COSIC

KU LEUVEN

Threshold Implementations

Benedikt Gierlichs

Reference:

A more efficient Threshold Implementation of AES

Begül Bilgin, Benedikt Gierlichs, Svetla Nikova, Ventzislav Nikov, Vincent Rijmen

Africacrypt 2014, available http://eprint.iacr.org/2013/697

Introduction - masking

- Masking countermeasure against side channel attacks
  - Process random shares instead of direct values
  - Often boolean masking:
    1st order masking: v → (mask, v⊕mask) with random mask
  - Masking linear function: f(v) = f(mask⊕v⊕mask) = f(mask)⊕f(v⊕mask)
    - Processing f() on either share cannot leak any information
    - Processing f() on both shares in parallel is 1st order DPA secure
  - Masking non-linear function (S-box): g(v) ≠ g(mask)⊕g(v⊕mask)
    - Need a 2nd function: g(v) = g(mask) ⊕ h(mask,v⊕mask)
    - Processing g() on one share cannot leak any information
    - Processing h() on both shares provable 1st order DPA secure?

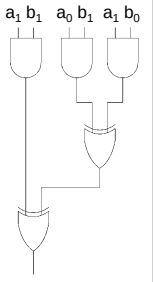
Introduction - glitches

- h(mask,v⊕mask)
  - Processing h() on both shares may not be 1st order secure!
  - Function h() knows both shares
  - Depends on implementation of function
- Glitches are temporary intermediate states of combinational logic
- Glitches can be a serious security problem

Introduction - glitches

- $ab = (a_0 \oplus a_1)(b_0 \oplus b_1) = a_0b_0 \oplus a_0b_1 \oplus a_1b_0 \oplus a_1b_1$
- Share1 = a_0b_0 Share2 = $(a_0b_1 \oplus a_1b_0) \oplus a_1b_1$
- Suppose a_1 arrives late, a_0 not relevant

a1	b0	b1	AND	XOR	#
0→1	0	0	0	0	0
1→0	0	0	0	0	0
0→1	1	1	2	2	4
1→0	1	1	2	2	4
0→1	1	0	1	2	3
1→0	1	0	1	2	3
0→1	0	1	1	1	2
1→0	0	1	1	1	2



Introduction - glitches

- ab = (a ⊕a)(b ⊕b) = a b ⊕ a b ⊕ a b ⊕ a b
- Sha

Not provable 1st order DPA secure!

- Sup

a1	b0	b1	AND	XOR	#
0→1	0	0	0	0	0
1→0	0	0	0	0	0
0→1	1	1	2	2	4
1→0	1	1	2	2	4
0→1	1	0	1	2	3
1→0	1	0	1	2	3
0→1	0	1	1	1	2
1→0	0	1	1	1	2

b = 0
average = 2

b = 1
average = 2.5

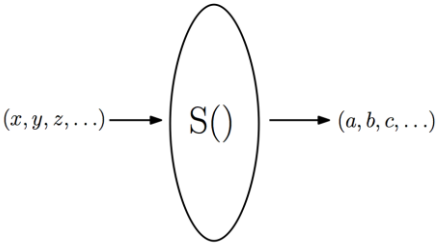
Solutions

- Engineering approach:
  - Try to fix implementation
  - Not provable secure but hopefully secure in practice
  - Requires design iterations and testing
- Mathematical approach:
  - Change the masking to entirely avoid the problem

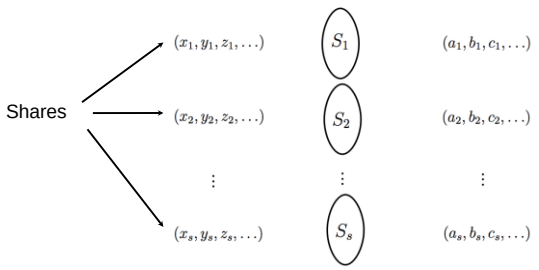
Threshold Implementations

- Based on Boolean masking and multiparty computation
- Pros
  - ✓ Security in a circuit with glitches, any HW technology
  - ✓ Provable secure against 1st order DPA
- Cons
  - Only 1st order DPA resistant
  - High-degree non-linear functions difficult to implement

