ONE PLUS ONE IS MORE THAN TWO

A PRACTICAL COMBINATION OF POWER AND FAULT ANALYSIS ATTACKS ON PRESENT AND PRESENT-LIKE BLOCK CIPHERS

Sikhar Patranabis, Debdeep Mukhopadhyay

Department of Computer Science and Engineering, IIT Kharagpur, India

Jakub Breier, Shivam Bhasin

Temasek Labs, Nanyang Technological University, Singapore

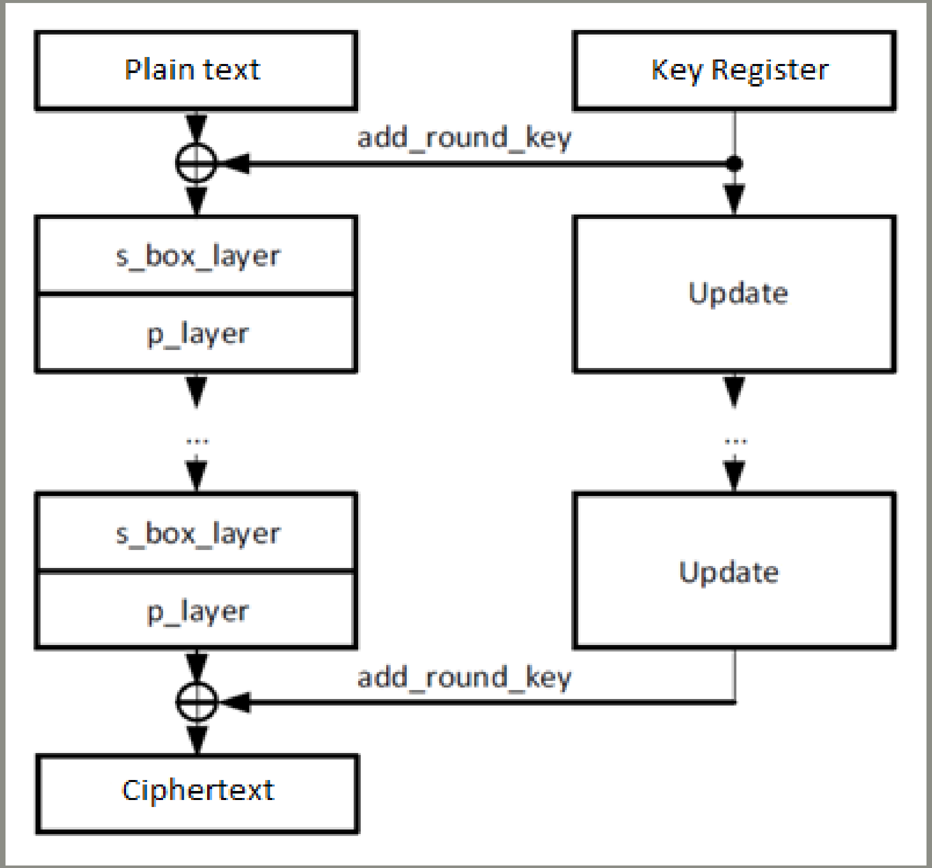
25 September, 2017 FDTC 2017 Taipei, Taiwan

BACKGROUND AND RELATED WORK

Lightweight Cryptography

- A subfield of cryptography that aims to provide solutions tailored for resource-constrained devices, typically encountered in IoT applications
- Targets a wide variety of resource-constrained devices at the lower end of the hardware/software spectrum
  - Embedded Systems
  - RFID and Sensor Networks
- Typically optimize area/power/energy requirements (depends on the target platform)
  - Look-Up Tables for FPGA implementations
  - Gate Equivalent for ASIC implementations
  - Register Count and RAM/ROM bytes used in Software implementations

The PRESENT Block Cipher



Bogdanov, Andrey, Lars R. Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew JB Robshaw, Yannick Seurin, and Charlotte Vikkelsoe. "PRESENT: An ultra-lightweight block cipher." In CHES, vol. 4727, pp. 450-466. 2007.

- One of the foremost lightweight block ciphers
- Substitution-Permutation Network (SPN) with 31 rounds
- Plaintext Block Size: 64 bits
- Key Size: 80/ 128 bits
- Non-Linear Layer: One 4x4 S-Box per nibble
  - Designed to consume low area in hardware
- Linear Layer: Bit Permutations
  - Provides optimal diffusion across 3 rounds
  - Zero area overhead in hardware
- Notable for its compact size in hardware (about 2.5 times smaller than the Advanced Encryption Standard)
- Currently standardized by the IOS and IEC for lightweight cryptographic applications

Existing Fault Attacks on PRESENT

All attacks listed below typically recover the last round key of PRESENT-80

Differential Fault Analysis (Bit/Nibble Faults)	Wang et al. (2010)	Fault Injection Instances: 64
		Key Recovery Complexity: 2^{29}
	Zhao et al. (2012)	Fault injection instances: 16
		Key Recovery Complexity: $2^{21.1}$
	Bagheri et al. (2013)	Fault Injection Instances: 18
		Key Recovery Complexity: 2^{16}
Differential Fault Analysis (Hardware Trojan-Horse)	Breier and He (2015)	Multiple Fault Attack
		Fault Injection Instances: 2
		Each Fault Instance targets 4 nibbles
		Key Recovery Complexity: 2^{16}
Differential Fault Intensity Analysis (Bit/Nibble Faults)	Ghalaty et al. (2015)	Requires only Faulty Ciphertexts
		Fault Injection Instances: 10-36
		Key Recovery Complexity: 2^{16}

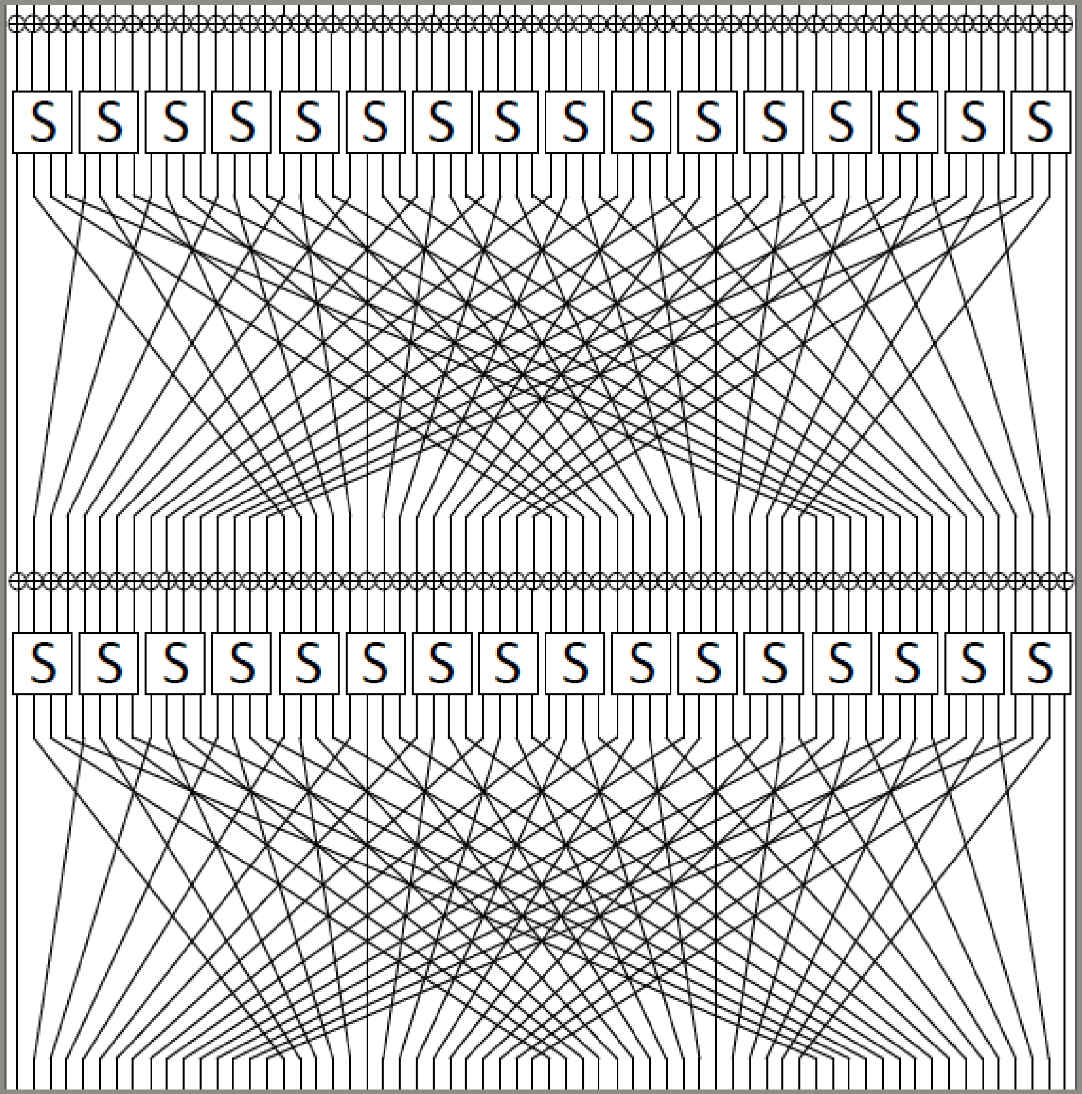
A MORE EFFICIENT FAULT ATTACK ON PRESENT

