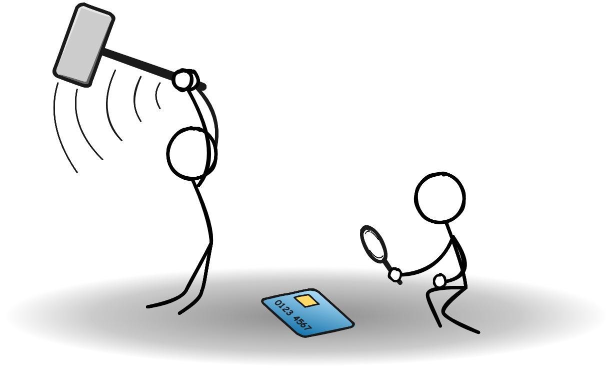
Side-Channel and Fault Analysis of Cryptographic Implementations

PhD Defense



Robert Primas

Assessors: Stefan Mangard, Joan Daemen

February 2023

Cryptography is typically used to provide:

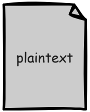
- Confidentiality
- Authenticity

Cryptography is typically used to provide:

- Confidentiality
- Authenticity

Realized using crypto algorithms and keys

- Transform plaintext to ciphertext
- Append authentication tag

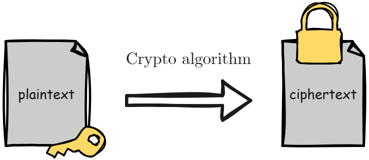


Cryptography is typically used to provide:

- Confidentiality
- Authenticity

Realized using crypto algorithms and keys

- Transform plaintext to ciphertext
- Append authentication tag

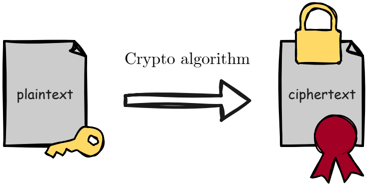


Cryptography is typically used to provide:

- Confidentiality
- Authenticity

Realized using crypto algorithms and keys

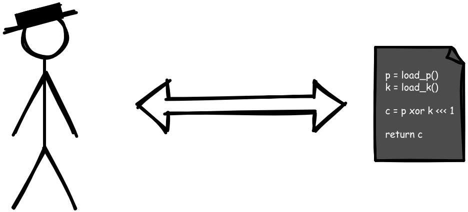
- Transform plaintext to ciphertext
- Append authentication tag



Black-box Setting

Crypto algorithms often analyzed as a "black box"

- Knowledge of algorithmic description
- Observations of non-secret inputs/outputs
- No observations of internal state

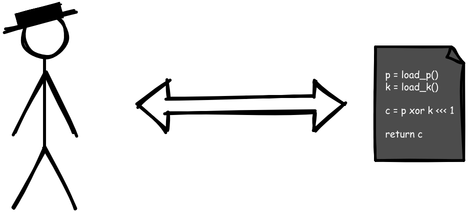


Crypto algorithms often analyzed as a “black box”

- Knowledge of algorithmic description
- Observations of non-secret inputs/outputs
- No observations of internal state

Black box is appropriate for many applications

- Ciphertexts of known/unknown plaintexts
- No direct access to devices



Gray-box Setting

Black box not appropriate for some applications

- Smart cards
- Passports
- Root-of-Trust silicon

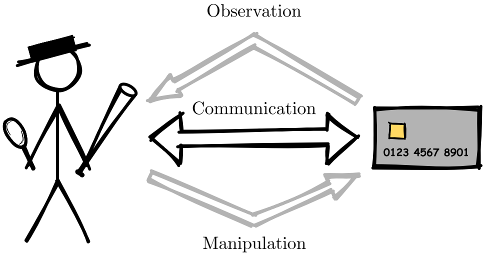
Gray-box Setting

Black box not appropriate for some applications

- Smart cards
- Passports
- Root-of-Trust silicon

Possibility of implementation attacks

- More like "gray-box" setting
- Observation of physical properties (passive)
- Tampering (active)



Presentation Outline

Part I - Passive Implementation Attacks

- **Single-Trace Side-Channel Attacks on Masked Lattice-Based Encryption** [CHES17]
- More Practical Single-Trace Attacks on the Number Theoretic Transform [LATINCRYPT19]

Part II - Active Implementation Attacks

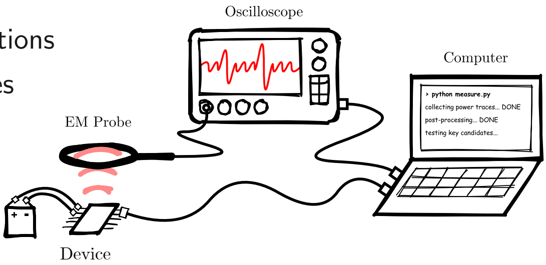
- **SIFA: Exploiting Ineffective Fault Inductions on Symmetric Cryptography** [CHES18]
- **Statistical Ineffective Fault Attacks on Masked AES with Fault Countermeasures** [ASIACRYPT18]
- **Protecting against Statistical Ineffective Fault Attacks** [CHES20]
- Fault Attacks on Nonce-Based Authenticated Encryption: Application to Keyak and Ketje [SAC18]

Other Works/Activities

Part I - Passive Implementation Attacks

Passive Implementation Attacks

- Breaking security of crypto implementations
- Observation of physical device properties
  - Electromagnetic emission
  - Power consumption
  - Photon emission
  - ...



Overview - Part I

- Power analysis attacks on implementations of PQC decryption
- Full key recovery from a single power measurement
- Applicability to protected implementations
- Improved attack method targeting encryption operations

CHES 2017 [PPM17]

Single-Trace Side-Channel Attacks on Masked Lattice-Based Encryption

Robert Primas, Peter Pessl, Stefan Mangard

Lattice-based cryptography is a promising future PQC candidate

- Conjectured to resist quantum computers
- Reasonably fast
- Okay key sizes
- 3/4 of selected candidates in NIST PQC

Lattice-based cryptography is a promising future PQC candidate

- Conjectured to resist quantum computers
- Reasonably fast
- Okay key sizes
- 3/4 of selected candidates in NIST PQC

Not a lot analysis of implementation security

- Attack techniques for RSA/ECC not really applicable

- **Key Generation:** generate small error polynomials r_1, r_2

  $p = r_1 - a \cdot r_2$

  public key $= (a, p)$

  private key $= r_2$

*variables are polynomials in $\mathbb{Z}_q[x]/\langle x^n + 1\rangle$

- **Key Generation:** generate small error polynomials r_1, r_2

  $p = r_1 - a \cdot r_2$

  public key $= (a, p)$

  private key $= r_2$

- **Encryption:** generate small error polynomials e_1, e_2, e_3

  $c_1 = a \cdot e_1 + e_2$

  $c_2 = p \cdot e_1 + e_3 + m$

*variables are polynomials in $\mathbb{Z}_q[x]/\langle x^n + 1\rangle$

Lattice-based Encryption (LPR, NewHope, Kyber, . . .)

- **Key Generation:** generate small error polynomials r_1, r_2

  $p = r_1 - a \cdot r_2$

  public key $= (a, p)$

  private key $= r_2$

- **Encryption:** generate small error polynomials e_1, e_2, e_3

  $c_1 = a \cdot e_1 + e_2$

  $c_2 = p \cdot e_1 + e_3 + m$

- **Decryption:** $m \approx c_2 - r_2 \cdot c_1$

*variables are polynomials in $\mathbb{Z}_q[x]/\langle x^n + 1\rangle$

Lattice-based Encryption (LPR, NewHope, Kyber, . . .)

