NPTEL ONLINE CERTIFICATION COURSES

Course Name: Hardware Security

Faculty Name: Prof Debdeep Mukhopadhyay

Department : Computer Science and Engineering

Topic

Lecture 44: Power Analysis Countermeasures

CONCEPTS COVERED

Concepts Covered:

- Masking
- TI
- Properties of TI
- Some Constructions
- Experimental Evaluations and Results

Nonlinear Masking

- It is challenging for nonlinear functions.
- Example: $f(X, Y) = Z \oplus XY$
- Masked Circuit:
  - $f_1(X_1, Y_1) = Z_1 \oplus X_1 Y_1$
  - $f_2(X_1, X_2, Y_1, Y_2) = \big((Z_2 \oplus X_1 Y_2) \oplus X_2 Y_1\big) \oplus X_2 Y_2$
- Note again the ordering of the operations is very important!
  - Don't do, $f_2(X_1, X_2, Y_1, Y_2) = (Z_2 \oplus X_1 Y_2) \oplus (X_2 Y_1 \oplus X_2 Y_2)$ …as the second parenthesis is dependent on Y
- However, this is not secure against higher order attacks.
- Actually, not even 1st order attacks if there are glitches.

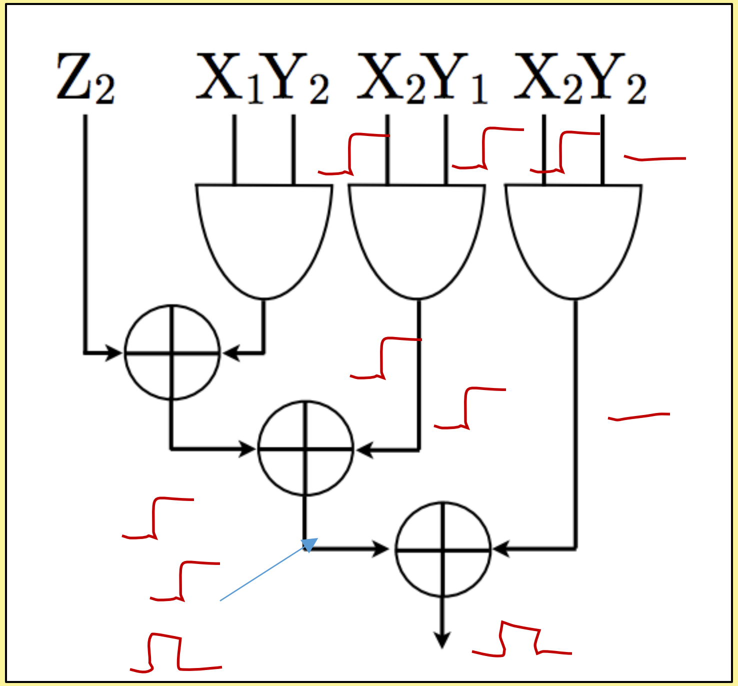
Higher Order DPA

- Like in a 1st order DPA, where we process on a single point on the power trace, in 2nd order attacks, we exploit the joint leakage of two intermediate values that are processed by the device.
- The attack works in the same way as the 1st order attack, except that we preprocess the trace first.
- For example, if we know the points in the trace when f_1 and f_2 are processed, then combining them would reveal information of unmasked data, and then DPA would still work.
- A common preprocessing operation is to take two power values at different times, say p_{t_1}, p_{t_2}, and determine $\left(p_{t_1} - p_{t_2}\right)^2$

Probing Model

- Here an attacker can observe the values of up to d intermediate wires of the circuit per bit during the computation within a certain time.
- The correspondence between the d-probing model and the d^{th} order DPA is summarized:
  - The attack order in a higher order DPA corresponds to the number of wires that are probed in the circuit (per unmasked bit)
- This result implies that if a circuit is secure against d probes, then combining d power consumption points will reveal no information.
- If the operations that correspond to the probed wires are in parallel this is equivalent to security against DPA exploiting d^{th} order statistical moment.
- Note that this is a stronger model than higher order DPA: if a system is secure against d-probing attacks, it is also secure against d^{th} order DPA.