Combining Differential Power Analysis with Differential Fault Analysis to Reduce the Number of Fault Injections

A Combined Side-Channel and Fault Analysis Attack on PRESENT

- We propose the first practically realizable combination of differential power analysis (DPA) and differential fault analysis (DFA) on PRESENT
- The combination works as follows:
  - The adversary injects a random nibble fault in the target round during an encryption operation of PRESENT
  - The adversary simultaneously monitors the power leakage from the algorithm execution to determine the corresponding **output fault mask**
- The knowledge of the fault mask is then used to trace the fault propagation across the subsequent encryption rounds
- Finally, the key is recovered nibble-wise using the knowledge of:
  - The fault propagation characteristics till the penultimate round
  - The differential between the correct and faulty ciphertext pairs

The Bit-Permutation Layer of PRESENT



- The **input of an S-Box** in round r comprises output bits from **four different S-Boxes** in round $r - 1$.
- The **output of an S-Box** in round r is distributed across the **inputs of four different S-Boxes** in round $r + 1$.
- Let $n, d, l \in \{0,1,2,3\}$. Consider the following bits for some round r:
  - **Bit-A**: *The l^{th} bit in the* ***output*** *of S-Box $4n + d$ in round* r
  - **Bit-B:** *The d^{th} bit in the* ***input*** *of S-Box $n + 4l$ in round $r + 1$*
- As per the permutation layer of PRESENT, the **Bit-A**, upon XOR-ing with the corresponding round key-bit, essentially transforms into the **Bit-B**
- **This observation plays a crucial role in the attack procedure described subsequently**

Fault Model and Fault Location

- **Fault Model:** Single Nibble Fault
- **Fault Timing:** Encryption Round 28
- **Fault Nature:** Random Fault

- **Figure-1 :** Depicts a fault injection scenario with output fault mask in round 28 = 0001
- **Figure-2** : Depicts a fault injection scenario with output fault mask in round 28 = 0011

