SIFA: Exploiting Ineffective Fault Inductions on Symmetric Cryptography

Christoph Dobraunig[1], Maria Eichlseder[1], Thomas Korak[2], Stefan Mangard[1], Florian Mendel[2], Robert Primas[1]

[1]Graz University of Technology, Austria
first.last@iaik.tugraz.at

[2]Infineon Technologies AG, Germany
first.last@infineon.com

Outlook

We present fault attacks that are ...

- Hard to prevent
  - Defy detection, any degree of redundancy
  - Defy infection
  - (Defy masking)
- Versatile
  - Many possible fault locations/effects
  - Applicable to many symmetric schemes
- Evaluated on various platforms

Outlook

We present fault attacks that are ...

- Hard to prevent
  - Defy detection, any degree of redundancy
  - Defy infection
  - (Defy masking)
- Versatile
  - Many possible fault locations/effects
  - Applicable to many symmetric schemes
- Evaluated on various platforms

Outlook

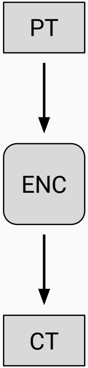
We present fault attacks that are ...

- Hard to prevent
  - Defy detection, any degree of redundancy
  - Defy infection
  - (Defy masking)
- Versatile
  - Many possible fault locations/effects
  - Applicable to many symmetric schemes
- Evaluated on various platforms

Fault Attacks

- Get device access:
  - Set plaintexts
  - Observe ciphertexts
- Cause (partially) erroneous computation
- Observe faulty and correct ciphertext
- Determine correct sub key guesses by verifying output pairs

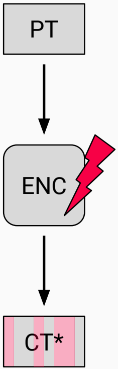
⇒ Differential Fault Attack (DFA)



Fault Attacks

- Get device access:
  - Set plaintexts
  - Observe ciphertexts
- Cause (partially) erroneous computation
- Observe faulty and correct ciphertext
- Determine correct sub key guesses by verifying output pairs

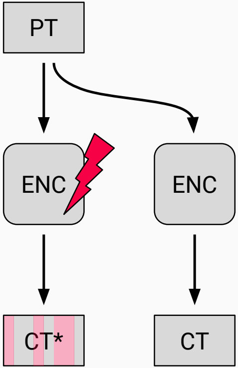
⇒ Differential Fault Attack (DFA)



Fault Attacks

- Get device access:
  - Set plaintexts
  - Observe ciphertexts
- Cause (partially) erroneous computation
- Observe faulty and correct ciphertext
- Determine correct sub key guesses by verifying output pairs

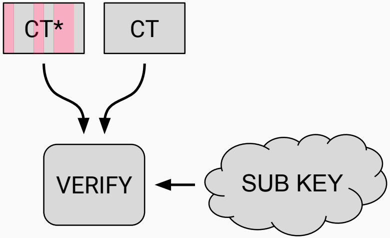
⇒ Differential Fault Attack (DFA)



Fault Attacks

- Get device access:
  - Set plaintexts
  - Observe ciphertexts
- Cause (partially) erroneous computation
- Observe faulty and correct ciphertext
- Determine correct sub key guesses by verifying output pairs

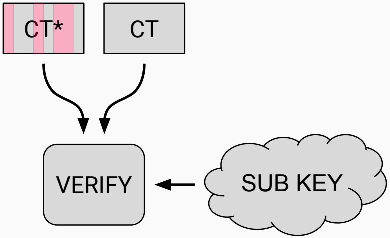
⇒ Differential Fault Attack (DFA)



Fault Attacks

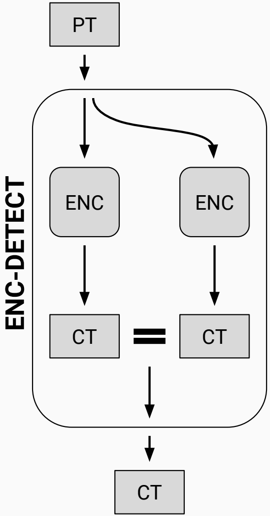
- Get device access:
  - Set plaintexts
  - Observe ciphertexts
- Cause (partially) erroneous computation
- Observe faulty and correct ciphertext
- Determine correct sub key guesses by verifying output pairs

⇒ Differential Fault Attack (DFA)