Security on a Glitchy Circuit



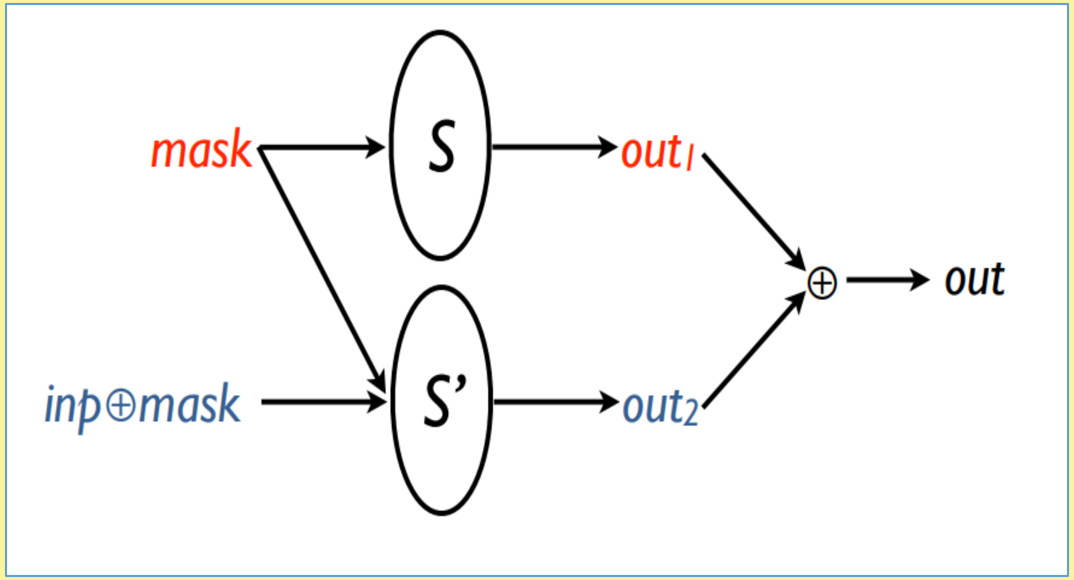
y	y_1	y_2	x_2	$z_2 \oplus x_1y_2$	AND	XOR
0	0	0	0	0	0+0	0+0
0	1	1	0	0	0+0	0+0
0	0	0	1	0	0+0	0+0
0	1	1	1	0	0+2	0+1
0	0	0	0	1	0+0	2+0
0	1	1	0	1	0+0	2+0
0	0	0	1	1	0+0	2+0
0	1	1	1	1	0+2	2+1
1	0	1	0	0	0+0	0+0
1	1	0	0	0	0+0	0+0
1	0	1	1	0	0+1	0+1
1	1	0	1	0	0+1	0+2
1	0	1	0	1	0+0	2+0
1	1	0	0	1	0+0	2+0
1	0	1	1	1	0+1	2+1
1	1	0	1	1	0+1	2+2

Average glitch power for the AND gate does not depend on y.

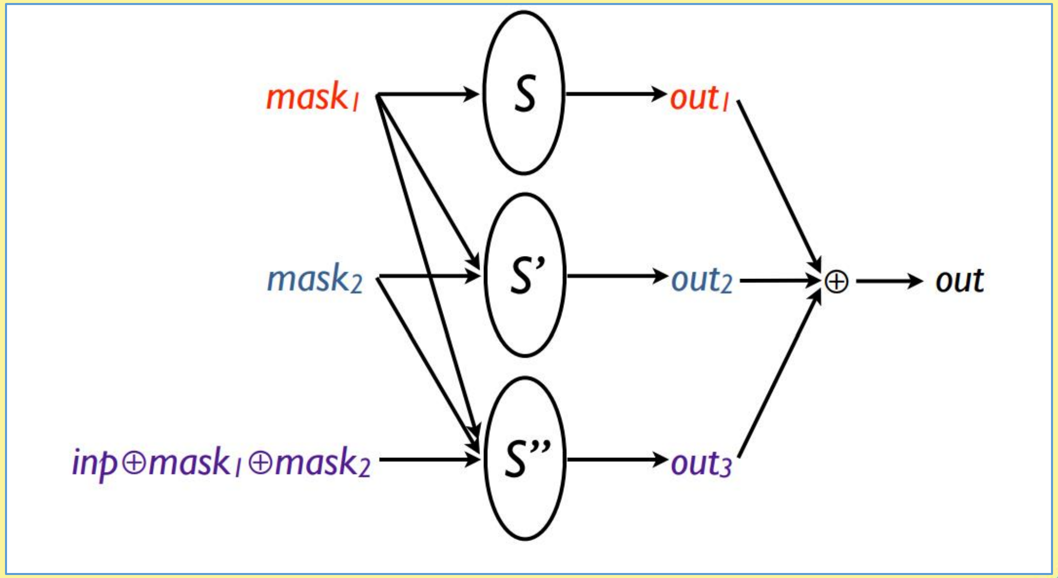
Average glitch power for the XOR gate depends on y.

