M23 + M30	M55 = M44 x T13
M8 = T26 + M6	M24 = M22 + M23	M40 = M35 + M36	M56 = M40 x T23
M9 = T20 x T17	M25 = M22 x M20	M41 = M38 + M40	M57 = M39 x T19
M10 = M9 + M6	M26 = M21 + M25	M42 = M37 + M39	M58 = M43 x T3
M11 = T1 x T15	M27 = M20 + M21	M43 = M37 + M38	M59 = M38 x T22
M12 = T4 x T27	M28 = M23 + M25	M44 = M39 + M40	M60 = M37 x T20
M13 = M12 + M11	M29 = M28 x M27	M45 = M42 + M41	M61 = M42 x T1
M14 = T2 x T10	M30 = M26 x M24	M46 = M44 x T6	M62 = M45 x T4
M15 = M14 + M11	M31 = M20 x M23	M47 = M40 x T8	M63 = M41 x T2
M16 = M3 + M2	M32 = M27 x M31	M48 = M39 x D	

Sbox tower fileds implementation

# BEING AN HONEST GUY

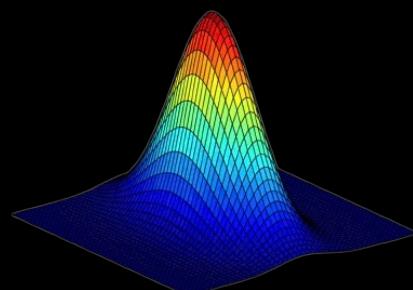
T1 = U0 + U3	T8 = U7 + T6	T15 = T5 + T11	T22 = T7 + T21
T2 = U0 + U5	T9 = U7 + T7	T16 = T5 + T12	T23 = T2 + T22
T3 = U0 + U6	T10 = T6 + T7	T17 = T9 + T16	T24 = T2 + T10
T4 = U3 + U5	T11 = U1 + U5	T18 = U3 + U7	T25 = T20 + T17
T5 = U4 + U6	T12 = U2 + U5	T19 = T7 + T18	T26 = T3 + T16
T6 = T1 + T5	T13 = T3 + T4	T20 = T1 + T19	T27 = T1 + T12
T7 = U1 + U2	T14 = T6 + T11	T21 = U6 + U7	

Figure 5: Top linear transform in forward direction.

T23 = U0 + U3	T19 = T22 + R5	T17 = U2 # T19	T6 = T22 + R17
T22 = U1 # U3	T9 = U7 # T1	T20 = T24 + R13	T16 = R13 + R19
T2 = U0 # U1	T10 = T2 + T24	T4 = U4 + T8	T27 = T1 + R18
T1 = U3 + U4	T13 = T2 + R5	R17 = U2 # U5	T15 = T10 + T27
T24 = U4 # U7	T3 = T1 + R5	R18 = U5 # U6	T14 = T10 + R18
R5 = U6 + U7	T25 = U2 # T1	R19 = U2 # U4	T26 = T3 + T16
T8 = U1 # T23	R13 = U1 + U6	Y5 = U0 + R17	

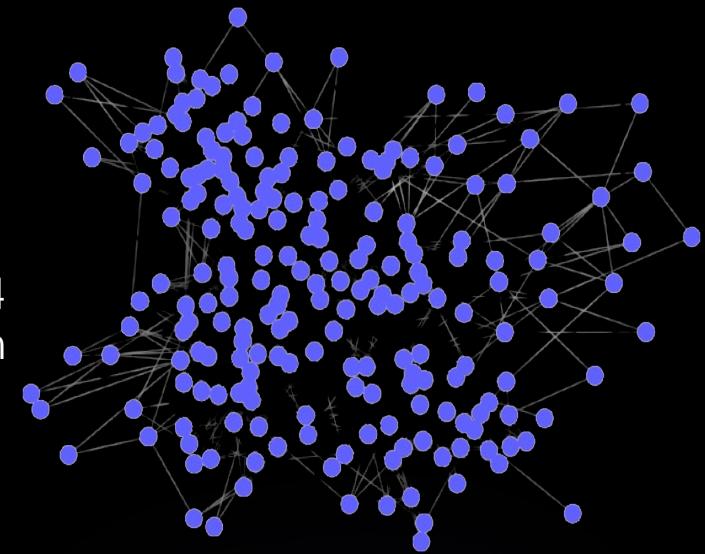
Figure 6: Top linear transform in reverse direction.