- **Key Generation:** generate small error polynomials r_1, r_2

  $p = r_1 - a \cdot r_2$

  public key $= (a, p)$

  private key $= r_2$

- **Encryption:** generate small error polynomials e_1, e_2, e_3

  $c_1 = a \cdot e_1 + e_2$

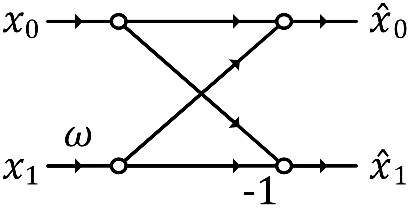
  $c_2 = p \cdot e_1 + e_3 + m$

- **Decryption:** $m \approx c_2 - r_2 \cdot c_1$

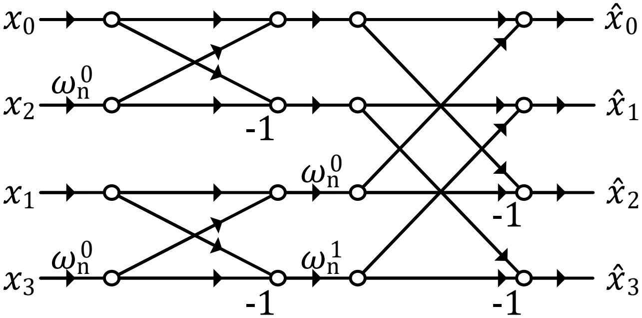
*variables are polynomials in $\mathbb{Z}_q[x]/\langle x^n + 1 \rangle$

Number Theoretic Transform

- Naive polynomial multiplication: $\mathcal{O}(n^2)$
- Better: Number Theoretic Transform (NTT)
  - $\approx$ FFT in $\mathbb{Z}_q[x]$
  - $a \cdot b = \mathrm{INTT}(\mathrm{NTT}(a) \circ \mathrm{NTT}(b))$
  - $\mathcal{O}(n \log n)$



Butterfly = 2-coefficient NTT



4-coefficient NTT

Attack Idea

- Given the ciphertext $(\hat{c}_1, \hat{c}_2)$ and private key $\hat{r}_2$, decryption is defined as:

$$m = \mathsf{INTT}(\underbrace{\hat{c}_1 \circ \hat{r}_2 + \hat{c}_2}_{\mathcal{I}_{\mathsf{INTT}}})$$

Attack Idea

- Given the ciphertext $(\hat{c}_1, \hat{c}_2)$ and private key $\hat{r}_2$, decryption is defined as:

$$m = \mathrm{INTT}(\underbrace{\hat{c}_1 \circ \hat{r}_2 + \hat{c}_2}_{\mathcal{I}_{\mathrm{INTT}}})$$

- Thus $\hat{r}_2$ can be expressed as:

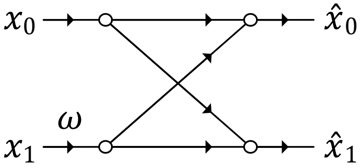
$$\hat{r}_2 = (\mathcal{I}_{\mathrm{INTT}} - \hat{c}_2) \circ \hat{c}_1^{-1}$$

Attack Strategy

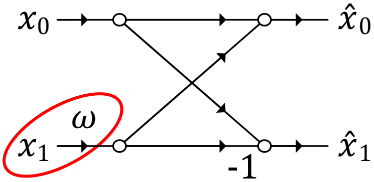
Steps:

1. Profiling of Butterfly operations
2. Leakage combination via Belief Propagation
3. Key recovery via lattice decoding

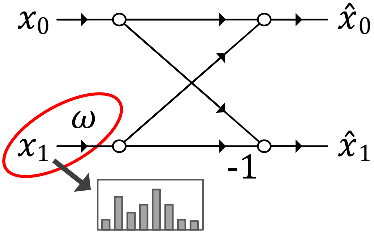
- Profiling of modular multiplication



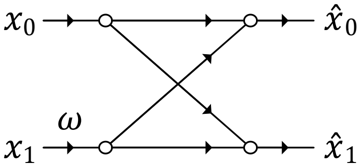
- Profiling of modular multiplication



- Profiling of modular multiplication
- Match profiles (templates)

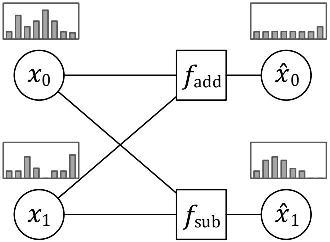


- Combination of leakage using Belief Propagation (BP) [VGS14]



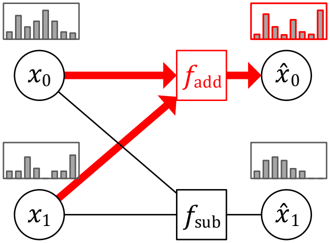
Attack - Step 2

- Combination of leakage using Belief Propagation (BP) [VGS14]
- Represent NTT as a graphical model



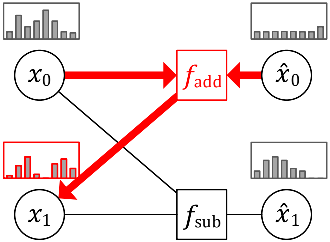
Attack - Step 2

- Combination of leakage using Belief Propagation (BP) [VGS14]
- Represent NTT as a graphical model
- Pass beliefs along edges and update



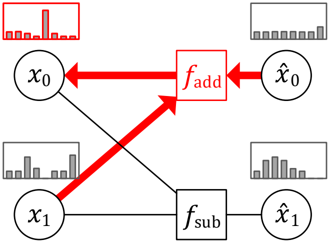
Attack - Step 2

- Combination of leakage using Belief Propagation (BP) [VGS14]
- Represent NTT as a graphical model
- Pass beliefs along edges and update



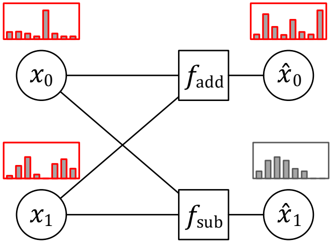
Attack - Step 2

- Combination of leakage using Belief Propagation (BP) [VGS14]
- Represent NTT as a graphical model
- Pass beliefs along edges and update



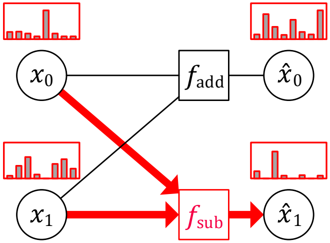
Attack - Step 2

- Combination of leakage using Belief Propagation (BP) [VGS14]
- Represent NTT as a graphical model
- Pass beliefs along edges and update

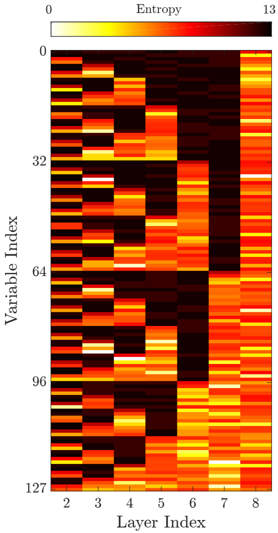


Attack - Step 2

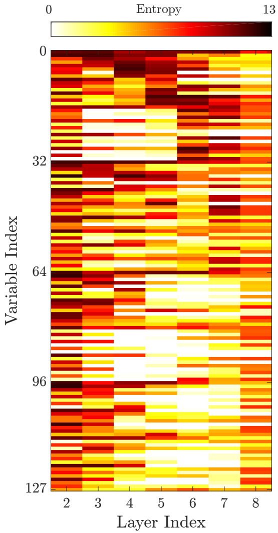
- Combination of leakage using Belief Propagation (BP) [VGS14]
- Represent NTT as a graphical model
- Pass beliefs along edges and update
- Repeat until convergence reached



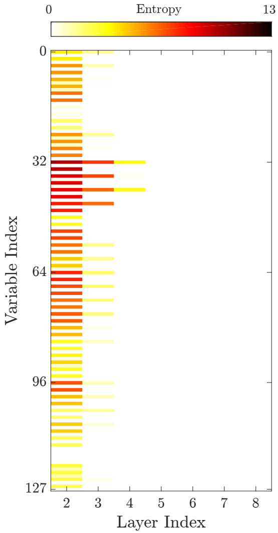
BP: Iteration 1



BP: Iteration 5



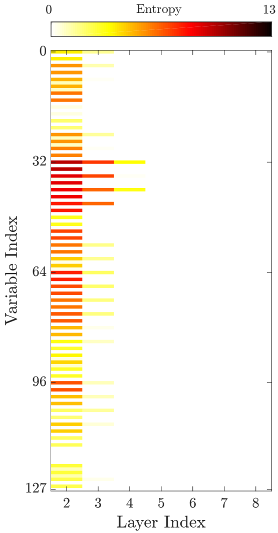
BP: Iteration ≥ 20



- Still a lot of uncertainty in the input layer
- We can exploit linearity of INTT
  to recover (in total) 192/256 inputs
- Brute forcing is still infeasible:

$$7681^{64} \approx 2^{826}$$

- Full key recovery still possible!



Step 3: Key Recovery

- Setup equations that relate 192 recovered coefficients to r_2
- Combine the equation system with the public key
- Recover r_2 by solving a reduced rank $(256 - 192 = 64)$ SVP problem
- Success rate of lattice decoding is 1

Extension to Masked Implementations

- Proposed by Reparaz et al. [Rep+16]
- Private key r_2 is split into r_2' and r_2'' s.t.:

$$\mathsf{r}_2 = \mathsf{r}_2' + \mathsf{r}_2'' \mod q$$

Extension to Masked Implementations

- Proposed by Reparaz et al. [Rep+16]
- Private key r_2 is split into r_2' and r_2'' s.t.:

$$r_2 = r_2' + r_2'' \mod q$$

- Recover 192 coefficients of one layer for both INTTs
- Perform pairwise addition of coefficients
- Proceed with Step 3

LATINCRYPT 2019 [PP19]

More Practical Single-Trace Attacks on the Number Theoretic Transform

Peter Pessl, Robert Primas

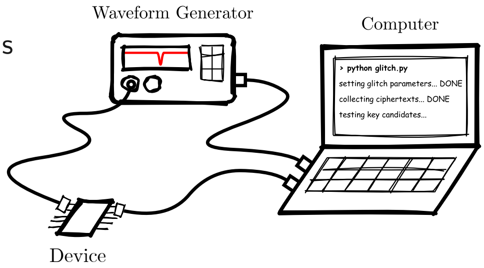
In this paper...