- Use redundancy to detect faults
- Fault detected → No ciphertext
- 2 identical faults necessary for attack

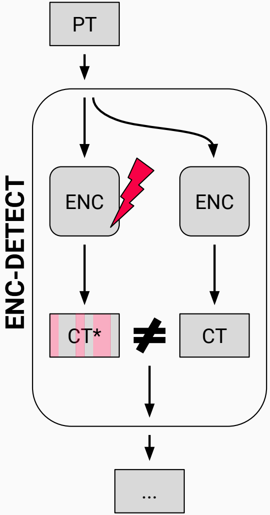
→ More redundancy, Enc-Dec, masking, etc...



Fault Countermeasures - Detection

- Use redundancy to detect faults
- Fault detected → No ciphertext
- 2 identical faults necessary for attack

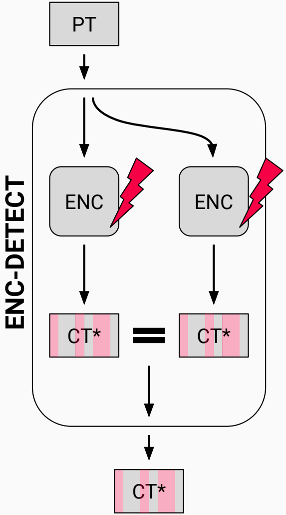
→ More redundancy, Enc-Dec, masking, etc...



Fault Countermeasures - Detection

- Use redundancy to detect faults
- Fault detected → No ciphertext
- 2 identical faults necessary for attack

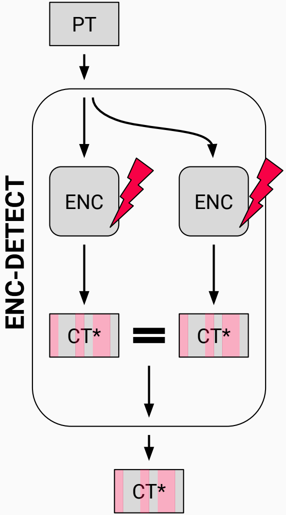
→ More redundancy, Enc-Dec, masking, etc...



Fault Countermeasures - Detection

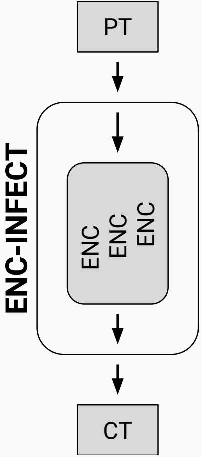
- Use redundancy to detect faults
- Fault detected → No ciphertext
- 2 identical faults necessary for attack

→ More redundancy, Enc-Dec, masking, etc...



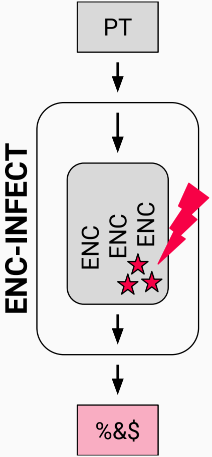
Fault Countermeasures - Infection

- Use redundancy, interleaved computation and dummy rounds
- Faults are amplified s.t. ciphertext is not related to the key anymore
- Key recovery not possible
- Attacks still possible but hard...



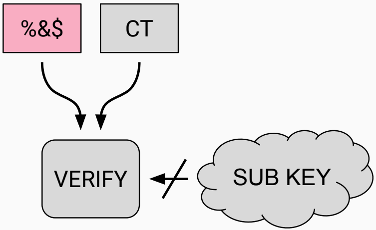
Fault Countermeasures - Infection

- Use redundancy, interleaved computation and dummy rounds
- Faults are amplified s.t. ciphertext is not related to the key anymore
- Key recovery not possible
- Attacks still possible but hard...



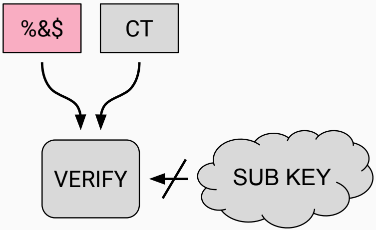
Fault Countermeasures - Infection

- Use redundancy, interleaved computation and dummy rounds
- Faults are amplified s.t. ciphertext is not related to the key anymore
- Key recovery not possible
- Attacks still possible but hard...