M1 = T13 x T6	M17 = M5 + T24	M33 = M27 + M25	M49 = M43 x T16
M2 = T23 x T8	M18 = M8 + M7	M34 = M21 x M22	M50 = M38 x T9
M3 = T14 + M1	M19 = M10 + M15	M35 = M24 x M34	M51 = M37 x T17
M4 = T19 x D	M20 = M16 + M13	M36 = M24 + M25	M52 = M42 x T15
M5 = M4 + M1	M21 = M17 + M15	M37 = M21 + M29	M53 = M45 x T27
M6 = T3 x T16	M22 = M18 + M13	M38 = M32 + M33	M54 = M41 x T10
M7 = T22 x T9	M23 = M19 + T25	M39 = M23 + M30	M55 = M44 x T13
M8 = T26 + M6	M24 = M22 + M23	M40 = M35 + M36	M56 = M40 x T23
M9 = T20 x T17	M25 = M22 x M20	M41 = M33 + M40	M57 = M39 x T19
M10 = M9 + M6	M26 = M21 + M25	M42 = M37 + M39	M58 = M43 x T8
M11 = T1 x T15	M27 = M20 + M21	M43 = M37 + M38	M59 = M38 x T22
M12 = T4 x T27	M28 = M23 + M25	M44 = M39 + M40	M60 = M37 x T20
M13 = M12 + M11	M29 = M28 x M27	M45 = M42 + M41	M61 = M42 x T1
M14 = T2 x T10	M30 = M26 x M24	M46 = M44 x T6	M62 = M45 x T4
M15 = M14 + M11	M31 = M20 x M23	M47 = M40 x T8	M63 = M41 x T2
M16 = M3 + M2	M32 = M27 x M31	M48 = M39 x D	



Apply belief propagation algorithm (SASCA) and recover the key

Build the huge and horrible graph from the equations



Make more than 4000 Gaussian templates (2 for each node since it's masked)



# BEING AN HONEST GUY

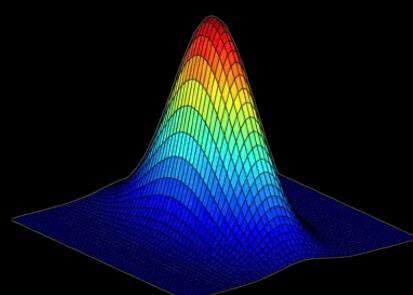
T1 = U0 + U3	T8 = U7 + T6	T15 = T5 + T11	T22 = T7 + T21
T2 = U0 + U5	T9 = U7 + T7	T16 = T5 + T12	T23 = T2 + T22
T3 = U0 + U6	T10 = T6 + T7	T17 = T9 + T16	T24 = T2 + T10
T4 = U3 + U5	T11 = U1 + U5	T18 = U3 + U7	T25 = T20 + T17
T5 = U4 + U6	T12 = U2 + U5	T19 = T7 + T18	T26 = T3 + T16
T6 = T1 + T5	T13 = T3 + T4	T20 = T1 + T19	T27 = T1 + T12
T7 = U1 + U2	T14 = T6 + T11	T21 = U6 + U7	

Figure 5: Top linear transform in forward direction.

T23 = U0 + U3	T19 = T22 + R5	T17 = U2 # T19	T6 = T22 + R17
T22 = U1 # U3	T9 = U7 # T1	T20 = T24 + R13	T16 = R13 + R19
T2 = U0 # U1	T10 = T2 + T24	T4 = U4 + T8	T27 = T1 + R18
T1 = U3 + U4	T13 = T2 + R5	R17 = U2 # U5	T15 = T10 + T27
T24 = U4 # U7	T3 = T1 + R5	R18 = U5 # U6	T14 = T10 + R18
R5 = U6 + U7	T25 = U2 # T1	R19 = U2 # U4	T26 = T3 + T16
T8 = U1 # T23	R13 = U1 + U6	Y5 = U0 + R17	

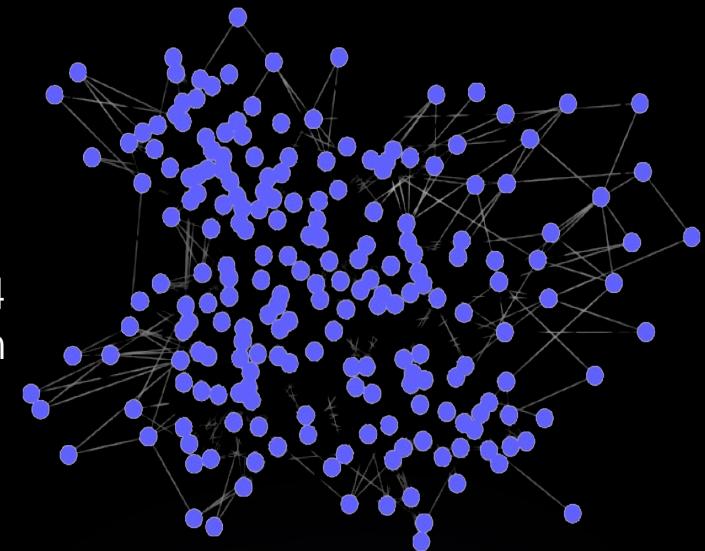
Figure 6: Top linear transform in reverse direction.