- Improved factor graph representation of NTT
- New message scheduling
- Usage of message damping
- Changed attack setting to encryption phase (KEM)
- Attack on real constant-time Kyber implementation on ARM Cortex-M4
- Simple univariate Hamming-weight templates

Part II - Active Implementation Attacks

Active Implementation Attacks

- Breaking security of crypto implementations
- Disturbance of normal device operation
  - Voltage glitch
  - Clock glitch
  - Laser fault induction
  - ...



Overview - Part II

- New fault attack exploitation techniques
- Particularly versatile and hard to prevent
- Largely unaffected by redundant computation or masking
- Efficient countermeasures

CHES 2018 [Dob+18b]

SIFA: Exploiting Ineffective Fault Inductions on Symmetric Cryptography

Christoph Dobraunig, Maria Eichlseder, Thomas Korak, Stefan Mangard, Florian Mendel, Robert Primas

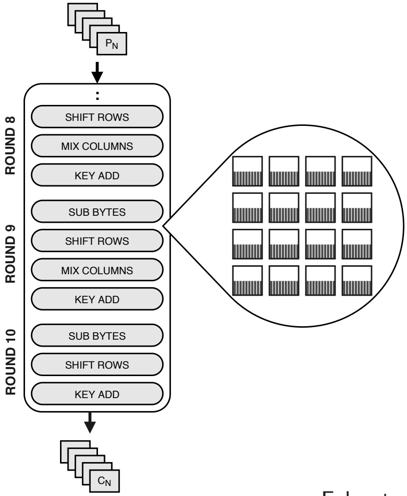
ASIACRYPT 2018 [Dob+18a]

Statistical Ineffective Fault Attacks on Masked AES with Fault Countermeasures

Christoph Dobraunig, Maria Eichlseder, Hannes Gross, Stefan Mangard, Florian Mendel, Robert Primas

AES block cipher is a PRP

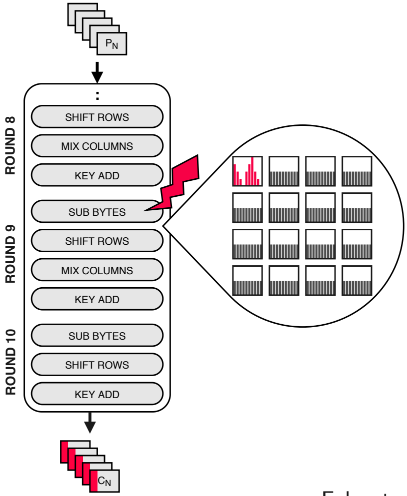
- Distribution of ciphertext bytes is uniform
- (Also after only 9 rounds)



Fuhr et al. [Fuh+13]

Assume a fault in one byte in round 9

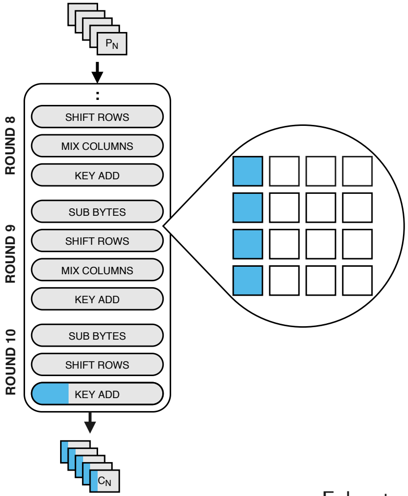
- 4 ciphertext bytes are affected



Fuhr et al. [Fuh+13]

Bias of 4 bytes in round 9 depends on:

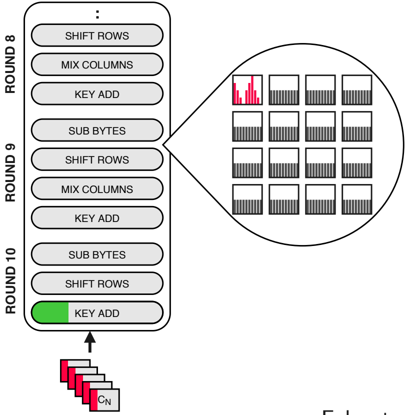
- 4 ciphertext bytes
- 4 key bytes



Fuhr et al. [Fuh+13]

Bias of 4 bytes in round 9 depends on:

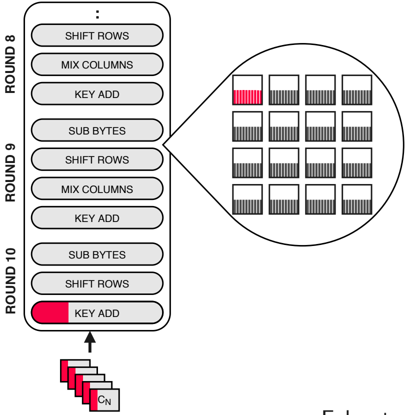
- 4 ciphertext bytes
- 4 key bytes (correct)



Fuhr et al. [Fuh+13]

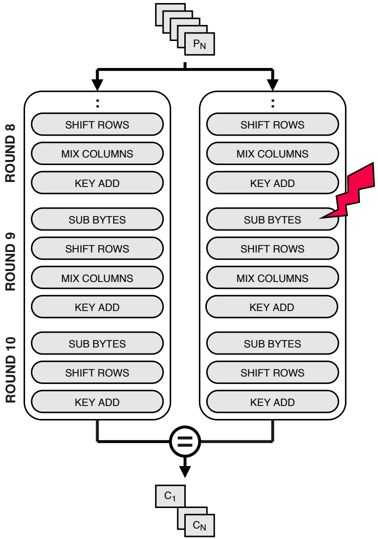
Bias of 4 bytes in round 9 depends on:

- 4 ciphertext bytes
- 4 key bytes (incorrect)

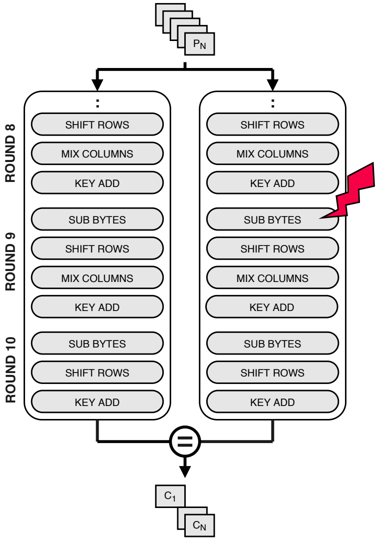


Fuhr et al. [Fuh+13]

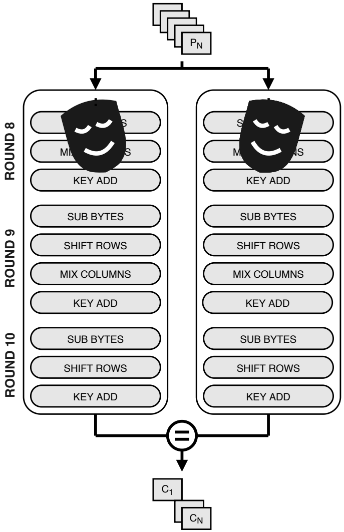
- Redundant computation fixes the problem!



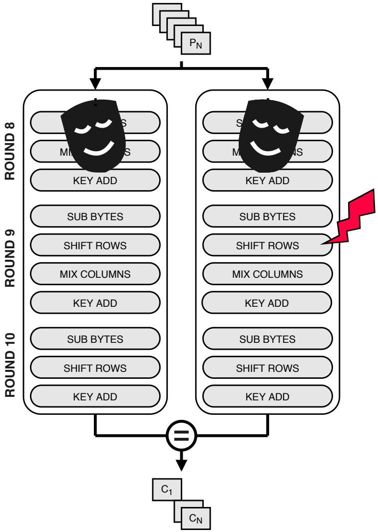
- Redundant computation fixes the problem!
- Except it doesn't
    - "Effective" faults are filtered out
    - Correct ciphertexts still show a bias
    - Exploitation works same as before



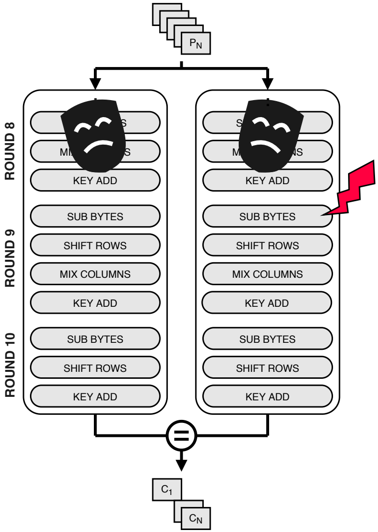
- Masking fixes the problem!