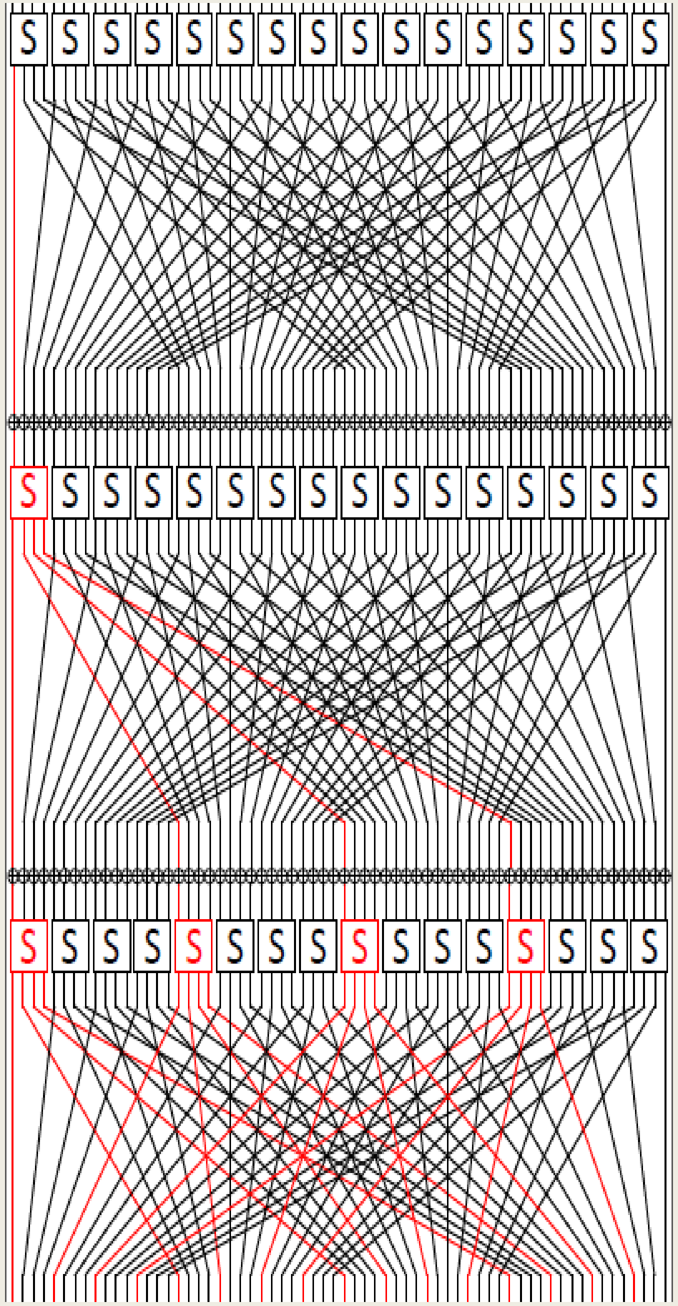


Figure-1

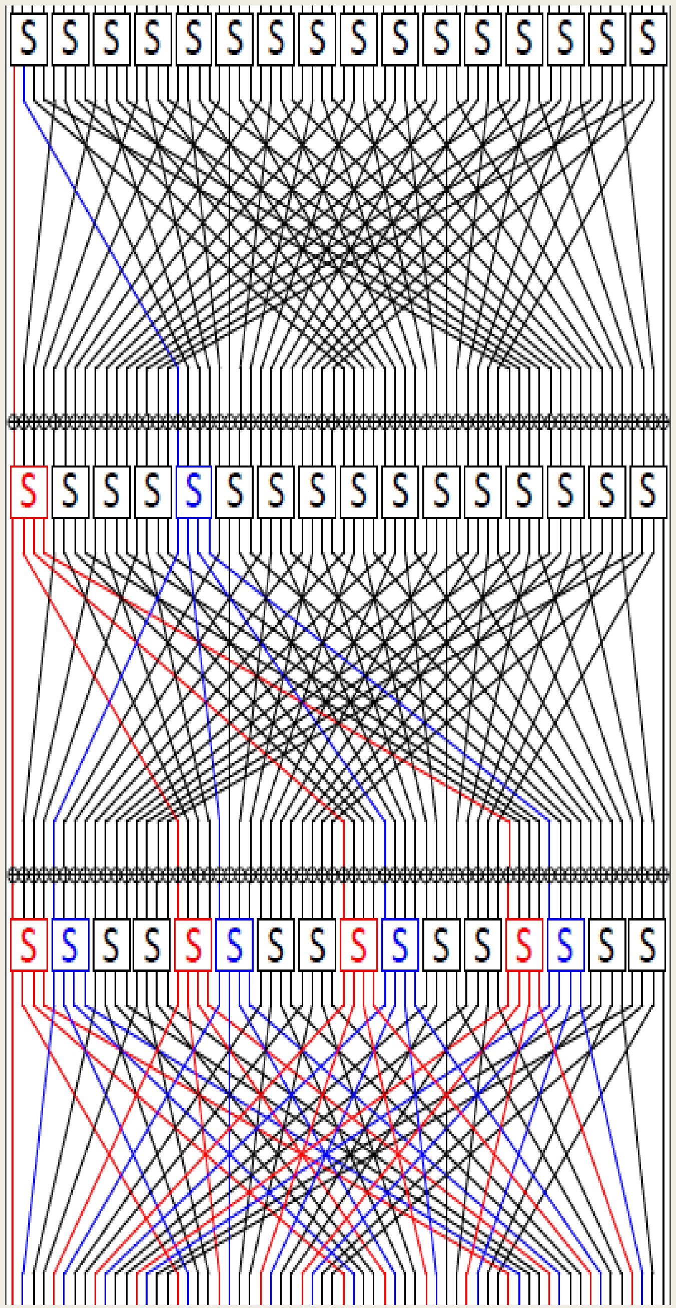
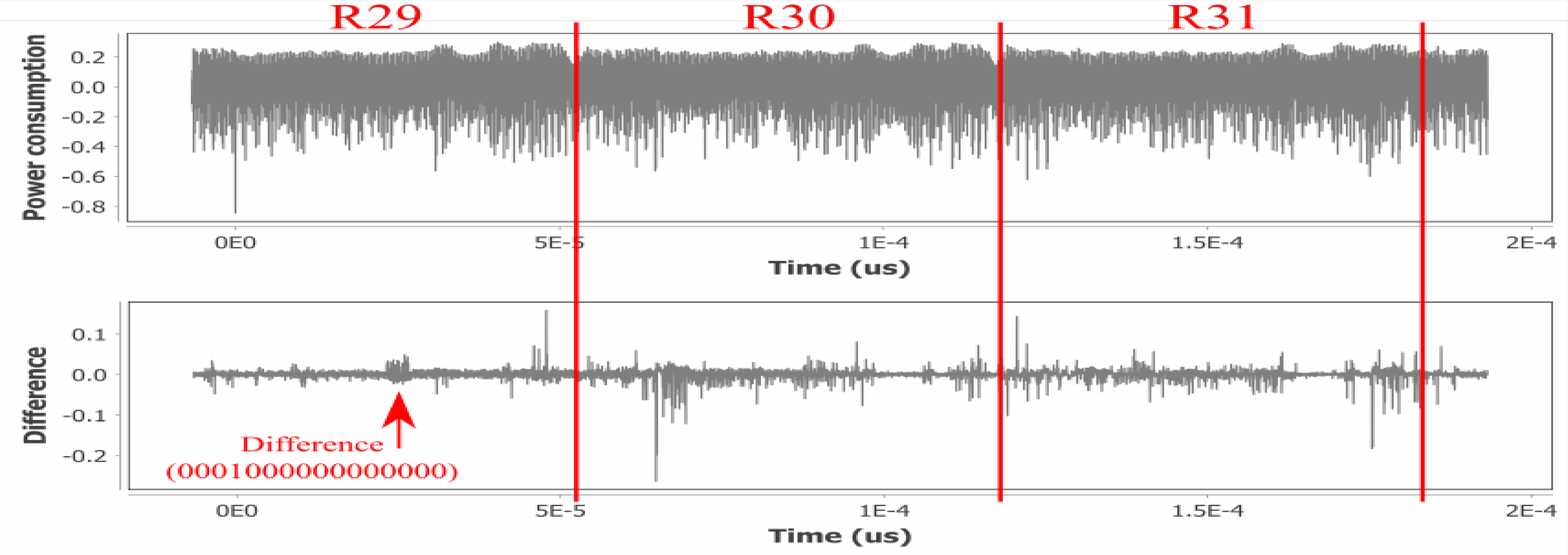


Figure-2

Role of Differential Power Analysis

- DPA is used to determine precisely the output fault mask in round 28 of PRESENT
- Since the fault model is random, the output fault mask is not pre-determined
- The fault mask is retrieved via a differential analysis between the power traces for the fault-free and faulty nibble operations in round 29 of PRESENT

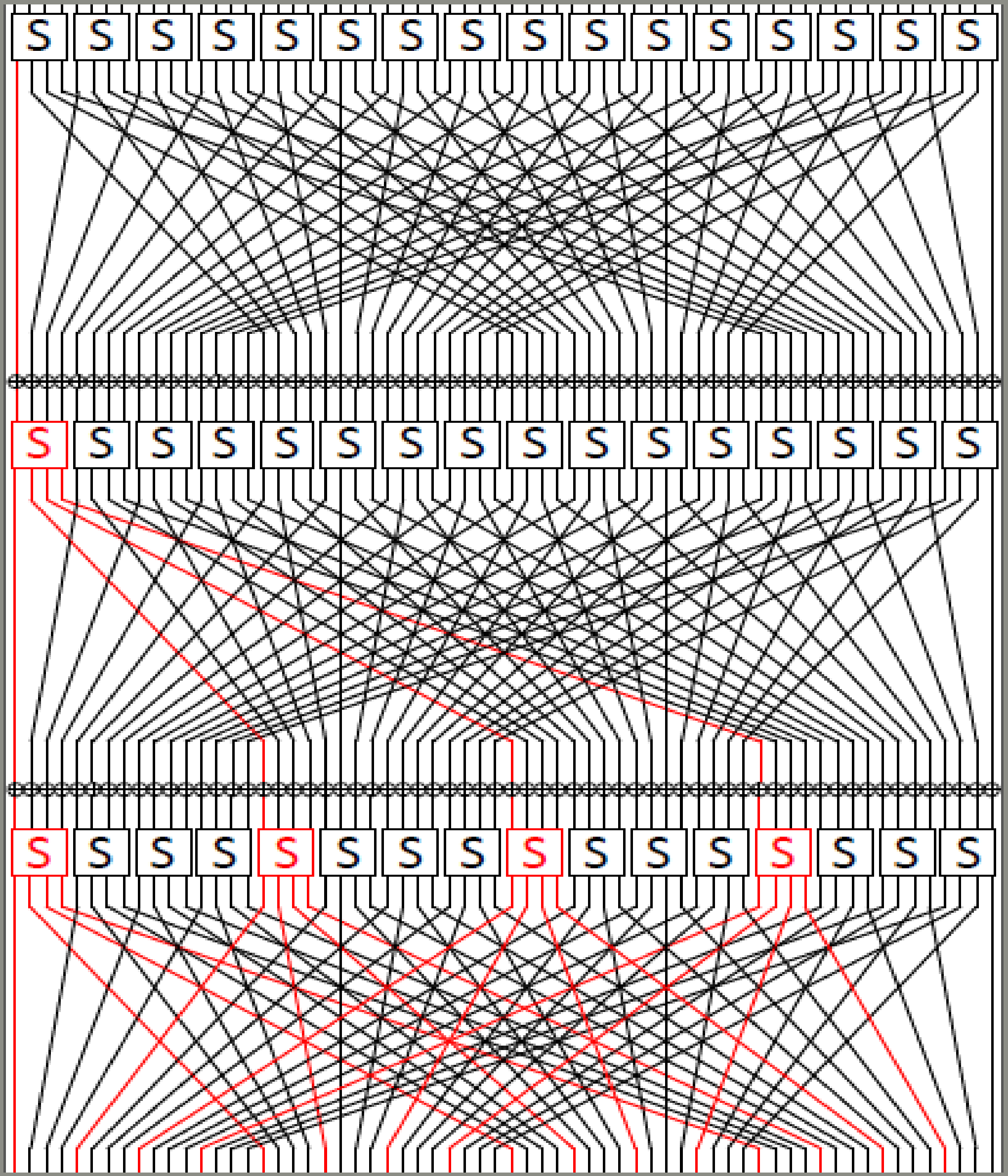


The Fault Propagation Characteristics

- **Theorem-1 :**
  - *Suppose the Hamming weight of the output fault mask of the target nibble in round 28 of PRESENT is x, where $x \in \{0,1,2,3,4\}$.*
  - ***Then, the Hamming Weight of the input fault mask of any nibble in round 31 is at most x.***

- **Theorem-2 :**
  - *Suppose the Hamming weight of the output fault mask of the target nibble in round 28 of PRESENT is x, where $x \in \{0,1,2,3,4\}$.*
  - ***Then, the input fault mask of any nibble in round 31 takes at most 2^x values.***