Threshold Implementations

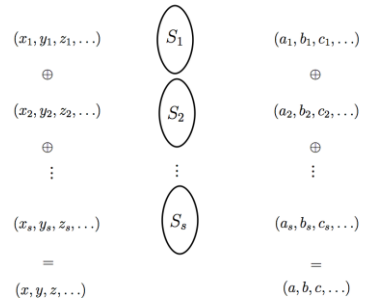


Threshold Implementations



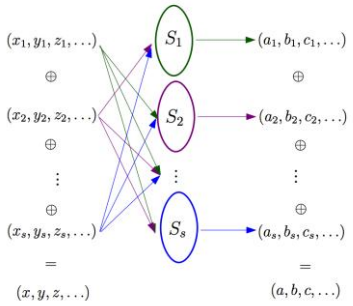
3 properties

Threshold Implementations



Correctness

Threshold Implementations



Correctness, non-completeness

Non-completeness

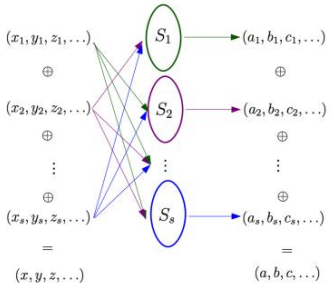
- Example

$$S(x, y, z) = x + yz$$

$$\begin{aligned} S_1 &= x_2 + y_2 z_2 + y_2 z_3 + y_3 z_2 \\ S_2 &= x_3 + y_3 z_3 + y_3 z_1 + y_1 z_3 \\ S_3 &= x_1 + y_1 z_1 + y_1 z_2 + y_2 z_1 \end{aligned}$$

- To protect a function with degree d, at least d+1 shares are required

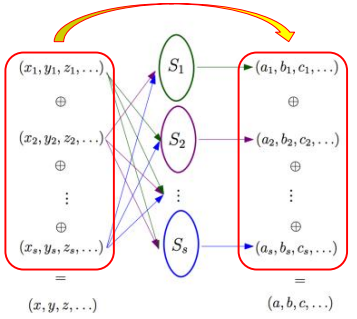
Threshold Implementations



Correctness, non-completeness, uniformity

Uniformity

uniformly distributed

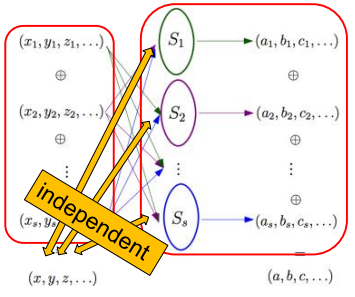


uniformly distributed

Security

non-complete

uniformly distributed

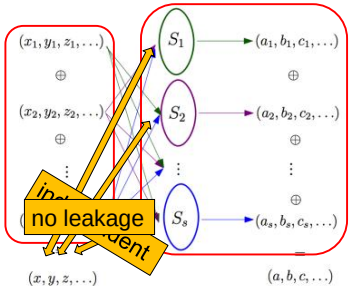


total power is linear function of components, e.g. sum

Security

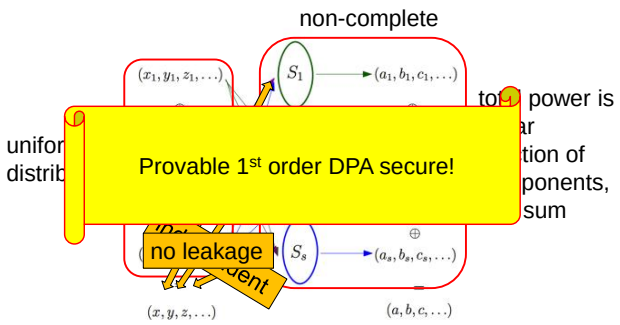
non-complete

uniformly distributed

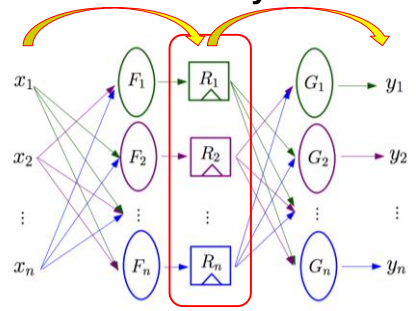


total power is linear function of components, e.g. sum

Security



Security

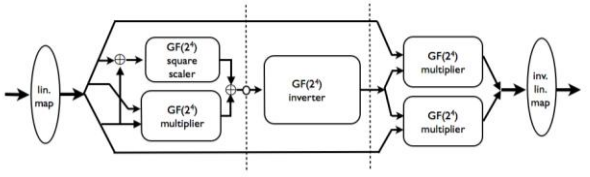


- Separate two non-linear functions by register, stop glitches
- Uniformity ensures that input of next function is uniformly distributed

IMPLEMENTATION

AES S-box

- Only nonlinear part of AES
- Based on multiplicative inverse in GF(256) and affine transformation
- Possible to implement with tower field approach (GF(2^4))