Fault Countermeasures - Infection

- Use redundancy, interleaved computation and dummy rounds
- Faults are amplified s.t. ciphertext is not related to the key anymore
- Key recovery not possible
- Attacks still possible but hard...



Statistical Ineffective Fault Attacks (SIFA)

Combines ...

- Ineffective Fault Attacks (IFA) by Clavier et al. [Cla07]
  - \+ Exploits only correct ciphertexts (similar to safe error attacks)
  - – Requires precise faults with known effect
- Statistical Fault Analysis (SFA) by Fuhr et al. [FJLT13]
  - \+ Any fault, even if effect is unknown
  - – Mitigated by detection/infection

⇒ Statistical Ineffective Fault Attacks (SIFA)

- \+ Exploits only correct ciphertexts
- \+ Any fault, even if effect is unknown

Statistical Ineffective Fault Attacks (SIFA)

Combines ...

- Ineffective Fault Attacks (IFA) by Clavier et al. [Cla07]
  - \+ Exploits only correct ciphertexts (similar to safe error attacks)
  - – Requires precise faults with known effect
- Statistical Fault Analysis (SFA) by Fuhr et al. [FJLT13]
  - \+ Any fault, even if effect is unknown
  - – Mitigated by detection/infection

⇒ Statistical Ineffective Fault Attacks (SIFA)

- \+ Exploits only correct ciphertexts
- \+ Any fault, even if effect is unknown

Statistical Ineffective Fault Attacks (SIFA)

Combines ...

- Ineffective Fault Attacks (IFA) by Clavier et al. [Cla07]
  - $+$ Exploits only correct ciphertexts (similar to safe error attacks)
  - $-$ Requires precise faults with known effect
- Statistical Fault Analysis (SFA) by Fuhr et al. [FJLT13]
  - $+$ Any fault, even if effect is unknown
  - $-$ Mitigated by detection/infection

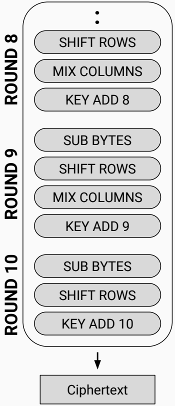
$\Rightarrow$ Statistical Ineffective Fault Attacks (SIFA)

- $+$ Exploits only correct ciphertexts
- $+$ Any fault, even if effect is unknown

SIFA on AES - Fault Injection Phase

Example for AES...

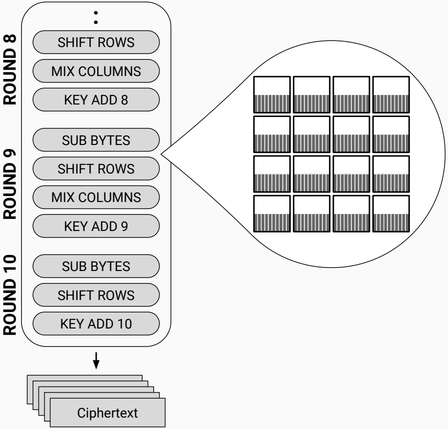
- Over multiple encryptions, state bytes are uniformly distributed
- Fault somewhere between MC in round 8-9
- Goal is some non-uniform distribution
  - Stuck-at fault, random fault, skips, flips...
  - Fault Granularity: 1 bit → a few bytes
- **Works even for ineffective faults**
  - i.e. a fault was injected but the computation is still correct
  - Attacker gets "access to subset of ciphertexts"



SIFA on AES - Fault Injection Phase

Example for AES...

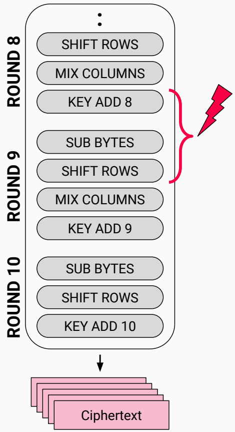
- Over multiple encryptions, state bytes are uniformly distributed
- Fault somewhere between MC in round 8-9
- Goal is some non-uniform distribution
  - Stuck-at fault, random fault, skips, flips...
  - Fault Granularity: 1 bit → a few bytes
- **Works even for ineffective faults**
  - i.e. a fault was injected but the computation is still correct
  - Attacker gets "access to subset of ciphertexts"



SIFA on AES - Fault Injection Phase

Example for AES...