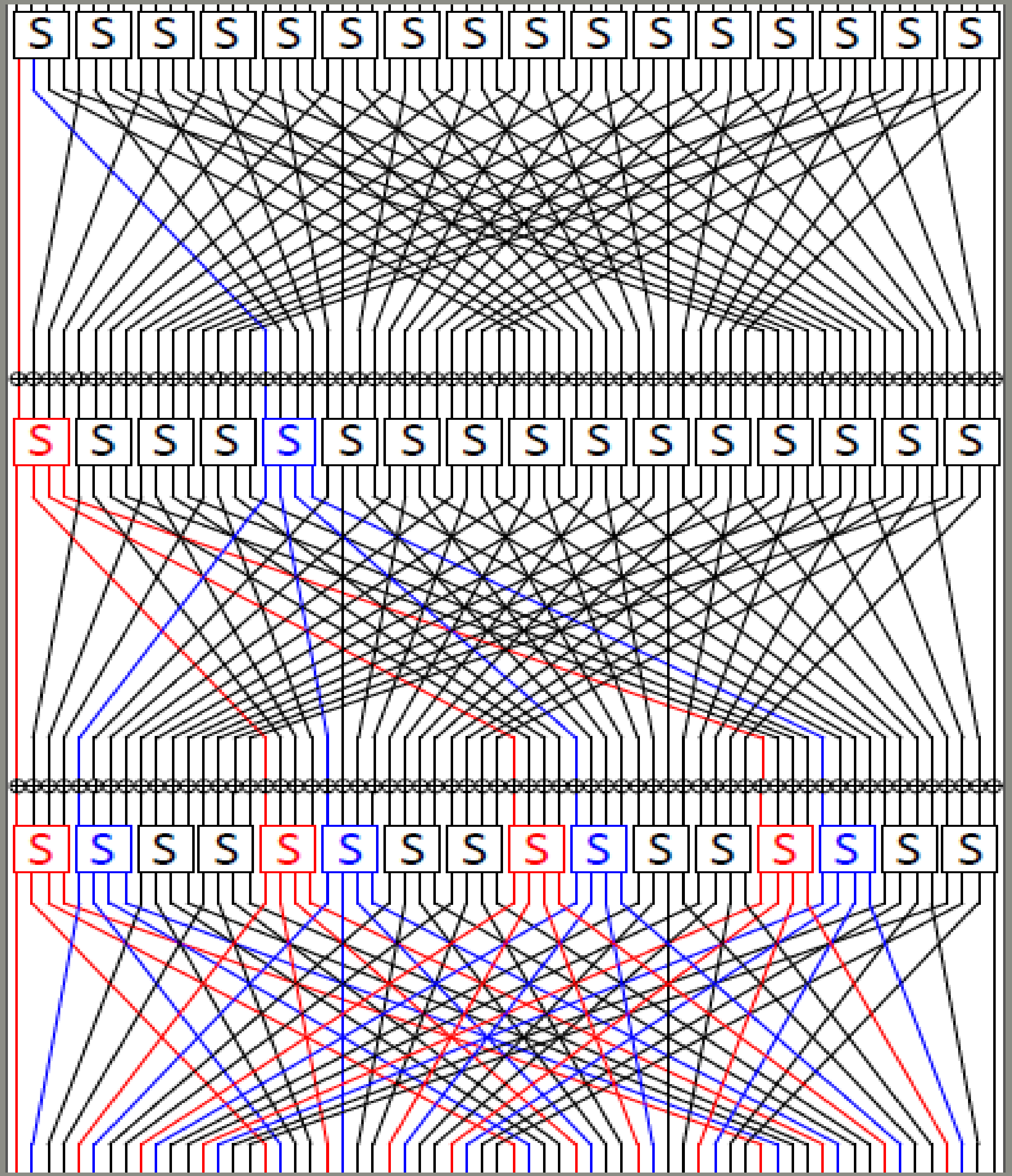
Fault Propagation



Example Scenario-1

- Suppose the output fault mask for nibble 0 in round 28 is **0001** (Hamming weight = 1).
  - *This is also the input fault mask in round 29 for nibble 0*
- Each of the nibbles 0, 4, 8 and 12 in round 30 have one of the following input fault masks:
  - **0000** *(implying no fault propagation)*
  - **0001** *(implying fault propagation)*
- If the input fault mask for nibble 0 in round 30 is 0001, then the input fault mask for nibbles 0, 4, 8 and 12 in round 31 are either **0000** or **0001**.
- If the input fault mask for nibble 4 in round 30 is 0001, then the input fault mask for nibbles 1, 5, 9 and 13 in round 31 are either **0000** or **0001**.
- If the input fault mask for nibble 8 in round 30 is 0001, then the input fault mask for nibbles 2, 6, 10 and 14 in round 31 are either **0000** or **0001**.
- If the input fault mask for nibble 12 in round 30 is 0001, then the input fault mask for nibbles 3, 7, 11 and 15 in round 31 are either **0000** or **0001**.

Fault Propagation



Example Scenario-2

- Suppose the output fault mask for nibble 0 in round 28 is 0011.
  - *The input fault masks in round 29 for nibble 0 and nibble 4 are* 0001 *and* 0001*, respectively*
- From the previous example, each of the nibbles 0, 4, 8 and 12 in round 30 have one of the following input fault masks:
  - 0000 *(implying no fault propagation)*
  - 0001 *(implying fault propagation)*
- Additionally, each of the nibbles 1, 5, 9 and 13 in round 30 have one of the following input fault masks:
  - 0000 *(implying no fault propagation)*
  - 0001 *(implying fault propagation)*
- It now follows that there are four possible input masks for each of the nibbles in round 31:
  - 0000 *(implying no fault propagation)*
  - 0001 *(implying fault propagation as in Example-1)*
  - 0010 *(implying only additional fault propagation)*
  - 0011 *(implying combined fault propagation)*

The Generalized Proof of Theorems-1 and 2

Round 28

- Suppose the adversary injects a fault in nibble $4n + d$, where $n, d \in \{0,1,2,3\}$
- Suppose the output fault mask has Hamming weight $x \in \{0,1,2,3,4\}$
- Let $l_1, \cdots, l_x \in \{0,1,2,3\}$ be the bits in the output fault mask that are set to 1

Round 29

- These faulty bits propagate to the nibbles $n + 4l_1, \cdots, n + 4l_x$ respectively, in round 29
(recall the generic diffusion property introduced in Slide 8)
- Each faulty bit creates an input fault mask of Hamming weight 1 in round 29

The Generalized Proof (contd.)

Round 30

- Consider the faulty nibble $n + 4l_1$ in round 29, as discussed in the previous slide
- The output of this faulty nibble will propagate to the n^{th} input bit of the quartet of nibbles $(l_1, l_1 + 4, l_1 + 8, l_1 + 12)$ in round 30
- This again follows from the generic diffusion properties discussed in Slide 8
- The case for the remaining faulty nibbles in round 29 follows analogously

Round 31

- Consider the faulty quartet of nibbles $(l_1, l_1 + 4, l_1 + 8, l_1 + 12)$ in round 30
- Each nibble in round 31 will potentially have its l_1^{th} bit affected by one of these quartet of nibbles
- The cases for $l_2, \cdots, l_x$ follow analogously
- Thus, each nibble in round 31 has an input fault mask of Hamming weight at most x. This completes the proof of Theorem-1.
- Since exactly x bits of each input fault mask are potentially 1, each input fault mask can take at most 2^x values. This completes the proof of Theorem-2.

