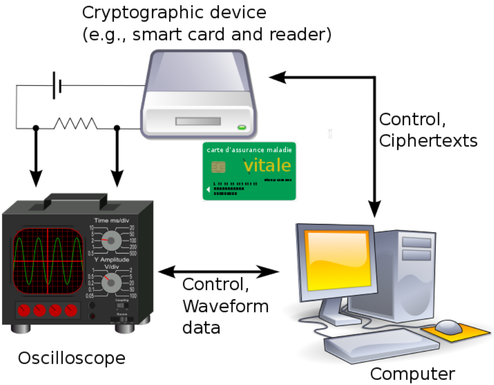
Side-Channel Attacks and Countermeasures

Jean-Sébastien Coron

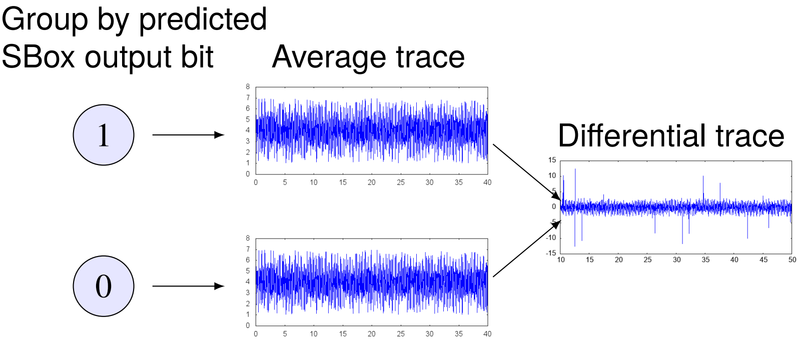
University of Luxembourg

Side-channel Attacks

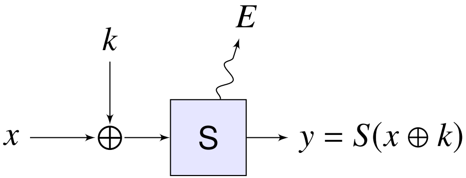
- Use side-channel information during execution
  - Timing attack, power attack, fault attack



Differential Power Analysis [KJJ99]



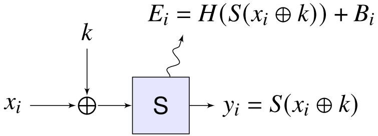
Side-channel Attack on SBOX computation



- SBOX computation $y = S(x \oplus k)$ for $x, k \in \{0,1\}^8$
  - We assume that the power consumption E is correlated to $S(x \oplus k)$
  - $E = H(S(x \oplus k)) + B$, where $H()$ is the Hamming weight and B is some noise.

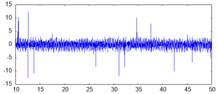
Statistical Analysis of Power Consumption

- We get many power acquisitions for unknown subkey k:



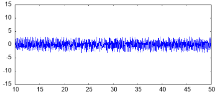
- Correct subkey k with $y_i = S(x_i \oplus k)$:

$$\mathrm{Corr}((E_i), (y_i)) \neq 0 \longrightarrow$$



- Incorrect subkey k' with $y'_i = S(x_i \oplus k')$:

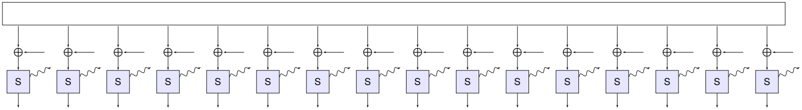
$$\mathrm{Corr}((E_i), (y'_i)) = 0 \longrightarrow$$



- We can distinguish the two and recover the subkey k

Recovering the secret-key

- For AES, we can apply the same attack separately on each of the 16 SBoxes of the first round



- Without countermeasures, only a few thousand power acquisitions are required to recover the secret-key.