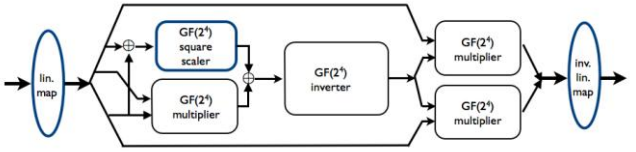


Our AES implementation

- Goal: compact and secure implementation
- Serial data path, 1 S-box

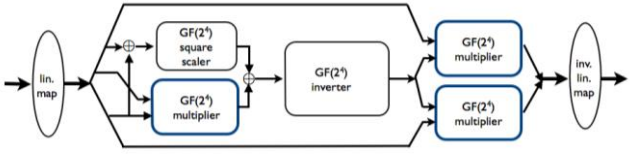
- Main idea: adjust number of shares as needed to minimize gate count
- Linear part: only 2 shares
- S-box: 2 to 5 shares

Our implementation of AES S-box



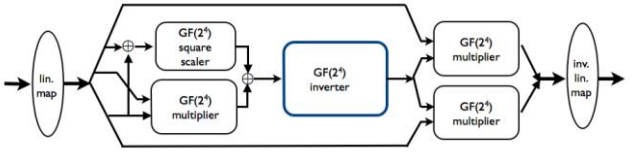
2 shares

Our implementation of AES S-box



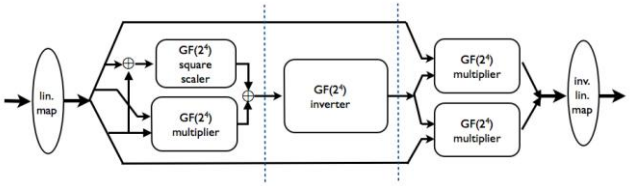
4 input, 3 output shares

Our implementation of AES S-box



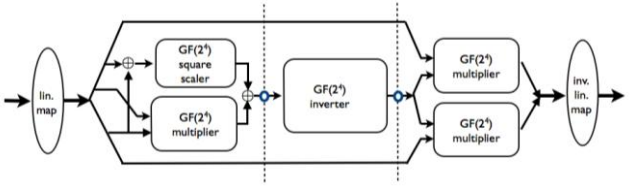
5 shares

Our implementation of AES S-box



registers after every nonlinear function

Our implementation of AES S-box



re-masking to change the number of shares and preserve uniformity

Implementation results

	State Array	Key Array	S-box	MixCol Col	Contr.[1]	Key XOR	MUX	Other	Total	cycles	rand bits[2]
[18]	2529	2526	4244	1120	166	64	376	89	11114/11031[3]	266	48
This paper	1698	1890	3708	770	221	48	746	21	9102	246	44
This paper[3]	1698	1890	3003	544	221	48	746	21	8171	246	44

[1] including round constant [2] per S-box [3] *compile_ultra*

- 18% smaller, 7.5% faster
- 8% less randomness for re-masking

- Our TI of S-box uses 3.7k GE (3k GE)
  - Based on plain Canright S-box 233 GE (Moradi et al.)
- Our TI of AES uses 9k GE (8k GE)
  - Based on plain AES 2.4k GE (Moradi et al.)

Implementation results

	State Array	Key Array	S-box	MixCol Col	Contr.[1]	Key XOR	MUX	Other	Total	cycles	rand bits[2]
[18]	2529	2526	4244	1120	166	64	376	89	11114/11031[3]	266	48
This paper	1698	1890	3708	770	221	48	746	21	9102	246	44
This paper[3]	1698									246	44

[1] including round c

But we still require 44 * 20 * 10 = 8800 random bits per AES operation. Not practical.

☹

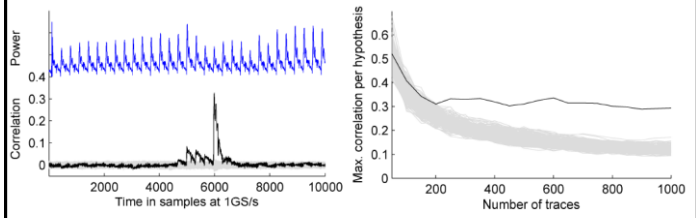
- 18% s
- 8% le
- Our T
  - Based on plain Canright S-box 233 GE (Moradi et al.)
- Our TI of AES uses 9k GE (8k GE)
  - Based on plain AES 2.4k GE (Moradi et al.)