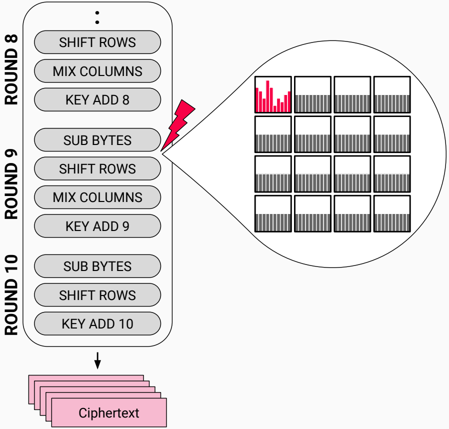
- Over multiple encryptions, state bytes are uniformly distributed
- Fault somewhere between MC in round 8-9
- Goal is some non-uniform distribution
  - Stuck-at fault, random fault, skips, flips...
  - Fault Granularity: 1 bit → a few bytes
- **Works even for ineffective faults**
  - i.e. a fault was injected but the computation is still correct
  - Attacker gets "access to subset of ciphertexts"



SIFA on AES - Fault Injection Phase

Example for AES...

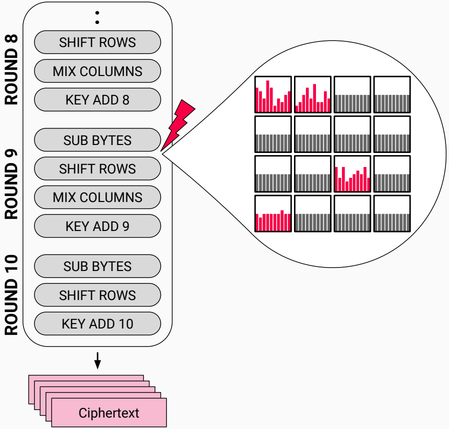
- Over multiple encryptions, state bytes are uniformly distributed
- Fault somewhere between MC in round 8-9
- Goal is some non-uniform distribution
  - Stuck-at fault, random fault, skips, flips...
  - Fault Granularity: 1 bit $\rightarrow$ a few bytes
- Works even for ineffective faults
  - i.e. a fault was injected but the computation is still correct
  - Attacker gets "access to subset of ciphertexts"



SIFA on AES - Fault Injection Phase

Example for AES...

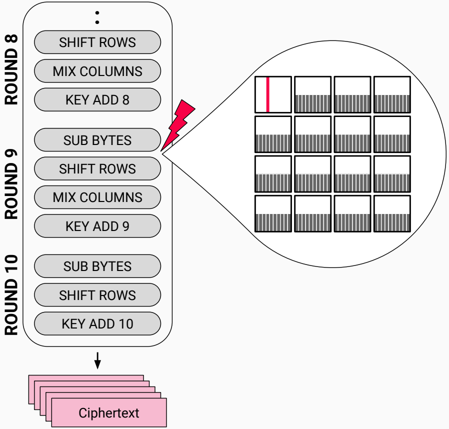
- Over multiple encryptions, state bytes are uniformly distributed
- Fault somewhere between MC in round 8-9
- Goal is some non-uniform distribution
    - Stuck-at fault, random fault, skips, flips...
    - Fault Granularity: 1 bit $\rightarrow$ a few bytes
- Works even for ineffective faults
    - i.e. a fault was injected but the computation is still correct
    - Attacker gets "access to subset of ciphertexts"



SIFA on AES - Fault Injection Phase

Example for AES...

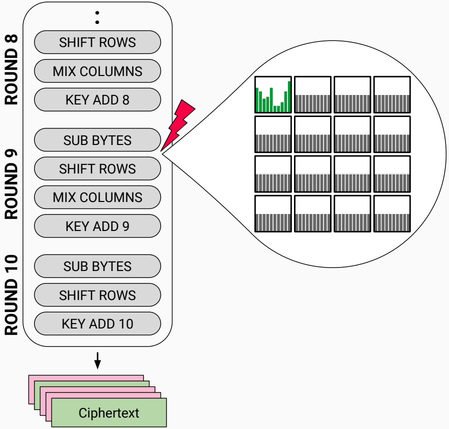
- Over multiple encryptions, state bytes are uniformly distributed
- Fault somewhere between MC in round 8-9
- Goal is some non-uniform distribution
  - Stuck-at fault, random fault, skips, flips...
  - Fault Granularity: 1 bit → a few bytes
- Works even for ineffective faults
  - i.e. a fault was injected but the computation is still correct
  - Attacker gets "access to subset of ciphertexts"



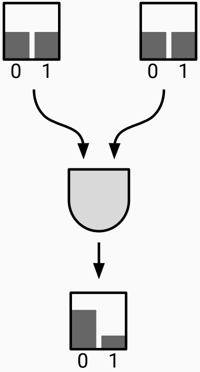
SIFA on AES - Fault Injection Phase

Example for AES...