The probing model captures the effect of glitches by letting the attacker gather information on inputs, like Y_1, Y_2, and intermediate values, $X_2Y_1, X_2Y_2, etc.$

Masking Overview

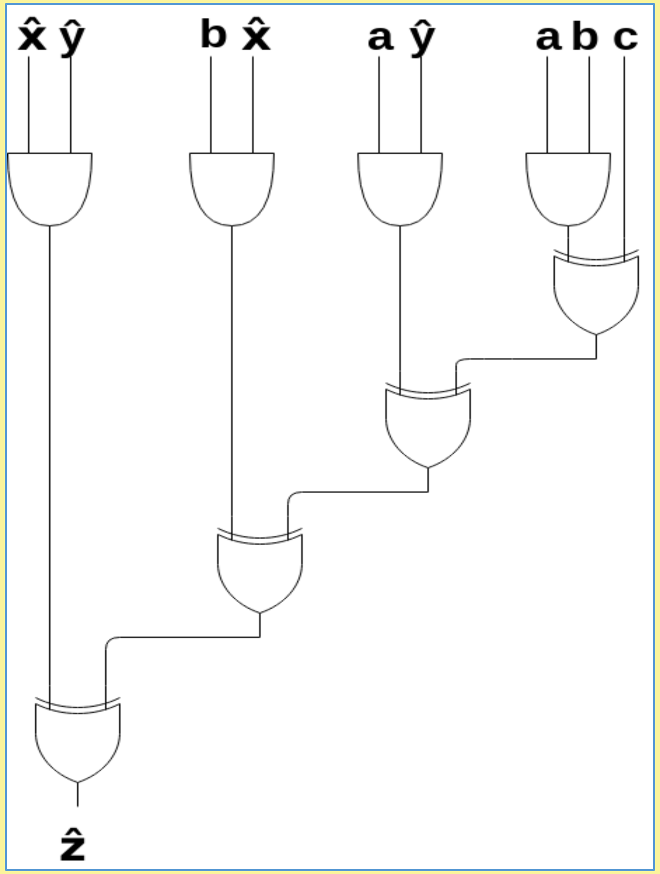


First Order Masking



Second Order Masking

Traditional Masking: A Closer Look



$$\hat{z} = \hat{x}\hat{y} \oplus (b\hat{x} \oplus (a\hat{y} \oplus (ab \oplus c)))$$

The Trichina AND Gate

Vulnerability to Glitches

- Consider a glitch in input $\hat{x}$
- Table shows the number of gates affected by the glitch depending on the value of **y**
- The power consumption caused by the glitch is related to the number of gates affected
- The power consumptions differ with different values of **y** leading to the leakage of value of **y**