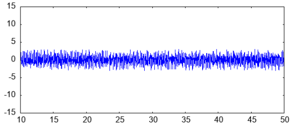
Countermeasure

Masking countermeasure

Let x be a variable dependent on the secret-key:

- Generate a random r (different for each execution)
- Mask x using r : $x' = x \oplus r$
- Manipulate x' (instead of x) and r independently

- r is random $\Rightarrow$ x' is random $\Rightarrow$ power consumption of x' is random $\Rightarrow$ no information on x leaks



☞ True only with one leakage point

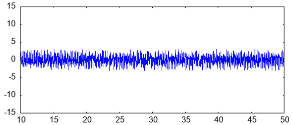
Countermeasure

Masking countermeasure

Let x be a variable dependent on the secret-key:

- Generate a random r (different for each execution)
- Mask x using r : $x' = x \oplus r$
- Manipulate x' (instead of x) and r independently

- r is random $\Rightarrow$ x' is random $\Rightarrow$ power consumption of x' is random $\Rightarrow$ no information on x leaks



☞ True only with one leakage point

First-order masking countermeasure

- How do we compute with $x' = x \oplus r$ instead of x ?

Linear operations: easy

$$x = x' \oplus r \quad \Rightarrow \quad f(x) = f(x') \oplus f(r)$$

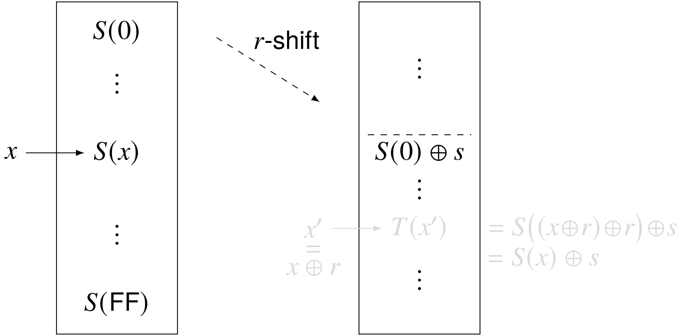
- We compute $f(x')$ and $f(r)$ separately.
- $f(x)$ is now masked with $f(r)$ instead of r
  - We can write $f(x) = (f(x') \oplus s \oplus f(r) \oplus r \oplus s) \oplus r$
  - $\Rightarrow \quad f(x)$ is still masked by r.
  - Example: `MixColumns` in AES

- Non-linear operations (SBOX): randomized table countermeasure [CJRR99]

Processing non-linear operations

Randomized table countermeasure [CJRR99]

- Table $S(x)$ is shifted as $T(u) = S(u \oplus r) \oplus s$
- One reads $y = T(x') = T(x \oplus r) = S(x) \oplus s$



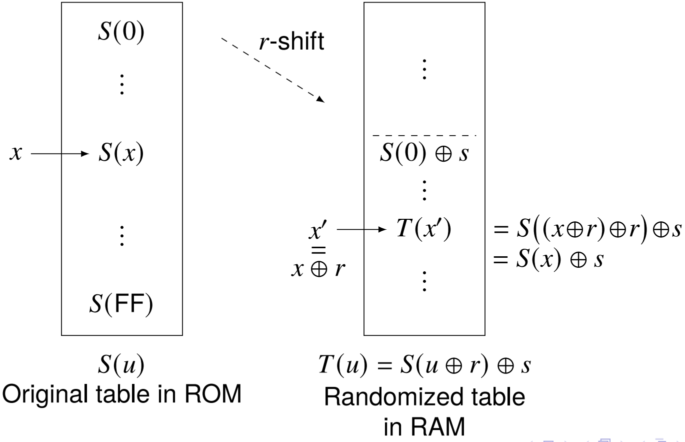
$S(u)$
Original table in ROM

$T(u) = S(u \oplus r) \oplus s$
Randomized table in RAM

Processing non-linear operations

Randomized table countermeasure [CJRR99]

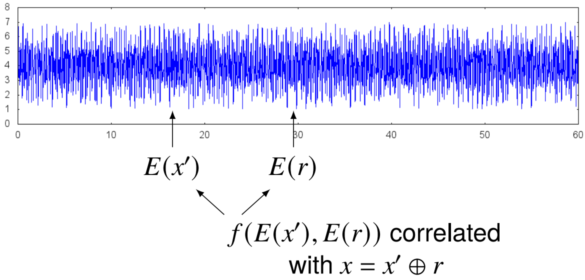
- Table $S(x)$ is shifted as $T(u) = S(u \oplus r) \oplus s$
- One reads $y = T(x') = T(x \oplus r) = S(x) \oplus s$