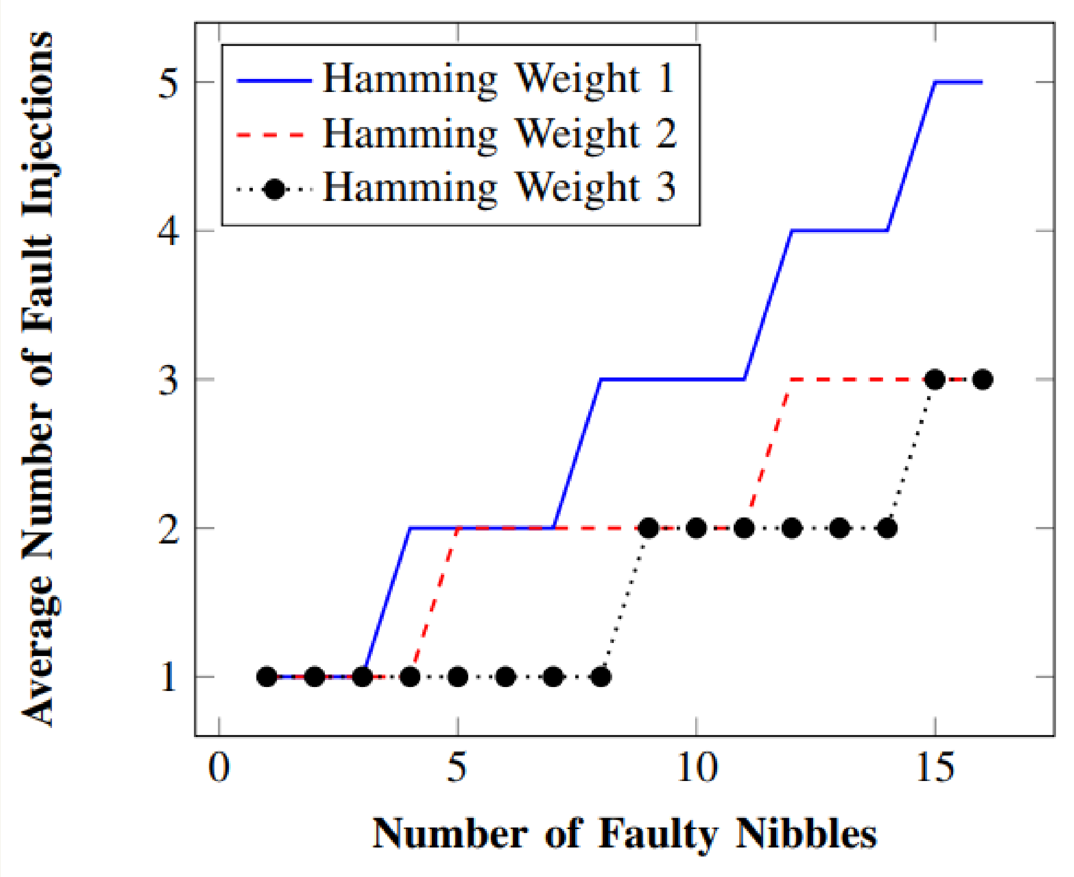
Key Recovery

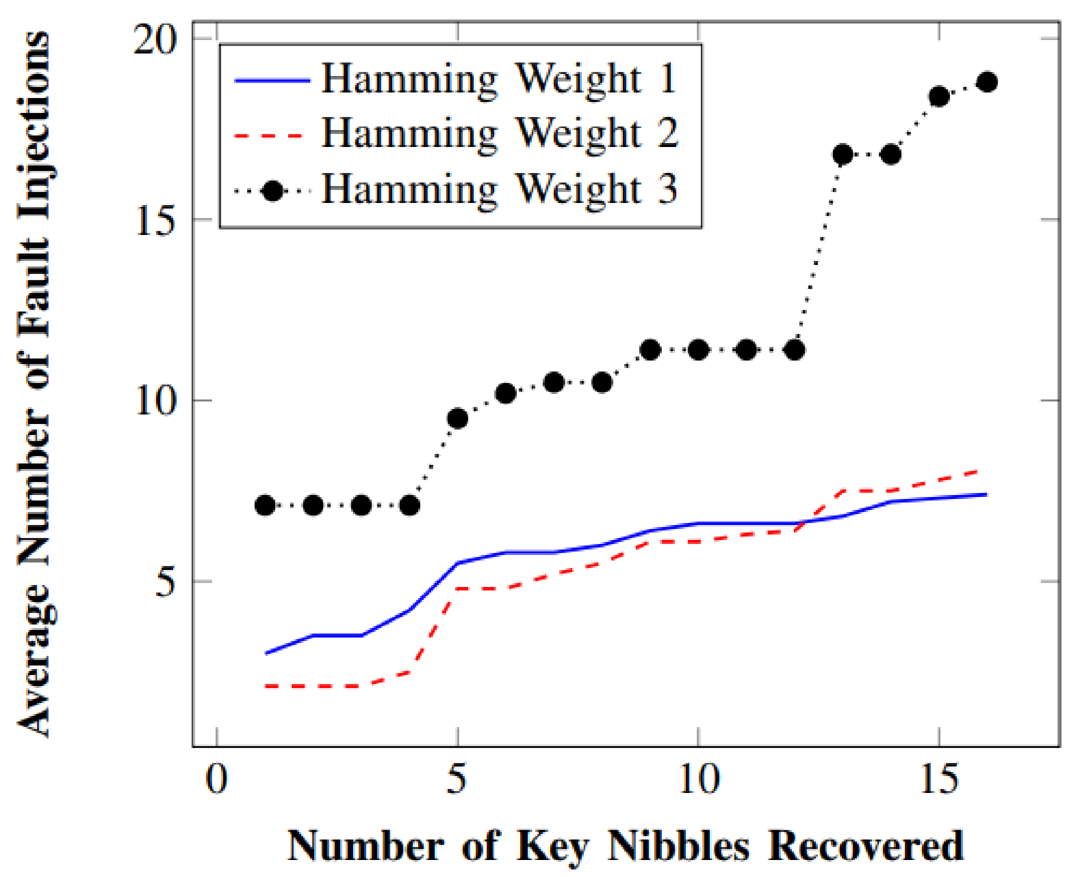
- Suppose we wish to recover a given nibble of the last round key of PRESENT. Let the key nibble be denoted as K
- Let the corresponding correct and faulty ciphertext nibbles be denoted as C and C', respectively.
- Finally, let β denote the input differential for the corresponding nibble in round 31. As already mentioned, for a output fault mask of Hamming weight x, there are precisely 2^x-1 non-zero values that β can take
- We solve the equation: $S^{-1}(C \oplus K) \oplus S^{-1}(C' \oplus K) = \beta$ for all possible values of β, and obtain the corresponding key hypothesis values for $\text{Type equation here.}$
  - *For a given set of (C, C', β) values, the equation yields one solution on an average for the PRESENT S-Box*
- Note that the above equation reduces the guessing entropy of K only if $x < 4$, that is, β does not take the full range of (2^4-1) values
- Hence, faults that result in a output mask of Hamming weight less than or equal to 3 in round 28 are useful for the attack

Key Recovery (contd.)

- The key-recovery process is efficient:
  - Multiple key nibbles may be recovered simultaneously using the same set of fault injections
- The Hamming weight x of the output fault mask in round 28 leads to an efficiency trade-off:
  - Greater the value of x, greater is the expected number of faulty nibbles per fault injection in round 31, and hence, more is the number of key nibbles that can be recovered simultaneously
  - Smaller the value of x, smaller is the number of values that the input differential β can take, and hence, lower is the number of key hypothesis per fault injection.