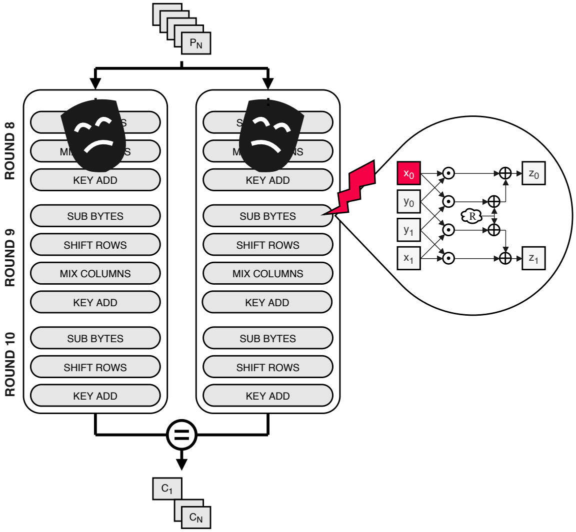
- Masking fixes the problem!



- Masking fixes the problem!
- Except it doesn't

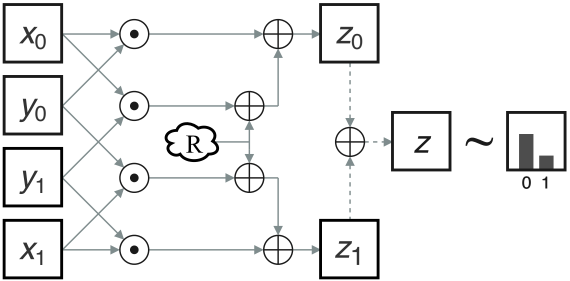


- Masking fixes the problem!
- Except it doesn't

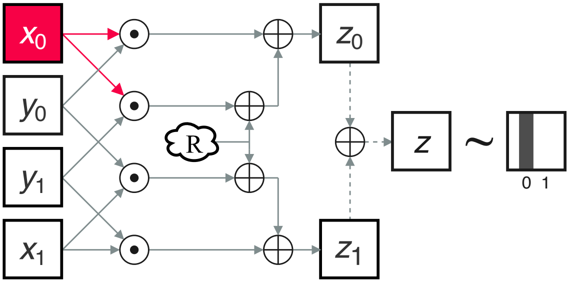


Masked AND-gate:

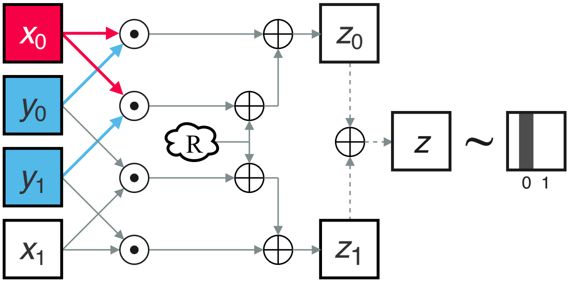
- Computes $x \times y = z$ on shares
- R indicates randomness



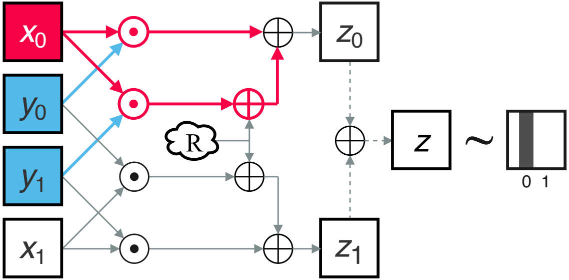
- Assume difference in x_0
  (to redundant computation)



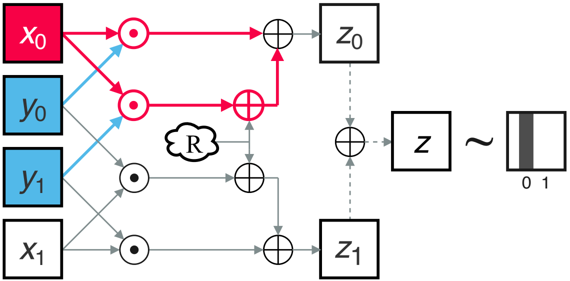
- Assume difference in x_0 (to redundant computation)
- Difference cancels if either:
  - y_0, y_1 are both 0



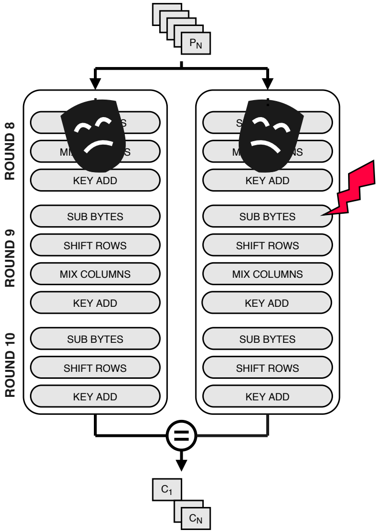
- Assume difference in x_0 (to redundant computation)
- Difference cancels if either:
  - y_0, y_1 are both 0
  - y_0, y_1 are both 1



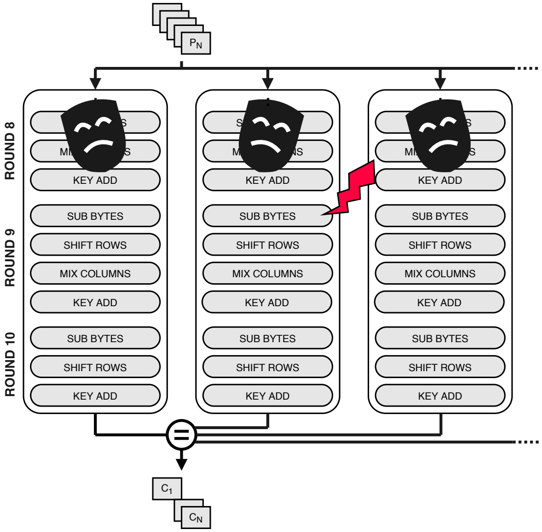
- Assume difference in x_0 (to redundant computation)
- Difference cancels if either:
  - y_0, y_1 are both 0
  - y_0, y_1 are both 1
- Fault is ineffective if y is zero



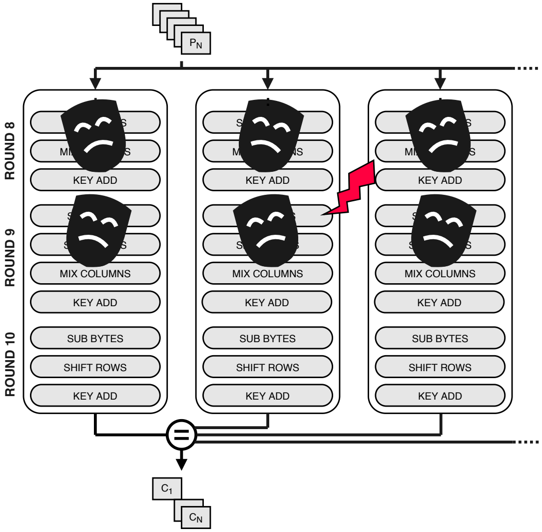
- SIFA can circumvent both masking and redundant computation



- SIFA can circumvent both masking and redundant computation
- More redundancy doesn't help



- SIFA can circumvent both masking and redundant computation
- More redundancy doesn't help
- Higher-order masking doesn't help



In the papers...

- Statistical model of SIFA
- Applicability to other fault countermeasures (infection, majority voting)
- Simulated attacks on many different crypto building blocks
- Practical evaluations for (protected) implementations on
    - Microcontrollers
    - Hardware co-processors

CHES 2020 [Dae+20]

Protecting against Statistical Ineffective Fault Attacks

Joan Daemen, Christoph Dobraunig, Maria Eichlseder, Hannes Gross, Florian Mendel, Robert Primas

The High-level Strategy

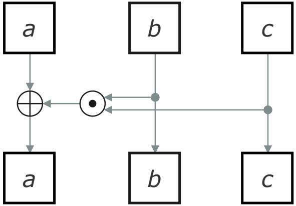
- Build cipher circuit such that "dangerous" faults can either be detected ...
  - at the S-box output
  - at the cipher output

The High-level Strategy

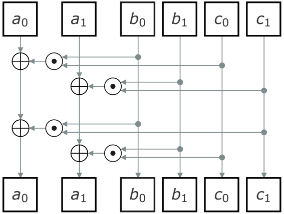
- Build cipher circuit such that “dangerous” faults can either be detected . . .
  - at the S-box output
  - at the cipher output
- Split masked cipher into “basic circuits” that . . .
  - operate on incomplete set of shares
  - are permutations