b	$\hat{y}$	y	AND	XOR
0	0	0	0	0
0	1	1	1	1
1	0	1	1	2
1	1	0	2	2

$$\hat{y} = y \oplus b$$

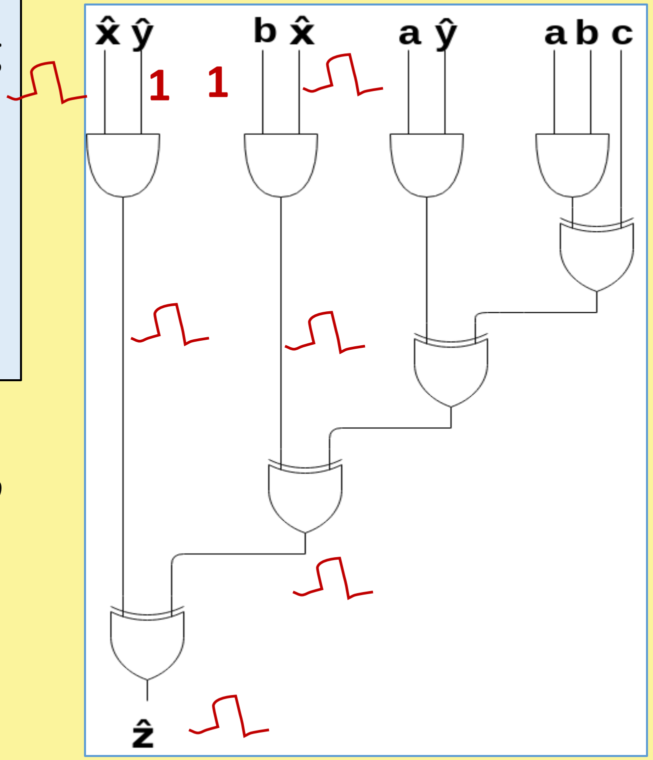


Table 1

Threshold Implementation (TI)

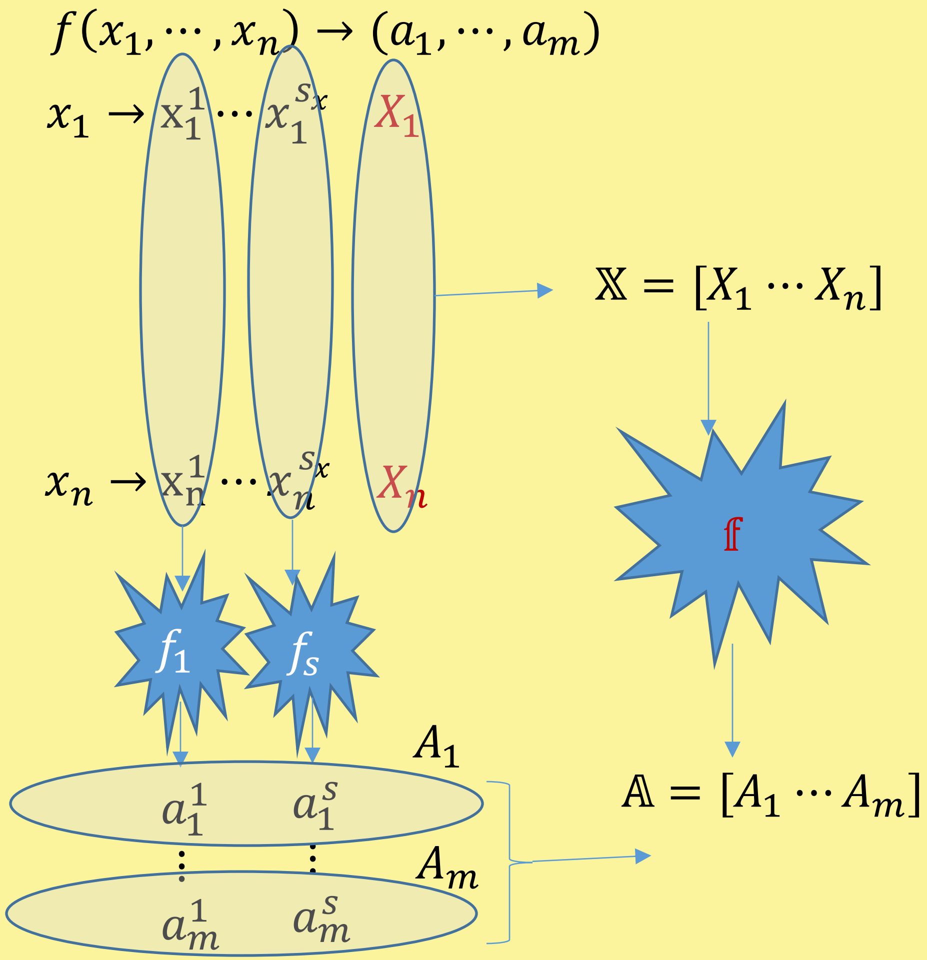
- TI is based on multi-party computations and secret sharing.
- TI of any function is used as a d^{th}-order DPA countermeasure on a device that reveals a linear combination of the intermediate values' noisy leakage.
- We define four properties:
  - Correctness
  - Uniform masking
  - Non-completeness
  - Uniform sharing of a Function