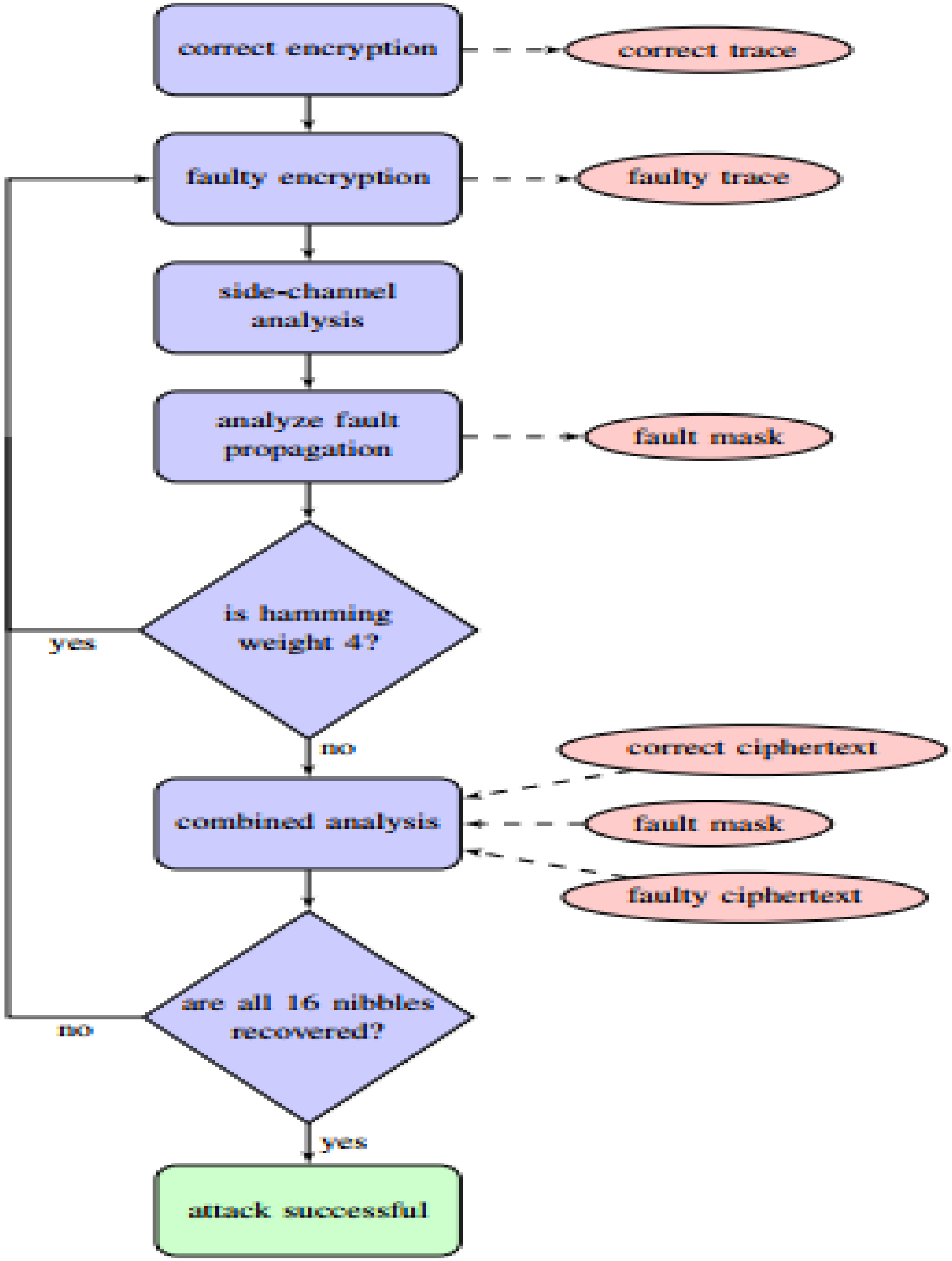
Key Recovery: Simulation Study





On an average, an output fault mask of greater Hamming Weight results in a greater number faulty nibbles and a greater number of recovered key nibbles per fault injection instance

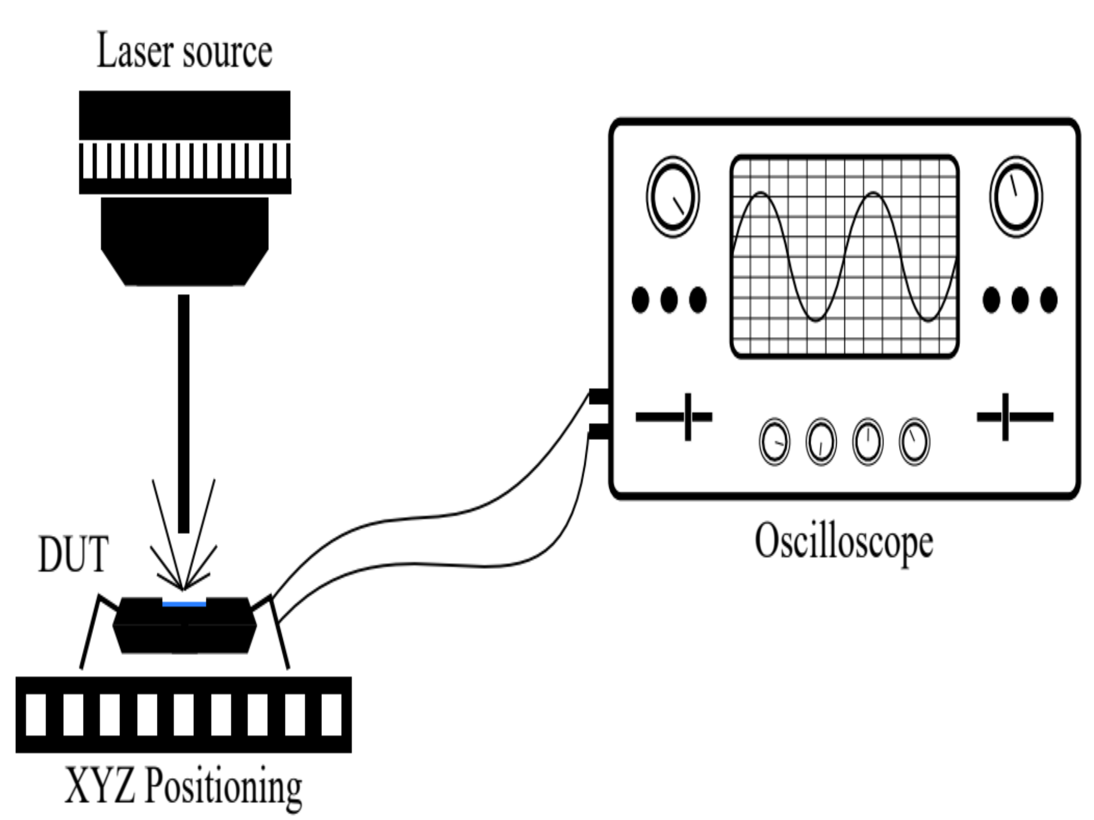
Attack Summary



EXPERIMENTAL VALIDATION OF THE PROPOSED ATTACK METHODOLOGY

Target Platform: ATmega328P Microcontroller

The Experimental Setup



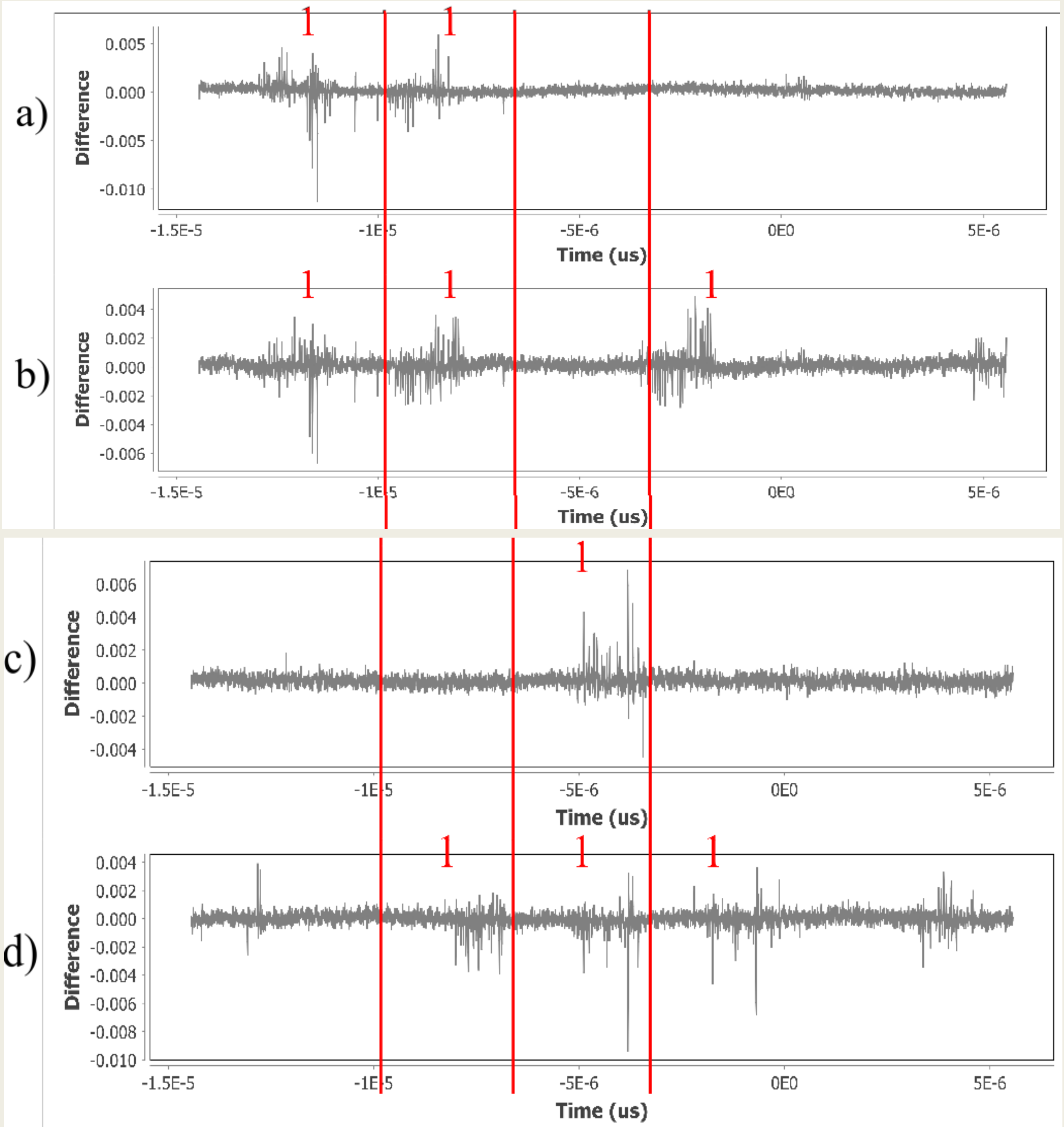
- Device Under Target (DUT):
  - Atmega328P microcontroller
  - Decapsulated from the back-side
  - Mounted on an Aurdnio UNO development board
  - XYZ positioning table with a spatial precision of 0.05 µm
- Fault Injection: Skipping a target S-Box operation using a laser pulse:
  - A near-infrared diode pulse laser (1064 nm wavelength) with the maximum output power of 20 W.
  - 20x objective lens to scale the effective spot size to 15x3.5 µm
  - Laser Activation Length: 150 ns
  - Laser Power: 0.24 W
- Side-Channel Measurement:
  - Digital Oscilloscope
  - Capture the time frame from one round after fault injection