The High-level Strategy

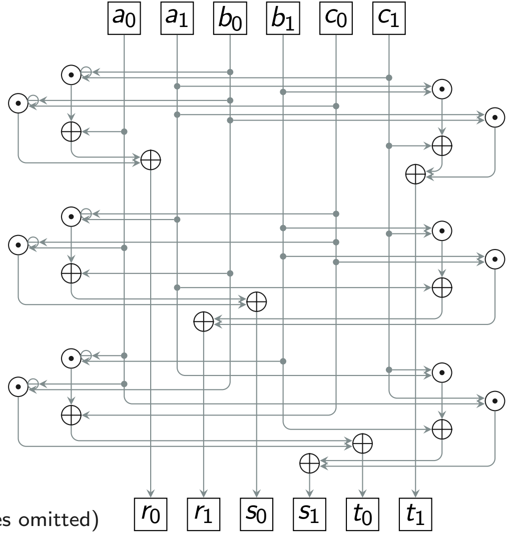
- Build cipher circuit such that "dangerous" faults can either be detected ...
  - at the S-box output
  - at the cipher output
- Split masked cipher into "basic circuits" that ...
  - operate on incomplete set of shares
  - are permutations
- Permutation can either be ...
  - a linear function
  - a variant of the Toffoli gate
    (simplest invertible non-linear function)



- Build cipher circuit such that "dangerous" faults can either be detected ...
  - at the S-box output
  - at the cipher output
- Split masked cipher into "basic circuits" that ...
  - operate on incomplete set of shares
  - are permutations
- Permutation can either be ...
  - a linear function
  - a variant of the Toffoli gate (simplest invertible non-linear function)

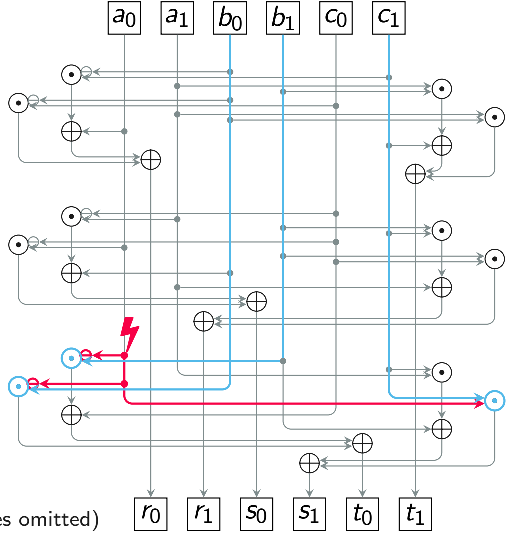


Case study: SIFA on Masked Chi-3 S-box



(Refreshing of shares omitted)

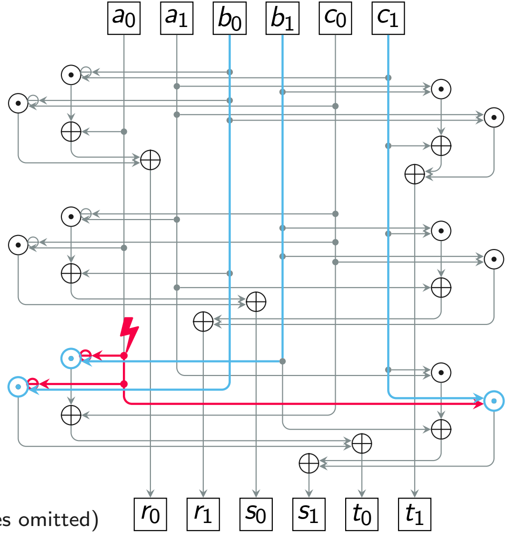
- Same problem as before . . .



(Refreshing of shares omitted)

Case study: SIFA on Masked Chi-3 S-box

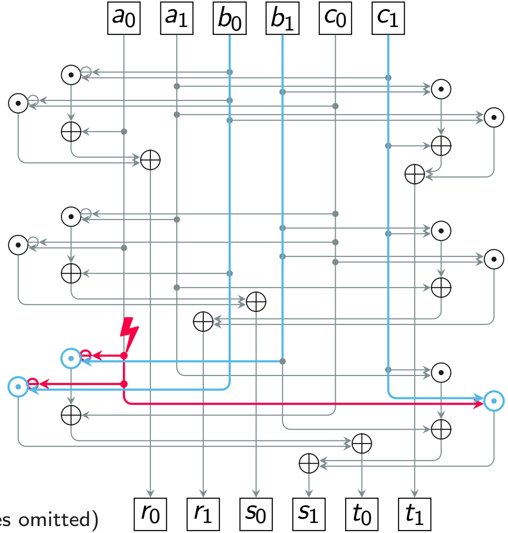
- Same problem as before ...
- Difference cancels depending on b_0, b_1 and c_1



(Refreshing of shares omitted)

Case study: SIFA on Masked Chi-3 S-box

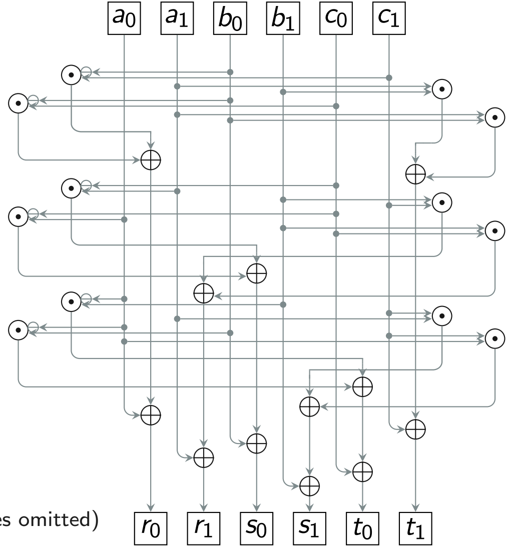
- Same problem as before . . .
- Difference cancels depending on b_0, b_1 and c_1
- If computation correct despite fault:
  - $b = 0$
  - Bias at S-box output



(Refreshing of shares omitted)

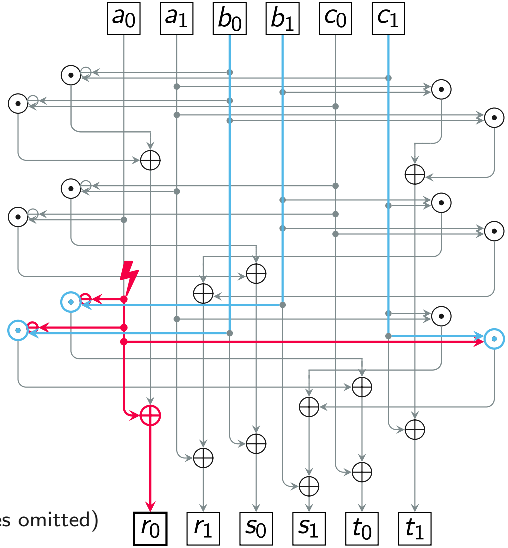
Case study: SIFA Protected Chi-3 S-box

- Basic circuits are incomplete (but not permutations)



(Refreshing of shares omitted)

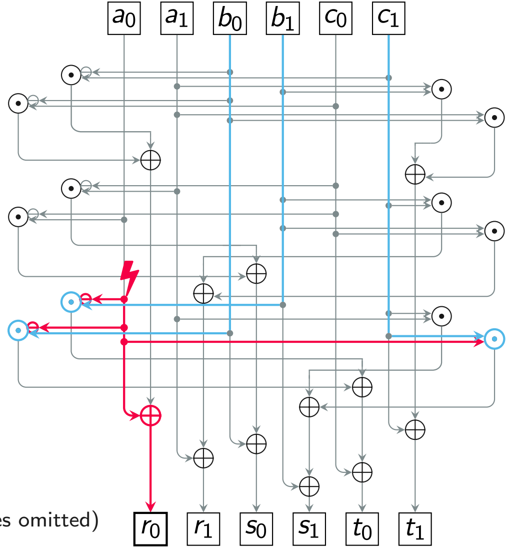
- Basic circuits are incomplete (but not permutations)
- "Dangerous" faults are always visible on the S-box output



(Refreshing of shares omitted)

Case study: SIFA Protected Chi-3 S-box

- Basic circuits are incomplete (but not permutations)
- "Dangerous" faults are always visible on the S-box output
- More precisely: Differences must not cancel based on all shares of a native variable



(Refreshing of shares omitted)

In the paper...