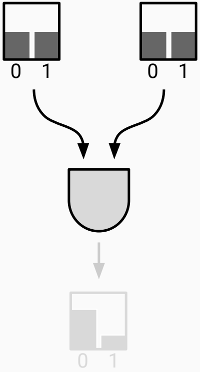
- Over multiple encryptions, state bytes are uniformly distributed
- Fault somewhere between MC in round 8-9
- Goal is some non-uniform distribution
    - Stuck-at fault, random fault, skips, flips...
    - Fault Granularity: 1 bit → a few bytes
- **Works even for ineffective faults**
    - i.e. a fault was injected but the computation is still correct
    - Attacker gets "access to subset of ciphertexts"



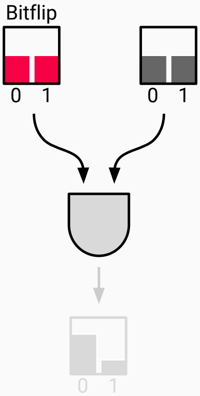
SIFA Intuition



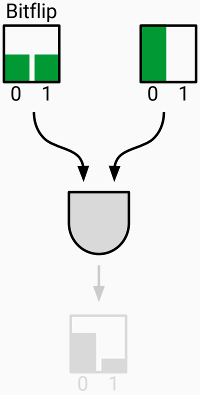
SIFA Intuition



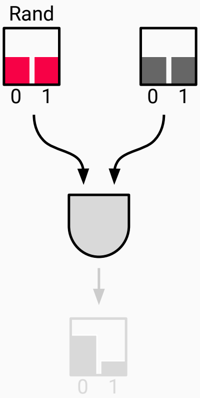
SIFA Intuition



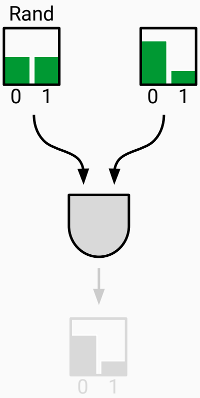
SIFA Intuition



SIFA Intuition



SIFA Intuition



SIFA on AES - Key Recovery Phase

- Collect set of correct ciphertexts $\mathcal{C}_1 \ldots \mathcal{C}_n$ from faulted encryptions
- Guess 32-bit sub key $\mathcal{K}_{10}$ and calculate state $\mathcal{S}_i$ in round 9 ($\mathcal{K}_9$ is not needed):

$$\mathcal{S}_i = \mathsf{MC}^{-1} \circ \mathsf{SB}^{-1} \circ \mathsf{SR}^{-1}(\mathcal{C}_i \oplus \mathcal{K}_{10})$$

- Measure uniformity of $\mathcal{S}_1 \ldots \mathcal{S}_n$ using e.g. the Squared Euclidean Imbalance (SEI)
- Uniform distribuiton expected for wrong key candidate
- Non-uniform distribuiton expected for correct key candidate
- Key candidate corresponding to highest SEI is likely correct

SIFA on AES - Key Recovery Phase

- Collect set of correct ciphertexts $\mathcal{C}_1 \ldots \mathcal{C}_n$ from faulted encryptions
- Guess 32-bit sub key $\mathcal{K}_{10}$ and calculate state $\mathcal{S}_i$ in round 9 ($\mathcal{K}_9$ is not needed):

$$\mathcal{S}_i = \mathsf{MC}^{-1} \circ \mathsf{SB}^{-1} \circ \mathsf{SR}^{-1}(\mathcal{C}_i \oplus \mathcal{K}_{10})$$

- Measure uniformity of $\mathcal{S}_1 \ldots \mathcal{S}_n$ using e.g. the Squared Euclidean Imbalance (SEI)
- Uniform distribuiton expected for wrong key candidate
- Non-uniform distribuiton expected for correct key candidate
- Key candidate corresponding to highest SEI is likely correct