Power Trace Measurement

- The information as to which nibble has been faulted is computed from the timing information with respect to the trigger
- Once the faulty nibble is identified, the differential of the correct and faulty trace reveals the output fault mask

An Example for Illustration

Trace	Offset (ns)	I/P Fault Mask:R29	O/P Fault Mask:R28
a)	4032	0000000000800080	0000000000C0000
b)	4914	0040000000400040	000000000D00000
c)	7686	0000080000000000	0000000200000000
d)	9072	0200020002000000	0000070000000000

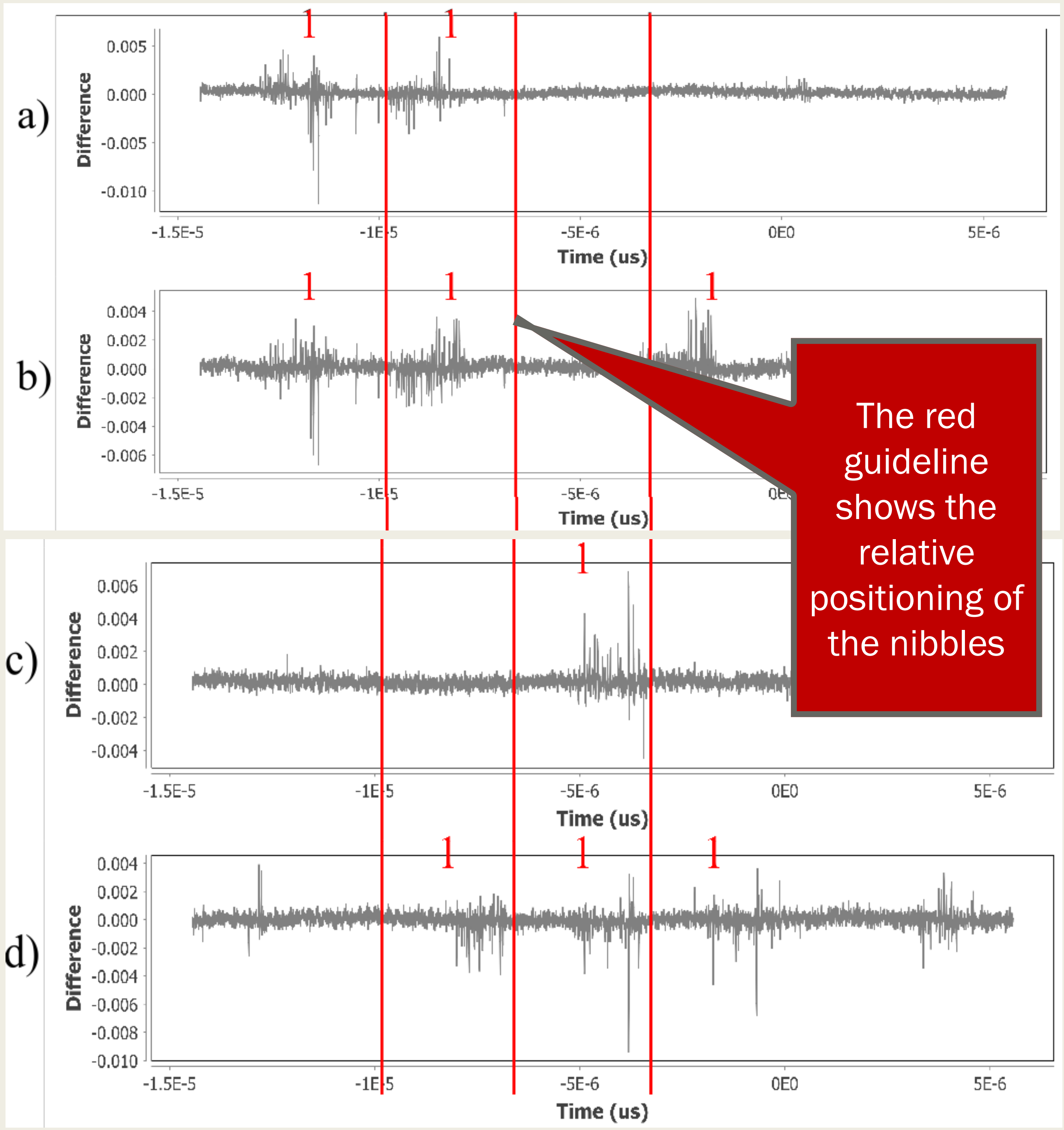


Power Trace Measurement

- The information as to which nibble has been faulted is computed from the timing information with respect to the trigger
- Once the faulty nibble is identified, the differential of the correct and faulty trace reveals the output fault mask

An Example for Illustration

Trace	Offset (ns)	I/P Fault Mask:R29	O/P Fault Mask:R28
a)	4032	0000000000800080	0000000000C0000
b)	4914	0040000000400040	000000000D00000
c)	7686	0000080000000000	0000000200000000
d)	9072	0200020002000000	0000070000000000