- We show applicability of Toffoli constructions . . .
  - for all 3-bit and many 4-bit S-boxes
  - for Chi-5-ish S-boxes and the AES S-box
- Alternative countermeasure strategy
  - Fine-grained redundancy checks
  - Protection against multi-fault SIFA (but less efficient)
- Discuss additional implementation aspects for SW/HW

SAC 2018 [Dob+18c]

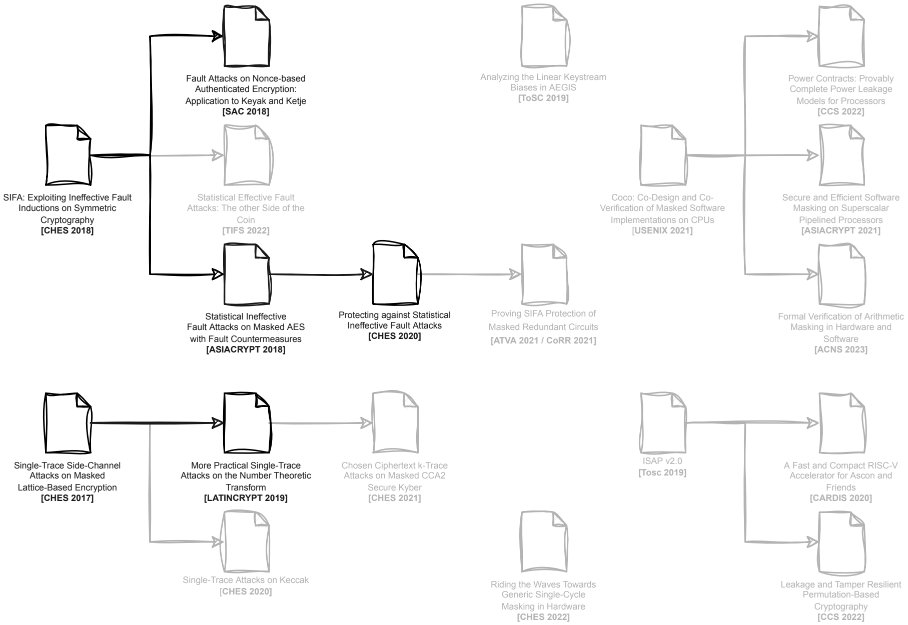
Fault Attacks on Nonce-Based Authenticated Encryption: Application to Keyak and Ketje

Christoph Dobraunig, Stefan Mangard, Florian Mendel, Robert Primas

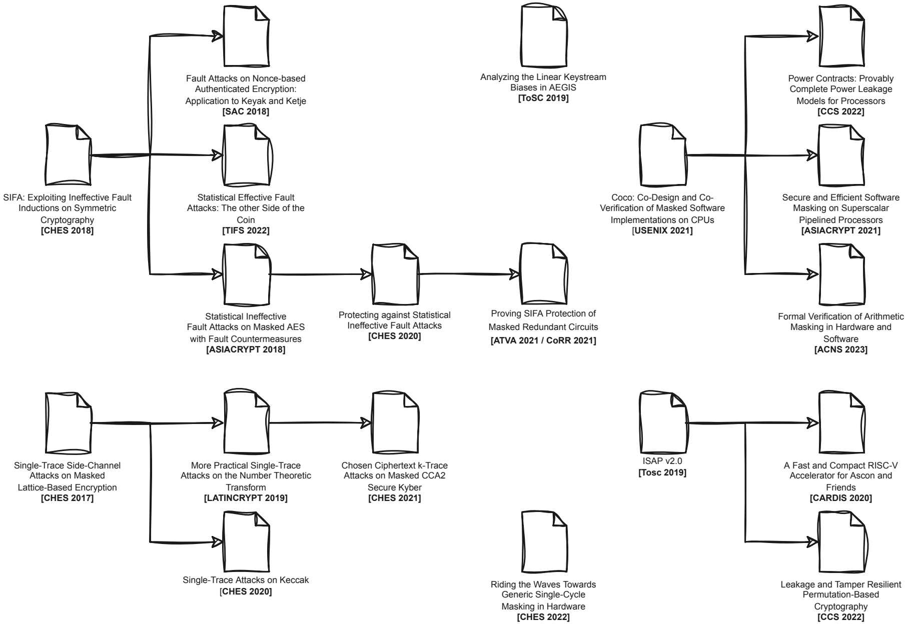
- Show applicability of SIFA for nonce-based AEAD
- Key-recovery strategies for Keyak and Ketje

Other Activities

- 20 peer-reviewed publications so far
  - 6x: CHES
  - 2x: ASIACRYPT, CCS, ToSC
  - 1x: ATVA, CARDIS, CoRR, LATINCRYPT, SAC, TIFS, USENIX
- Collaboration with 35 different co-authors



Fault Attacks on Nonce-based Authenticated Encryption: Application to Keyak and Ketje [SAC 2018]
Analyzing the Linear Keystream Biases in AEGIS [ToSC 2019]
Power Contracts: Provably Complete Power Leakage Models for Processors [CCS 2022]
SIFA: Exploiting Ineffective Fault Inductions on Symmetric Cryptography [CHES 2018]
Statistical Effective Fault Attacks: The other Side of the Coin [TIFS 2022]
Coco: Co-Design and Co-Verification of Masked Software Implementations on CPUs [USENIX 2021]
Secure and Efficient Software Masking on Superscalar Pipelined Processors [ASIACRYPT 2021]
Statistical Ineffective Fault Attacks on Masked AES with Fault Countermeasures [ASIACRYPT 2018]
Protecting against Statistical Ineffective Fault Attacks [CHES 2020]
Proving SIFA Protection of Masked Redundant Circuits [ATVA 2021 / CoRR 2021]
Formal Verification of Arithmetic Masking in Hardware and Software [ACNS 2023]
Single-Trace Side-Channel Attacks on Masked Lattice-Based Encryption [CHES 2017]
More Practical Single-Trace Attacks on the Number Theoretic Transform [LATINCRYPT 2019]
Chosen Ciphertext k-Trace Attacks on Masked CCA2 Secure Kyber [CHES 2021]
ISAP v2.0 [Tosc 2019]
A Fast and Compact RISC-V Accelerator for Ascon and Friends [CARDIS 2020]
Single-Trace Attacks on Keccak [CHES 2020]
Riding the Waves Towards Generic Single-Cycle Masking in Hardware [CHES 2022]
Leakage and Tamper Resilient Permutation-Based Cryptography [CCS 2022]



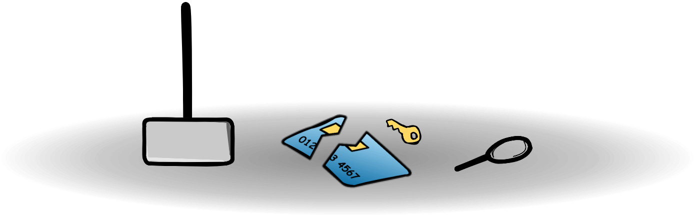
SIFA: Exploiting Ineffective Fault Inductions on Symmetric Cryptography [CHES 2018]
Fault Attacks on Nonce-based Authenticated Encryption: Application to Keyak and Ketje [SAC 2018]
Statistical Effective Fault Attacks: The other Side of the Coin [TIFS 2022]
Statistical Ineffective Fault Attacks on Masked AES with Fault Countermeasures [ASIACRYPT 2018]
Protecting against Statistical Ineffective Fault Attacks [CHES 2020]
Proving SIFA Protection of Masked Redundant Circuits [ATVA 2021 / CoRR 2021]
Analyzing the Linear Keystream Biases in AEGIS [ToSC 2019]
Coco: Co-Design and Co-Verification of Masked Software Implementations on CPUs [USENIX 2021]
Power Contracts: Provably Complete Power Leakage Models for Processors [CCS 2022]
Secure and Efficient Software Masking on Superscalar Pipelined Processors [ASIACRYPT 2021]
Formal Verification of Arithmetic Masking in Hardware and Software [ACNS 2023]
Single-Trace Side-Channel Attacks on Masked Lattice-Based Encryption [CHES 2017]
More Practical Single-Trace Attacks on the Number Theoretic Transform [LATINCRYPT 2019]
Chosen Ciphertext k-Trace Attacks on Masked CCA2 Secure Kyber [CHES 2021]
Single-Trace Attacks on Keccak [CHES 2020]
Riding the Waves Towards Generic Single-Cycle Masking in Hardware [CHES 2022]
ISAP v2.0 [Tosc 2019]
A Fast and Compact RISC-V Accelerator for Ascon and Friends [CARDIS 2020]
Leakage and Tamper Resilient Permutation-Based Cryptography [CCS 2022]

Activities

- Talks at 10+ conferences/workshops
- Supervised 15+ students during bachelor/master project/thesis
- Lectures and practicals of the SCS course
- Involved in the submission of ISAP to NIST LWC standardization
- PC member: FDTC
- Artifact evaluator: CHES
- Refereeing:

  AFRICACRYPT, ASIACRYPT, CCDS, CHES, COMJNL, COSADE, CRYPTO, CSUR, CT-RSA, EUROCRYPT, EuroS&P, MICPRO, SAC, TC, TCAD, TIFS

Thank you!



References

[Dae+20] J. Daemen, C. Dobraunig, M. Eichlseder, H. Groß, F. Mendel, and R. Primas. Protecting against Statistical Ineffective Fault Attacks. In: IACR Trans. Cryptogr. Hardw. Embed. Syst. 2020.3 (2020), pp. 508–543.