Threshold Implementations

Consider a Boolean function $f(x) = a$ from $F_2^n \rightarrow F_2^m$

We split each variable x into s_x shares $x_1, \cdots, x_{s_x}$.

This is called s_x-sharing of x.



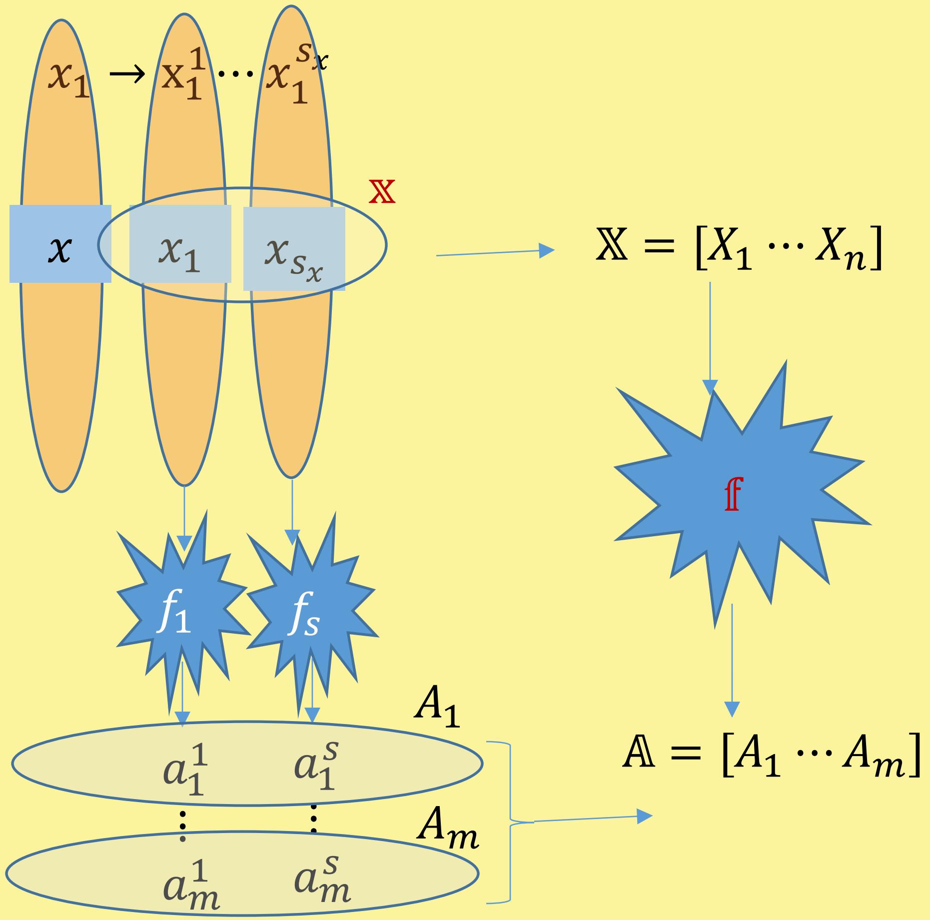
f is implemented as a vector of functions $\mathbb{f} = (f_1, \cdots, f_s)$.

Here each function is called component function.

Correctness

For all $a \in F_2^m, \mathbb{A} = \mathbb{f}(\mathbb{X})$, implies that $a = \Sigma_i a_i = \Sigma_i f_i(\mathbb{x})$, for all $\mathbb{x}$ satisfying $\Sigma x_i = x, x \in F_2^n$

$$f(x_1, \cdots, x_n) \rightarrow (a_1, \cdots, a_m)$$



NPTEL ONLINE CERTIFICATION COURSES

Thank you!