SIFA on AES - Key Recovery Phase

- Collect set of correct ciphertexts $\mathcal{C}_1 \dots \mathcal{C}_n$ from faulted encryptions
- Guess 32-bit sub key $\mathcal{K}_{10}$ and calculate state $\mathcal{S}_i$ in round 9 ($\mathcal{K}_9$ is not needed):

$$\mathcal{S}_i = \mathsf{MC}^{-1} \circ \mathsf{SB}^{-1} \circ \mathsf{SR}^{-1}(\mathcal{C}_i \oplus \mathcal{K}_{10})$$

- Measure uniformity of $\mathcal{S}_1 \dots \mathcal{S}_n$ using e.g. the Squared Euclidean Imbalance (SEI)
- Uniform distribuiton expected for wrong key candidate
- Non-uniform distribuiton expected for correct key candidate
- Key candidate corresponding to highest SEI is likely correct

SIFA on AES - Key Recovery Phase

- Collect set of correct ciphertexts $\mathcal{C}_1 \dots \mathcal{C}_n$ from faulted encryptions
- Guess 32-bit sub key $\mathcal{K}_{10}$ and calculate state $\mathcal{S}_i$ in round 9 ($\mathcal{K}_9$ is not needed):

$$\mathcal{S}_i = \mathsf{MC}^{-1} \circ \mathsf{SB}^{-1} \circ \mathsf{SR}^{-1}(\mathcal{C}_i \oplus \mathcal{K}_{10})$$

- Measure uniformity of $\mathcal{S}_1 \dots \mathcal{S}_n$ using e.g. the Squared Euclidean Imbalance (SEI)
- Uniform distribuiton expected for wrong key candidate
- Non-uniform distribuiton expected for correct key candidate
- Key candidate corresponding to highest SEI is likely correct

SIFA on AES - Key Recovery Phase

- Collect set of correct ciphertexts $\mathcal{C}_1 \dots \mathcal{C}_n$ from faulted encryptions
- Guess 32-bit sub key $\mathcal{K}_{10}$ and calculate state $\mathcal{S}_i$ in round 9 ($\mathcal{K}_9$ is not needed):

$$\mathcal{S}_i = \mathsf{MC}^{-1} \circ \mathsf{SB}^{-1} \circ \mathsf{SR}^{-1}(\mathcal{C}_i \oplus \mathcal{K}_{10})$$

- Measure uniformity of $\mathcal{S}_1 \dots \mathcal{S}_n$ using e.g. the Squared Euclidean Imbalance (SEI)
- Uniform distribuiton expected for wrong key candidate
- Non-uniform distribuiton expected for correct key candidate
- Key candidate corresponding to highest SEI is likely correct

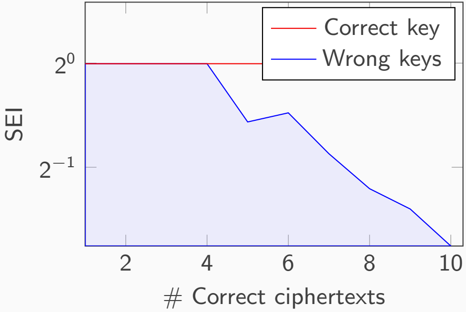
SIFA on AES - Key Recovery Phase

- Collect set of correct ciphertexts $\mathcal{C}_1 \ldots \mathcal{C}_n$ from faulted encryptions
- Guess 32-bit sub key $\mathcal{K}_{10}$ and calculate state $\mathcal{S}_i$ in round 9 ($\mathcal{K}_9$ is not needed):

$$\mathcal{S}_i = \mathsf{MC}^{-1} \circ \mathsf{SB}^{-1} \circ \mathsf{SR}^{-1}(\mathcal{C}_i \oplus \mathcal{K}_{10})$$

- Measure uniformity of $\mathcal{S}_1 \ldots \mathcal{S}_n$ using e.g. the Squared Euclidean Imbalance (SEI)
- Uniform distribuiton expected for wrong key candidate
- Non-uniform distribuiton expected for correct key candidate
- Key candidate corresponding to highest SEI is likely correct