[Dob+18a] C. Dobraunig, M. Eichlseder, H. Groß, S. Mangard, F. Mendel, and R. Primas. Statistical Ineffective Fault Attacks on Masked AES with Fault Countermeasures. In: ASIACRYPT (2). Vol. 11273. Lecture Notes in Computer Science. Springer, 2018, pp. 315–342.

[Dob+18b] C. Dobraunig, M. Eichlseder, T. Korak, S. Mangard, F. Mendel, and R. Primas. SIFA: Exploiting Ineffective Fault Inductions on Symmetric Cryptography. In: IACR Trans. Cryptogr. Hardw. Embed. Syst. 2018.3 (2018), pp. 547–572.

[Dob+18c] C. Dobraunig, S. Mangard, F. Mendel, and R. Primas. Fault Attacks on Nonce-Based Authenticated Encryption: Application to Keyak and Ketje. In: SAC. Vol. 11349. Lecture Notes in Computer Science. Springer, 2018, pp. 257–277.

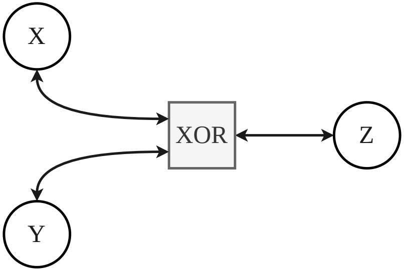
[Fuh+13] T. Fuhr, É. Jaulmes, V. Lomné, and A. Thillard. Fault Attacks on AES with Faulty Ciphertexts Only. In: FDTC. IEEE Computer Society, 2013, pp. 108–118.

[PP19] P. Pessl and R. Primas. More Practical Single-Trace Attacks on the Number Theoretic Transform. In: LATINCRYPT. Vol. 11774. Lecture Notes in Computer Science. Springer, 2019, pp. 130–149.

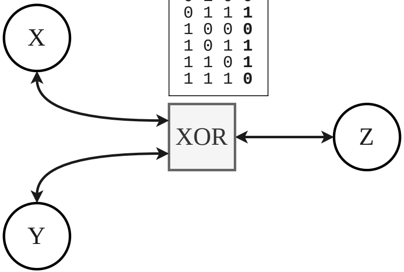
[PPM17] R. Primas, P. Pessl, and S. Mangard. Single-Trace Side-Channel Attacks on Masked Lattice-Based Encryption. In: CHES. Vol. 10529. Lecture Notes in Computer Science. Springer, 2017, pp. 513–533.

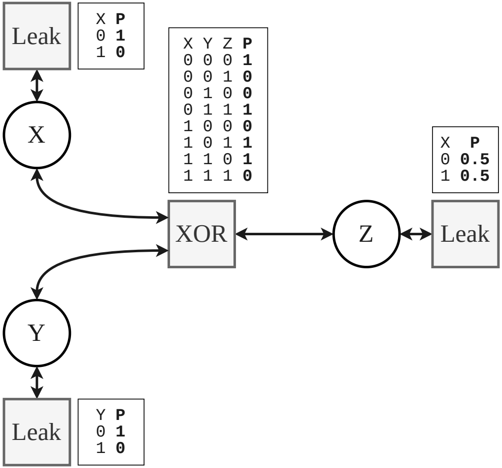
[Rep+16] O. Reparaz, S. S. Roy, R. de Clercq, F. Vercauteren, and I. Verbauwhede. Masking ring-LWE. In: J. Cryptogr. Eng. 6.2 (2016), pp. 139–153.

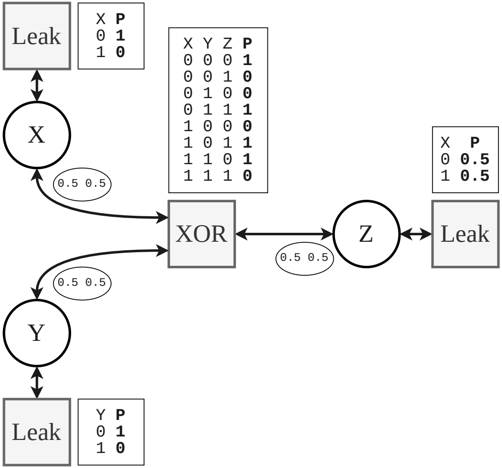
[VGS14] N. Veyrat-Charvillon, B. Gérard, and F.-X. Standaert. Soft Analytical Side-Channel Attacks. In: ASIACRYPT (1). Vol. 8873. Lecture Notes in Computer Science. Springer, 2014, pp. 282–296.

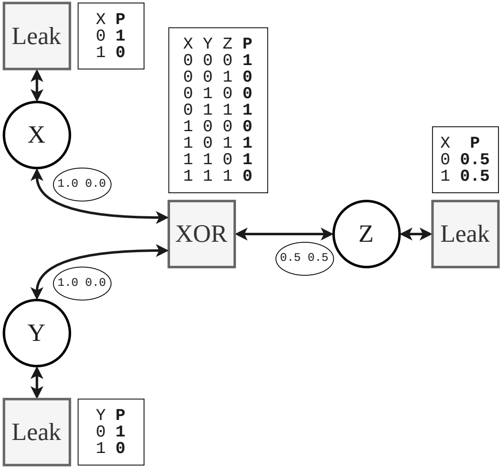

X
XOR
Z
Y

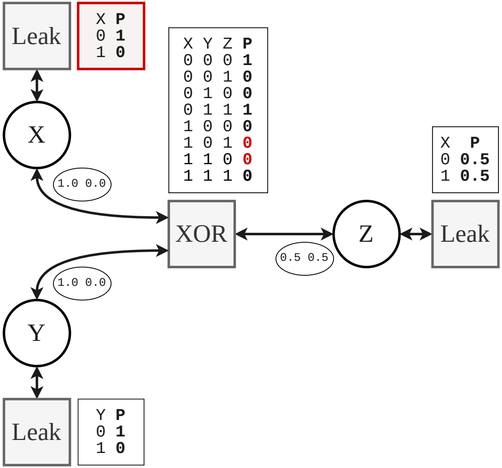
X	Y	Z	**P**
0	0	0	**1**
0	0	1	**0**
0	1	0	**0**
0	1	1	**1**
1	0	0	**0**
1	0	1	**1**
1	1	0	**1**
1	1	1	**0**

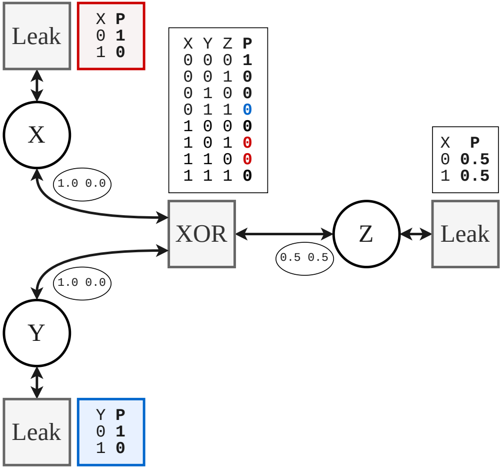


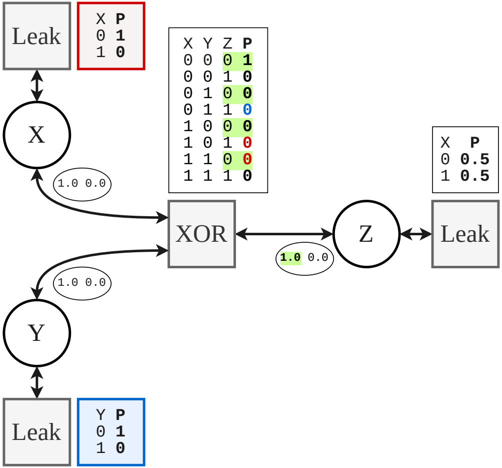

Leak
X P
0 1
1 0
X
X Y Z P
0 0 0 1
0 0 1 0
0 1 0 0
0 1 1 1
1 0 0 0
1 0 1 1
1 1 0 1
1 1 1 0
X P
0 0.5
1 0.5
XOR
Z
Leak
Y
Leak
Y P
0 1
1 0