Practical security evaluation

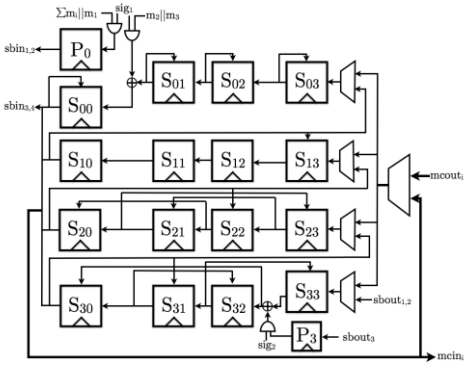
- FPGA implementation on SASEBO-G
  - Crypto FPGA: only TI-AES and PRNG
  - Use *keep hierarchy* constraint for TI-AES module
  - Measurements: 1ohm resistor, passive probe, 1GS/s
  - Measurements cover first 1.5 rounds
- Low noise measurements
  - Real attacks will be more difficult
- Goals
  - Verify 1st order attack security
  - Check higher order attack security

Practical security evaluation

- PRNG off, 1st order CPA, HD model at S-box output
- Highest peak 3 cycles later, input to MC

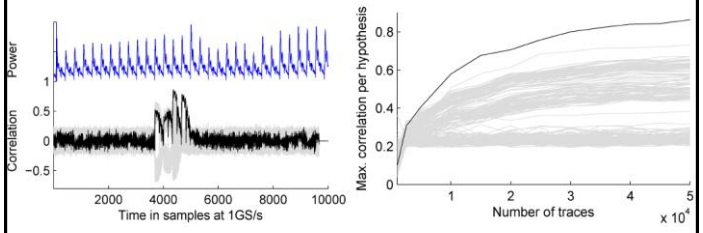


AES implementation



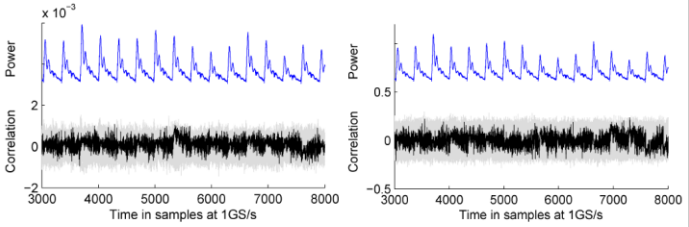
Practical security evaluation

- PRNG off, 1st order correlation collision attack



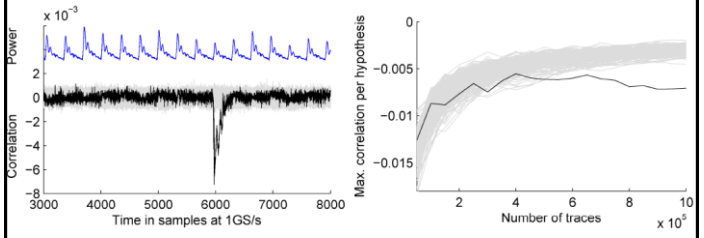
Practical security evaluation

- PRNG on, 1st order CPA / correlation collision attack
- 10 million traces



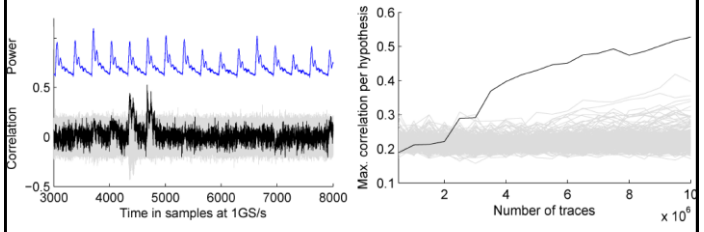
Practical security evaluation

- PRNG on, 2nd order CPA, HD model at S-box output



Practical security evaluation

- PRNG on, 2nd order correlation collision attack



Practical security evaluation

- PRNG switched on
- 1st order attack resistant using 10 million traces
- Best 2nd order attack requires 600k traces

- Noise makes 2nd order attacks more difficult
- Number of traces increases quadratically in noise standard deviation
- We had very little noise in the measurements

More recent results

- 1st order TI of AES with >2 shares: requires several million traces to break with 2nd order attack
  [not yet published]

- Extension of Threshold Implementation to higher orders
  - Theory & Proofs
  - Example: 2nd order TI of KATAN block cipher
  - No leakage in 1st and 2nd moment with 300 MILLION traces

  [Asiacrypt 2014] available http://eprint.iacr.org/2014/751

谢 谢

www.cosic.be
benedikt.gierlichs@esat.kuleuven.be