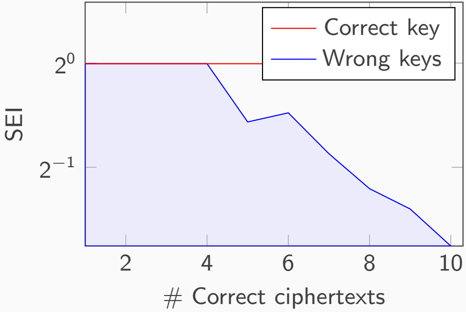
Practical Results - Detection

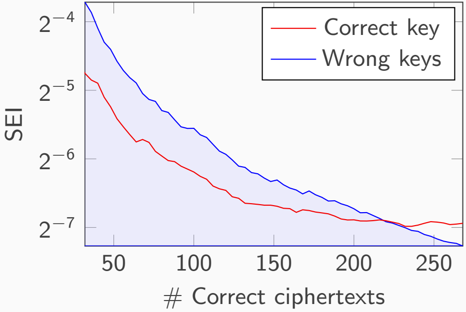


- Clock glitch on ATXmega 128D4
- SW-AES from AVR-crypto-lib
- $\approx$ **5** correct ciphertexts
- $\approx$ **1 300** faulted encryptions

- Clock glitch on ATXmega 256A3
- HW-AES co-processor
- $\approx$ **220** correct ciphertexts
- $\approx$ **1 000** faulted encryptions

Practical Results - Detection



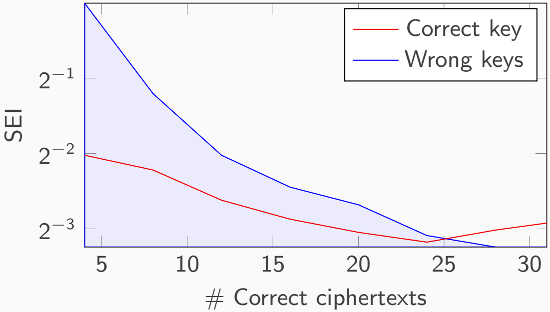


- Clock glitch on ATXmega 128D4
- SW-AES from AVR-crypto-lib
- $\approx$ **5** correct ciphertexts
- $\approx$ **1 300** faulted encryptions

- Clock glitch on ATXmega 256A3
- HW-AES co-processor
- $\approx$ **220** correct ciphertexts
- $\approx$ **1 000** faulted encryptions

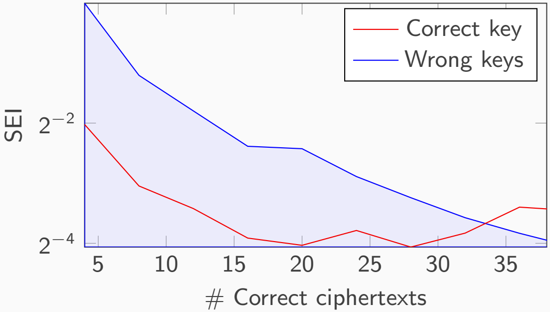
Results - Infection by Tupsamudre et al. [TBM14]

- Clock glitch: ATXmega128D4
- SW-AES with infection
- 22 real + **11** dummy rounds
- $\approx$ **25** correct ciphertexts
- $\approx$ **6 500** faulted encryptions



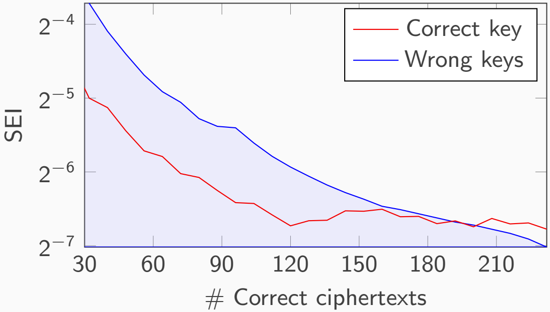
Results - Infection by Tupsamudre et al. [TBM14]

- Clock glitch: ATXmega128D4
- SW-AES with infection
- 22 real + **22** dummy rounds
- $\approx$ **34** correct ciphertexts
- $\approx$ **9 000** faulted encryptions



Results - Infection by Tupsamudre et al. [TBM14]

- Clock glitch: ATXmega128D4
- SW-AES with infection
- 22 real + **66** dummy rounds
- ≈ **180** ciphertexts needed
- ≈ **46 000** faulted encryptions



Summary

SIFA ...