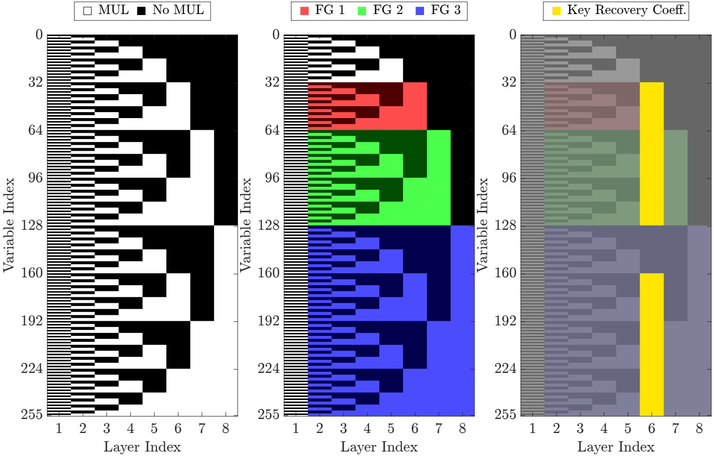

Leak
X P
0 1
1 0
X Y Z P
0 0 0 1
0 0 1 0
0 1 0 0
0 1 1 1
1 0 0 0
1 0 1 1
1 1 0 1
1 1 1 0
X
X P
0 0.5
1 0.5
0.5 0.5
XOR
Z
Leak
0.5 0.5
0.5 0.5
Y
Leak
Y P
0 1
1 0


Leak
X P
0 1
1 0
X Y Z P
0 0 0 1
0 0 1 0
0 1 0 0
0 1 1 1
1 0 0 0
1 0 1 1
1 1 0 1
1 1 1 0
X P
0 0.5
1 0.5
X
1.0 0.0
XOR
Z
Leak
0.5 0.5
1.0 0.0
Y
Leak
Y P
0 1
1 0

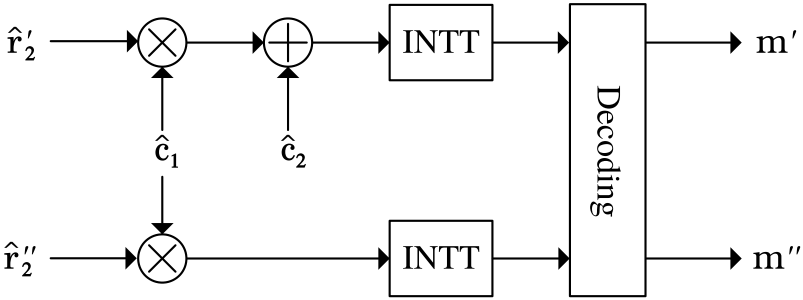

Leak
X P
0 1
1 0
X
1.0 0.0
X Y Z P
0 0 0 1
0 0 1 0
0 1 0 0
0 1 1 1
1 0 0 0
1 0 1 0
1 1 0 0
1 1 1 0
XOR
0.5 0.5
Z
Leak
X P
0 0.5
1 0.5
1.0 0.0
Y
Leak
Y P
0 1
1 0


Leak
X P
0 1
1 0
X Y Z P
0 0 0 1
0 0 1 0
0 1 0 0
0 1 1 0
1 0 0 0
1 0 1 0
1 1 0 0
1 1 1 0
X
1.0 0.0
XOR
Z
Leak
X P
0 0.5
1 0.5
0.5 0.5
1.0 0.0
Y
Leak
Y P
0 1
1 0


Leak
X P
0 1
1 0
X Y Z P
0 0 0 1
0 0 1 0
0 1 0 0
0 1 1 0
1 0 0 0
1 0 1 0
1 1 0 0
1 1 1 0
X
1.0 0.0
XOR
1.0 0.0
Z
Leak
X P
0 0.5
1 0.5
1.0 0.0
Y
Leak
Y P
0 1
1 0



MUL
No MUL
FG 1
FG 2
FG 3
Key Recovery Coeff.
Variable Index
Layer Index
0
32
64
96
128
160
192
224
255
1
2
3
4
5
6
7
8



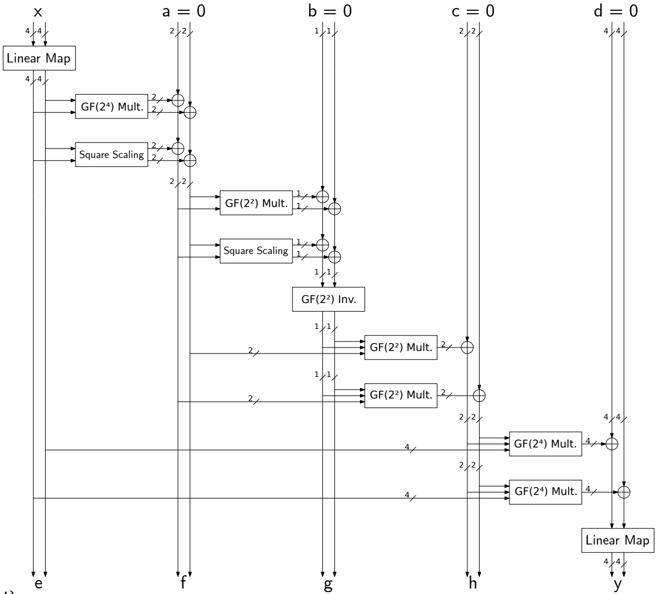
$\hat{r}_2'$
$\hat{r}_2''$
$\hat{c}_1$
$\hat{c}_2$
INTT
INTT
Decoding
m'
m''

Inputs:

- x (8-bits)
- a, b, c, d (18-bits)

Outputs:

- y (8-bits)
- e, f, g, h (18-bits)



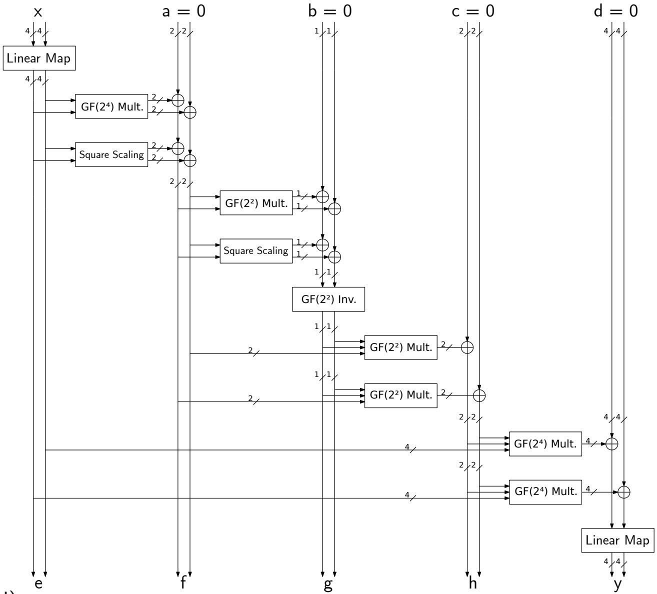
(Masking omitted)

When masked:

- x_0, x_1 (16-bits)
- y_0, y_1 (16-bits)
- a_0, b_0, c_0, d_0 (18-bits, random)
- e_0, f_0, g_0, h_0 (18-bits, reusable)

Properties:

- No additional randomness
- Checks after each S-box



(Masking omitted)

Input to	Bit positions				
	C-62-C1---9C---1	8E-12----C3----C	6-1--45----E-3-4	-384983--118---6	1---4228184--181
	C-62-C1-189C----	8E-12----C38---C	661--45----E-3--	-384982--118-3-6	1---5A28184--181
θ_1	D-62-C1---9C----	8E212----C3----C	6-1--45---3E-3--	-384986--118---6	1---4228194--181
	C-62-C1---DC----	8E-12----C34---C	6-11-45----E-3--	-3849C2--118---6	1-8-4228184--181
	C-624C1---9C----	CE-12----C3----C	6-1--45----E-3-8	-384982--118-1-6	1--44228184--181
	---------------1	8--------------1	8--------------1	8---------------	---------------1
	---------------1	8--------------1	8---------------	8---------------	---------------1
χ_1	---------------1	8--------------1	8---------------	8---------------	---------------1
	---------------1	8--------------1	8---------------	8---------------	---------------1
	---------------1	8--------------1	8---------------	8---------------	---------------1
	---------------1	8---------------	----------------	----------------	---------------1
	----------------	8---------------	----------------	----------------	---------------1
θ_2	----------------	8---------------	----------------	----------------	---------------1
	----------------	8---------------	----------------	----------------	---------------1
	----------------	8---------------	----------------	----------------	---------------1
	---------------1	----------------	----------------	----------------	----------------
	----------------	----------------	----------------	----------------	----------------
χ_2	----------------	----------------	----------------	----------------	----------------
	----------------	----------------	----------------	----------------	----------------
	----------------	----------------	----------------	----------------	----------------

Input to	Bit positions				
	ff	bf	7f	bf	fb
	fe	bf	7f	bf	fb
θ_1	fe	bf	7f	ff	fb
	fe	bf	7f	bf	fb
	fe	ff	7f	bf	ff
	-1	81	81	8-	-1
	-1	81	8-	8-	-1
χ_1	-1	81	8-	8-	-1
	-1	81	8-	8-	-1
	-1	81	8-	8-	-1
	-1	8-	--	--	-1
	--	8-	--	--	-1
θ_2	--	8-	--	--	-1
	--	8-	--	--	-1
	--	8-	--	--	-1
	-1	--	--	--	--
	--	--	--	--	--
χ_2	--	--	--	--	--
	--	--	--	--	--
	--	--	--	--	--