Second-order power attacks

Second-order DPA

- Combine the leakage of $x' = x \oplus r$ and the leakage of r to recover information about x
- Requires more power curves but can be practical



Solution: Higher-Order Boolean Masking

Basic principle

Each sensitive variable x is shared into n variables:

$$x = x_1 \oplus x_2 \oplus \cdots \oplus x_n$$

- Generate $n-1$ random variables $x_1, x_2, \ldots, x_{n-1}$
- Initially let $x_n = x \oplus x_1 \oplus x_2 \oplus \cdots \oplus x_{n-1}$

Security against DPA attack of order $n-1$

- Any subset of $n-1$ shares is uniformly and independently distributed

  $\Longrightarrow$ If we probe at most $n-1$ shares x_i, we learn nothing about x

Solution: Higher-Order Boolean Masking

Basic principle

Each sensitive variable x is shared into n variables:

$$x = x_1 \oplus x_2 \oplus \cdots \oplus x_n$$

- Generate $n-1$ random variables $x_1, x_2, \ldots, x_{n-1}$
- Initially let $x_n = x \oplus x_1 \oplus x_2 \oplus \cdots \oplus x_{n-1}$

Security against DPA attack of order $n-1$

- Any subset of $n-1$ shares is uniformly and independently distributed

$\Rightarrow$ If we probe at most $n-1$ shares x_i, we learn nothing about x

High-order masking of Boolean circuits

Ishai-Sahai-Wagner private circuit [ISW03]

- The adversary can probe any subset of at most t wires
- Algorithm to transform any Boolean circuit C of size $|C|$ into a circuit of size $O(|C| \cdot t^2)$ that is perfectly secure against such an adversary.

- Any Boolean circuit can be written with only Xor gates $c = a \oplus b$ and And gates $c = a \times b$.
  - High-order masking of $c = a \oplus b$: easy since linear.
  - High-order masking of $c = a \times b$: more complex.
- For security against t probes, one must use at least $n = 2t + 1$ shares.

High-order masking of $c = a \oplus b$

Computation of $a \oplus b$

- **Inputs:** $(a_i)_i$ and $(b_i)_i$ such that
  - $a_1 \oplus a_2 \oplus \cdots \oplus a_n = a$
  - $b_1 \oplus b_2 \oplus \cdots \oplus b_n = b$
- **Output:** $(c_i)_i$ such that
  - $(a_1 \oplus b_1) \oplus (a_2 \oplus b_2) \oplus \cdots \oplus (a_n \oplus b_n) = a \oplus b \quad \Rightarrow$
    $c_1 \oplus c_2 \oplus \cdots \oplus c_n = a \oplus b$

- We compute $c_i = a_i \oplus b_i$ independently for each i
- Complexity: $\mathcal{O}(n)$ for n shares
  $\Rightarrow$ very efficient

High-order secure multiplication

Secure Computation of $a \times b$

- **Inputs:** $(a_i)_i$ and $(b_i)_i$ such that
  - $a_1 \oplus a_2 \oplus \cdots \oplus a_n \;=\; a$
  - $b_1 \oplus b_2 \oplus \cdots \oplus b_n \;=\; b$
- **Output:** $(c_i)_i$ such that
  - $c_1 \oplus c_2 \oplus c_2 \oplus \cdots \oplus c_n \;=\; a \times b$

Ishai-Sahai-Wagner private circuit [ISW03]

- Secure against t probes for $n = 2t + 1$ shares.
- Number of operations: $\mathcal{O}(t^2)$
- Requires $\mathcal{O}(t^2)$ randoms per multiplication.

High-order secure multiplication

Secure Computation of $a \times b$

- **Inputs:** $(a_i)_i$ and $(b_i)_i$ such that
  - $a_1 \oplus a_2 \oplus \cdots \oplus a_n = a$
  - $b_1 \oplus b_2 \oplus \cdots \oplus b_n = b$
- **Output:** $(c_i)_i$ such that
  - $c_1 \oplus c_2 \oplus c_2 \oplus \cdots \oplus c_n = a \times b$

Ishai-Sahai-Wagner private circuit [ISW03]

- Secure against t probes for $n = 2t + 1$ shares.
- Number of operations: $\mathcal{O}(t^2)$
- Requires $\mathcal{O}(t^2)$ randoms per multiplication.