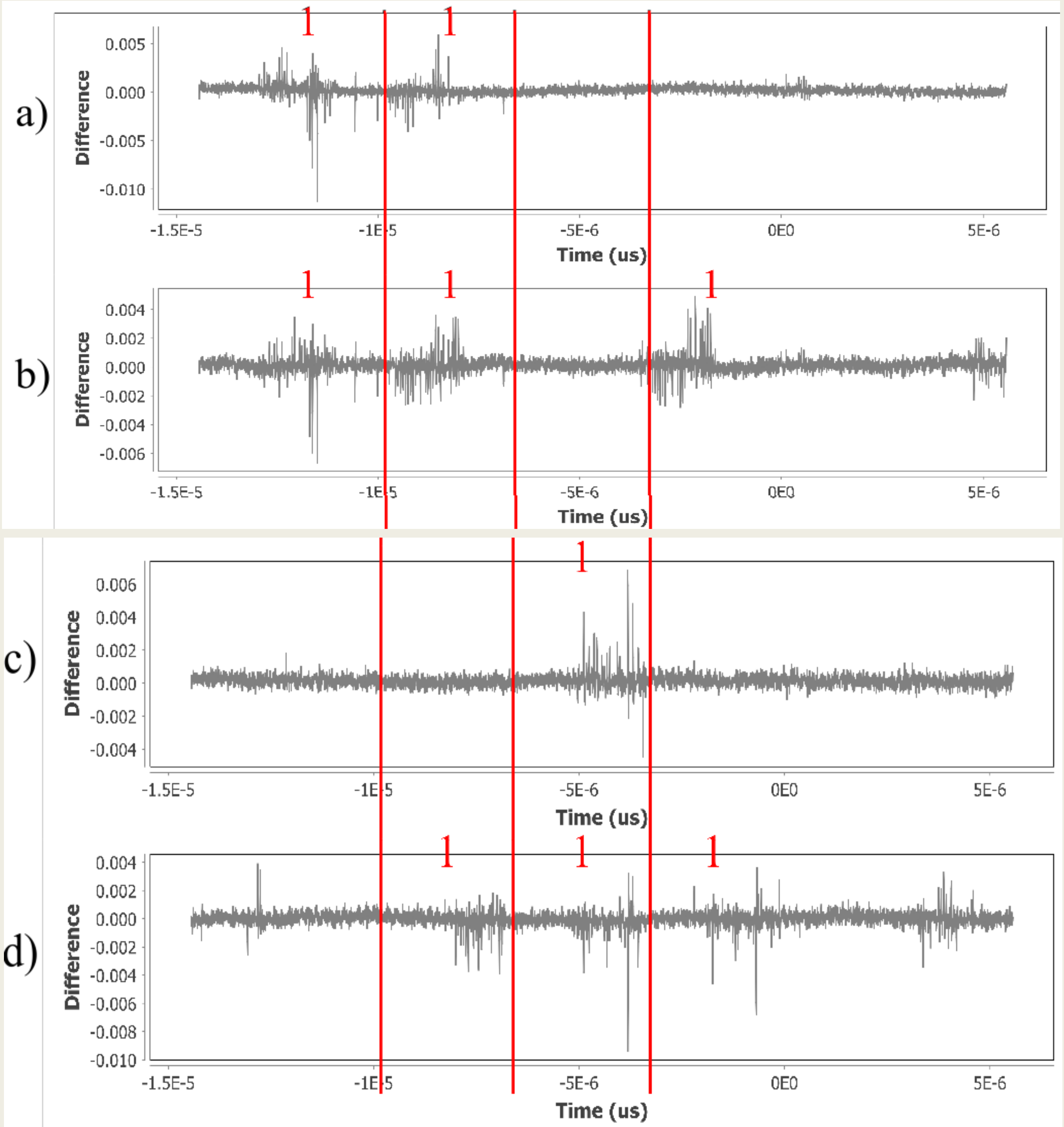


Power Trace Measurement

- The information as to which nibble has been faulted is computed from the timing information with respect to the trigger
- Once the faulty nibble is identified, the differential of the correct and faulty trace reveals the output fault mask

An Example for Illustration

Trace	Offset (ns)	I/P Fault Mask:R29	O/P Fault Mask:R28
a)	4032	0000000000800080	0000000000C0000
b)	4914	0040000000400040	000000000D00000
c)	7686	0000080000000000	0000000200000000
d)	9072	0200020002000000	0000070000000000

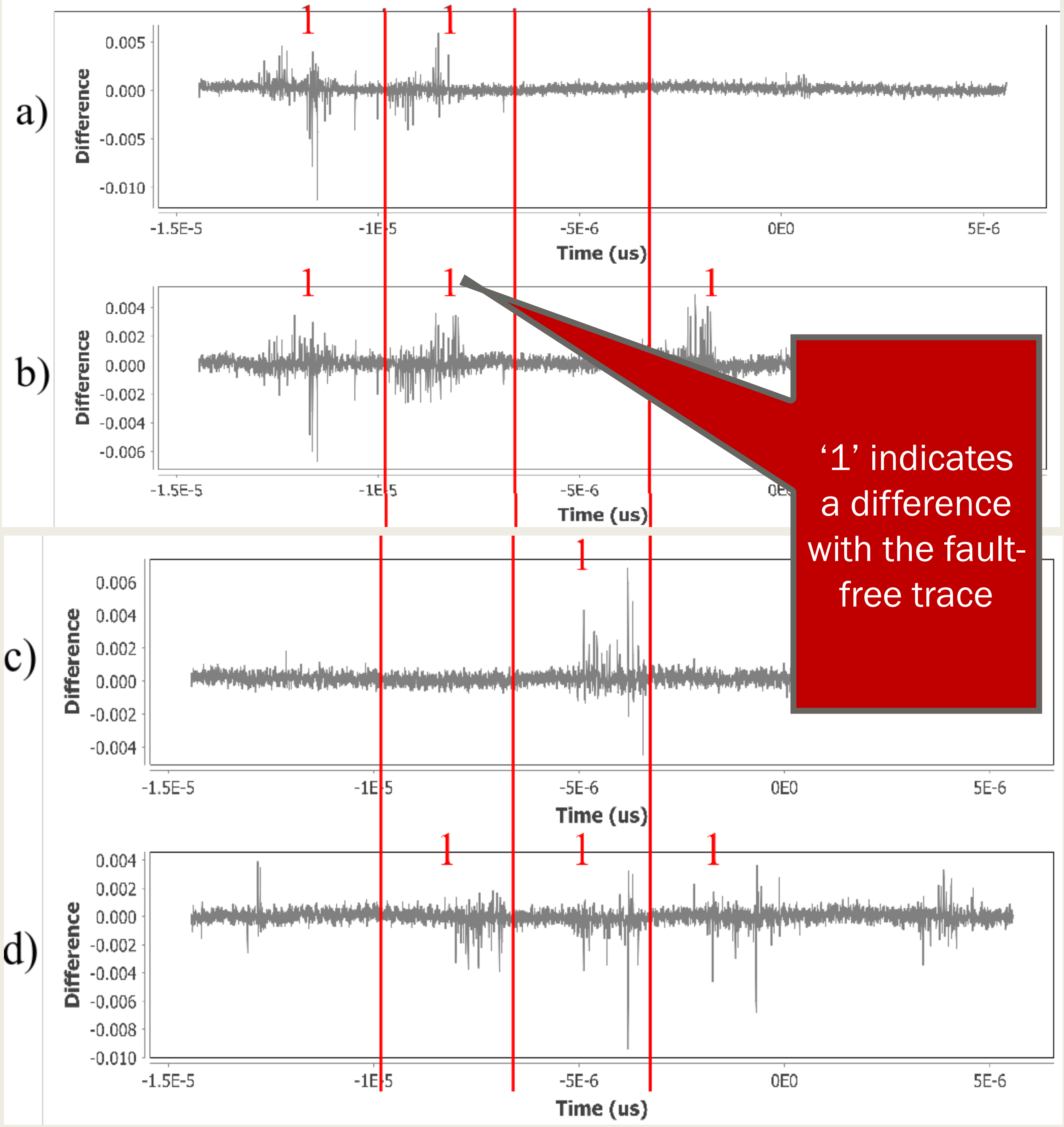


Power Trace Measurement

- The information as to which nibble has been faulted is computed from the timing information with respect to the trigger
- Once the faulty nibble is identified, the differential of the correct and faulty trace reveals the output fault mask

An Example for Illustration

Trace	Offset (ns)	I/P Fault Mask:R29	O/P Fault Mask:R28
a)	4032	0000000000800080	0000000000C0000
b)	4914	0040000000400040	000000000D00000
c)	7686	0000080000000000	0000000200000000
d)	9072	0200020002000000	0000070000000000



Attack Performance and Efficiency

Differential Fault Analysis (Bit/Nibble Faults)	Bagheri et al. (2013)	Fault Injection Instances: 18 Key Recovery Complexity: 2^{16}
Differential Fault Analysis (Hardware Trojan-Horse)	Breier and He (2015)	Multiple Fault Attack Fault Injection Instances: 2 Each Fault Instance targets 4 nibbles Key Recovery Complexity: 2^{16}
Differential Fault Intensity Analysis (Bit/Nibble Faults)	Ghalaty et al. (2015)	Requires only Faulty Ciphertexts Fault Injection Instances: 10-36 Key Recovery Complexity: 2^{16}
DPA+DFA **(Random Nibble Faults)**	**Our Work (2017)**	**Fault Injection Instances (Best Case): 3** **Fault Injection Instances (Worst Case): 19** **Fault Injection Instances (Avg. Case): 7-8** **Key Recovery Complexity: 2^{16}**

ATTACK EXTENSIONS AND COUNTERMEASURES

Extensions to Our Attack

- Extensions to other rounds of PRESENT
  - While it is relatively easy to determine the faulty nibbles in round 29, this process becomes harder once the propagation of the fault produces collisions
  - Requires creation of SCA templates for each nibble and each fault mask, resulting in total of 256 different templates
- Extensions to other block ciphers
  - Our attack can be extended to GIFT – a cryptanalytically stronger version of PRESENT (to be presented at CHES 2017)
  - **Conjecture: Our attack is applicable to any block cipher that uses bit-permutations with optimal diffusion characteristics**
  - The attack is not applicable to block ciphers using MDS matrices

Possible Countermeasures

- Standard fault detection mechanisms such as spatial and temporal redundancy don't work:
  - They can be easily bypassed using **biased fault injections**
  - Only serve to increase the number of fault instances required
  - Do not eliminate chances of the attack
- Side-Channel countermeasures such as Masking:
  - Make the attack potentially harder
  - Might require higher order analysis over the collected traces

Thank You