M1 = T13 x T6	M17 = M5 + T24	M33 = M27 + M25	M49 = M43 x T16
M2 = T23 x T8	M18 = M8 + M7	M34 = M21 x M22	M50 = M38 x T9
M3 = T14 + M1	M19 = M10 + M15	M35 = M24 x M34	M51 = M37 x T17
M4 = T19 x D	M20 = M16 + M13	M36 = M24 + M25	M52 = M42 x T15
M5 = M4 + M1	M21 = M17 + M15	M37 = M21 + M29	M53 = M45 x T27
M6 = T3 x T16	M22 = M18 + M13	M38 = M32 + M33	M54 = M41 x T10
M7 = T22 x T9	M23 = M19 + T25	M39 = M23 + M30	M55 = M44 x T13
M8 = T26 + M6	M24 = M22 + M23	M40 = M35 + M36	M56 = M40 x T23
M9 = T20 x T17	M25 = M22 x M20	M41 = M33 + M40	M57 = M39 x T19
M10 = M9 + M6	M26 = M21 + M25	M42 = M37 + M39	M58 = M43 x T8
M11 = T1 x T15	M27 = M20 + M21	M43 = M37 + M38	M59 = M38 x T22
M12 = T4 x T27	M28 = M23 + M25	M44 = M39 + M40	M60 = M37 x T20
M13 = M12 + M11	M29 = M28 x M27	M45 = M42 + M41	M61 = M42 x T1
M14 = T2 x T10	M30 = M26 x M24	M46 = M44 x T6	M62 = M45 x T4
M15 = M14 + M11	M31 = M20 x M23	M47 = M40 x T8	M63 = M41 x T2
M16 = M3 + M2	M32 = M27 x M31	M48 = M39 x D	



Apply belief propagation algorithm (SASCA) and get the key

Build the huge and horrible graph from the equations



Make more than 4000 Gaussian templates  
(2 for each node since it's masked)

290k traces



## BEING AN HONEST GUY

But it...

- Requires to understand a lot of theory (graphs, BP algorithm, dealing with the loops etc...)
- Is very long
- Does not even guarantee to win

# BEING AN HONEST GUY

But it...

- Requires to understand a lot of theory (graphs, BP algorithm, dealing with the loops etc...)
- Is very long
- Does not even guarantee to win



One need a 0 trace attack !

# BEING A BAD GUY

Let's use another side-channel ? Power leakage is so old shcool...

The evaluation framework  
may leak some information



**Upper bound of  $\log_2(\text{KeyRank})$**

Aggregating many well-crafted submissions may allow to extract enough information on the key



60 bits is enough !

# BEING AN BAD GUY

*How many submissions?*

- Uniform probability for all bytes except one
  - Return a different score for each of the 256 values with a uniform spacing (ex: 1, 2 ..., 256)
  - Upload the submission and store the  $\log_2(\text{KeyRank})$
- 
- Obfuscate this behind a neural network...



Average of **4.9** bits of information per submissions

$$4.9 \times 13 = 63.7$$

**Require 13 submissions !**

# BEING A BAD GUY

Read it  
backwards...

I created a new account named **Sec-artorez**

Hawai	A7_d2	200000	✗	128.0
Everest	A7_d2	210000	✗	126.7
Dubai	A7_d2	220000	✗	123.8
Inazawa	A7_d2	225000	✗	127.7
Bahamas	A7_d2	215000	✗	127.8
Zanzibar	A7_d2	200000	✗	127.0
Antarctica	A7_d2	180000	✗	127.3
Capri	A7_d2	205000	✗	128.0
Faliraki	A7_d2	220000	✗	125.2
Gaios	A7_d2	180000	✗	127.9
Jakarta	A7_d2	189000	✗	125.0
Kuala Lumpur	A7_d2	230000	✗	123.3

- First letter is a reminder for the concerned byte
- Space the submission by ~ 2 days...

Local analysis reveals that we gained 66.1 bits. Means that we should have :

$$\log_2(\text{KeyRank}) = 61.9$$

- Aggregate the results and mount the final attack.

And...