- defies popular fault countermeasures: detection/infection
- requires hundreds/thousands faulted computations
- requires only one fault per computation
- does not require precise fault locations
- works with any type of fault, even if effect is unknown ($\rightarrow$ blackbox attacks)

$\Rightarrow$ works for AE schemes (SAC 2018) [DMMP18]

- $\rightarrow$ including stream-cipher, sponge-based schemes
- $\rightarrow$ e.g. all CAESAR finalists

$\Rightarrow$ works for masked implementations (ASIACRYPT 2018) [DEG+18]

- $\rightarrow$ just faulting one share is sufficient
- $\rightarrow$ same performance, no real overhead
- $\rightarrow$ essentially independent of degree of masking and redundancy

Summary

SIFA ...

- defies popular fault countermeasures: detection/infection
- requires hundreds/thousands faulted computations
- requires only one fault per computation
- does not require precise fault locations
- works with any type of fault, even if effect is unknown ($\rightarrow$ blackbox attacks)

$\Rightarrow$ works for AE schemes (SAC 2018) [DMMP18]

- $\rightarrow$ including stream-cipher, sponge-based schemes
- $\rightarrow$ e.g. all CAESAR finalists

$\Rightarrow$ works for masked implementations (ASIACRYPT 2018) [DEG+18]

- $\rightarrow$ just faulting one share is sufficient
- $\rightarrow$ same performance, no real overhead
- $\rightarrow$ essentially independent of degree of masking and redundancy

Summary

SIFA ...

- defies popular fault countermeasures: detection/infection
- requires hundreds/thousands faulted computations
- requires only one fault per computation
- does <u>not</u> require precise fault locations
- works with any type of fault, even if effect is unknown ($\rightarrow$ blackbox attacks)

$\Rightarrow$ works for AE schemes (SAC 2018) [DMMP18]

$\rightarrow$ including stream-cipher, sponge-based schemes

$\rightarrow$ e.g. all CAESAR finalists

$\Rightarrow$ works for masked implementations (ASIACRYPT 2018) [DEG$^+$18]

$\rightarrow$ just faulting one share is sufficient

$\rightarrow$ same performance, no real overhead

$\rightarrow$ essentially independent of degree of masking and redundancy

Thank you for your attention!

Christophe Clavier.
Secret external encodings do not prevent transient fault analysis.
In Pascal Paillier and Ingrid Verbauwhede, editors, *Cryptographic Hardware and Embedded Systems – CHES 2007*, volume 4727 of *LNCS*, pages 181–194. Springer, 2007.

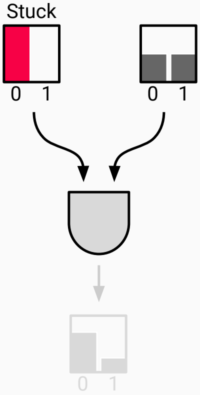
Christoph Dobraunig, Maria Eichlseder, Hannes Gross, Stefan Mangard, Florian Mendel, and Robert Primas.
Statistical ineffective fault attacks on masked AES with fault countermeasures.
Cryptology ePrint Archive, 2018.
`https://eprint.iacr.org/2018/357`.

References ii

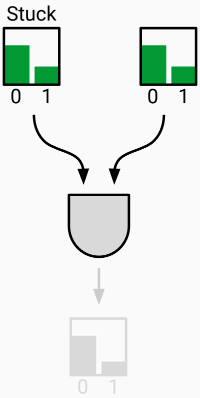
- Christoph Dobraunig, Stefan Mangard, Florian Mendel, and Robert Primas. **Fault attacks on nonce-based authenticated encryption: Application to keyak and ketje.** To appear at: Selected Areas of Cryptography, 2018.
- Thomas Fuhr, Éliane Jaulmes, Victor Lomné, and Adrian Thillard. **Fault attacks on AES with faulty ciphertexts only.** In Wieland Fischer and Jörn-Marc Schmidt, editors, *Fault Diagnosis and Tolerance in Cryptography – FDTC 2013*, pages 108–118. IEEE Computer Society, 2013.

Harshal Tupsamudre, Shikha Bisht, and Debdeep Mukhopadhyay.
Destroying fault invariant with randomization – A countermeasure for AES against differential fault attacks.
In Lejla Batina and Matthew Robshaw, editors, *Cryptographic Hardware and Embedded Systems – CHES 2014*, volume 8731 of *LNCS*, pages 93–111. Springer, 2014.

SIFA Intuition (cont.)



SIFA Intuition (cont.)



SIFA Intuition (cont.)