# BEING A BAD GUY

Number of traces



One\_Shot

A7\_d2

1



61.9

**Current challenger!**

*The SMAesH challenge has been SMASHED*



# SMAesH Challenge : Or how to enjoy your summer

Thomas Marquet

September 11, 2023

## Spartan-6 dataset

- Hardware masked AES with two shares ( $r$  and  $x \oplus r$ )
- No access to  $r$
- Perfectly synchronized traces
- Low SnR
- Problem : How to pick up enough signal ?
- Solution : Praying to the deep learning god

## Intermediates under attack

`bytes_to_SB[8d - 1 : 0]`  
`bytes_to_SB[16d - 1 : 8d]`  
`bytes_to_SB[24d - 1 : 16d]`  
`bytes_to_SB[32d - 1 : 24d]`

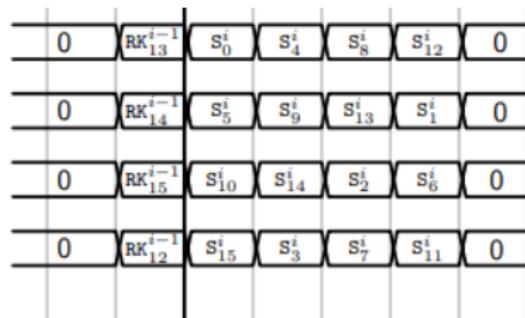


Figure: The victim

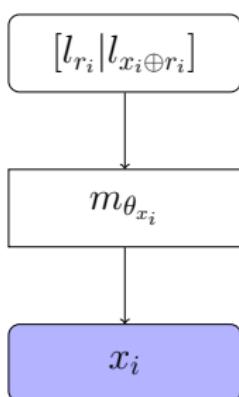
Strategy :

- Recover  $S_{12}, S_1, S_6, S_{11}$
- Recover  $S_i$  from  $S_i \oplus S_{i+4} \pmod{16}$

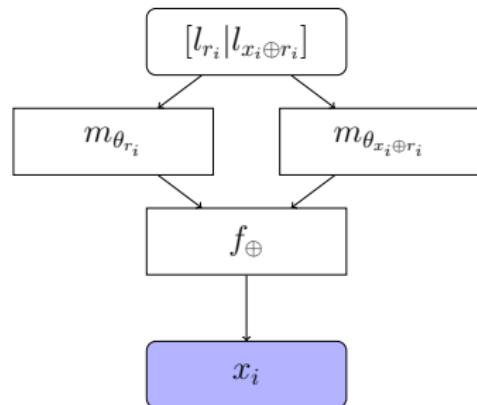
# Deep learning when randomness isn't known

## Single-task

With  $m_{\theta_{x_i}}$  the set of layers expected to fit the intermediate  $x_i$



(a) A model that do not work  
(most of the time)



(b) A model that do work  
(sometimes)

**Figure:** Hard encoding of the masking scheme inside the network

# Simply better model I swear it's not that ugly

$l_{c_0} = \text{clk } 3 \text{ to } 11$ ,  $l_{c_1} = \text{clk } 4 \text{ to } 12$ ,  $l_{c_2} = \text{clk } 5 \text{ to } 13$

$$x_i = S_i \oplus S_{i+4} \pmod{16}$$

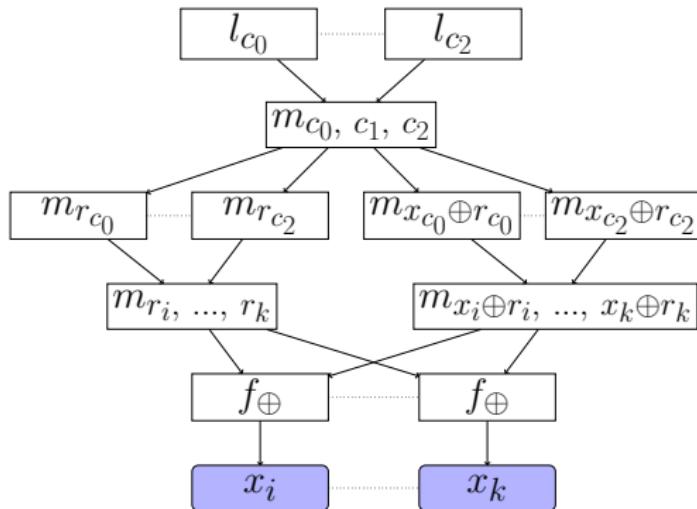


Figure: Multi-task model to recover the transitions  $x_i$

## Conclusion

- It leaks less than ASCAD
- Cross entropy go from 5.5452 to 5.5452
- Sun light is overrated

## Acknowledgments

- Supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (grant agreement No 725042)

# What's next?

---

- ▶ Secret fixed key datasets will be published.
- ▶ Leaderboard will be updated with SOTA attack.
- ▶ SMAesH public evaluation continues...
- ▶ ... And more are coming!  
→ What are you waiting for?



# SIMPLE-crypto

---

Interested? Want to participate? Question or suggestion?

- ▶ SIMPLE-crypto website

<https://www.simple-crypto.org/>

- ▶ SMAesh challenge website

<https://smaesh-challenge.simple-crypto.org/>

- ▶ Contact

[info@simple-crypto.org](mailto:info@simple-crypto.org)

(or with a beer now ;))

THANKS