High-order secure multiplication (AND Gate)

- To high-order compute $c = a \times b$, one writes

$$c = a \times b = \left(\bigoplus_{i=1}^{n} a_i \right) \cdot \left(\bigoplus_{i=1}^{n} b_i \right)$$
$$= \bigoplus_{1 \leqslant i,j \leqslant n} a_i b_j$$

- The cross-products $a_i b_j$ are recombined without leaking information about the original inputs a and b.
  - For this, one needs $n(n-1)/2$ additional random bits r_{ij}.

The secure multiplication [ISW03]

Algo. SecMult

Input: $\bigoplus_i a_i = a$ and $\bigoplus_i b_i = b$

Output: shares c_i satisfying $\bigoplus_i c_i = a\,b$

for $i = 1$ **to** n
    for $j = i + 1$ **to** n
        $r_{i,j} \leftarrow \{0,1\}$
        $r_{j,i} \leftarrow (r_{i,j} \oplus a_i b_j) \oplus a_j b_i$
for $i = 1$ **to** n
    $c_i \leftarrow a_i b_i$
    for $j = 1$ **to** $n, j \neq i$ **do** $c_i \leftarrow c_i \oplus r_{i,j}$
return $(c_1, c_1, \ldots, c_n)$

$$\begin{pmatrix} a_1 b_1 & r_{1,2} & r_{1,3} \\ (r_{1,2} \oplus a_2 b_1) \oplus a_1 b_2 & a_2 b_2 & r_{2,3} \\ (r_{1,3} \oplus a_3 b_1) \oplus a_1 b_3 & (r_{2,3} \oplus a_3 b_2) \oplus a_2 b_3 & a_3 b_3 \end{pmatrix} \begin{matrix} \longrightarrow & c_1 \\ \longrightarrow & c_2 \\ \longrightarrow & c_3 \end{matrix}$$

Ishai-Sahai-Wagner (ISW) Scheme

Decomposition of the c_i

$$\bigoplus_i c_i = \left(\bigoplus_i a_i\right)\left(\bigoplus_i b_i\right) = \bigoplus_{i,j} a_i b_j$$

Example for $n = 3$

$$\begin{pmatrix} a_1b_1 & a_1b_2 & a_1b_3 \\ a_2b_1 & a_2b_2 & a_2b_3 \\ a_3b_1 & a_3b_2 & a_3b_3 \end{pmatrix}$$

☞ For n shares: requires $n(n-1)/2$ fresh random values

Ishai-Sahai-Wagner (ISW) Scheme

Decomposition of the c_i

$$\bigoplus_i c_i = \left(\bigoplus_i a_i\right)\left(\bigoplus_i b_i\right) = \bigoplus_{i,j} a_i b_j$$

Example for $n = 3$

$$\begin{pmatrix} a_1b_1 & a_1b_2 & a_1b_3 \\ a_2b_1 & a_2b_2 & a_2b_3 \\ a_3b_1 & a_3b_2 & a_3b_3 \end{pmatrix} \quad \begin{matrix} \longrightarrow c_1 \\ \longrightarrow c_2 \\ \longrightarrow c_3 \end{matrix}$$

☞ For n shares: requires $n(n-1)/2$ fresh random values

Ishai-Sahai-Wagner (ISW) Scheme

Decomposition of the c_i

$$\bigoplus_i c_i = \left(\bigoplus_i a_i\right)\left(\bigoplus_i b_i\right) = \bigoplus_{i,j} a_i b_j$$

Example for $n = 3$

$$\begin{pmatrix} a_1b_1 & 0 & 0 \\ a_2b_1 & a_2b_2 & 0 \\ a_3b_1 & a_3b_2 & a_3b_3 \end{pmatrix} \oplus \begin{pmatrix} 0 & a_1b_2 & a_1b_3 \\ 0 & 0 & a_2b_3 \\ 0 & 0 & 0 \end{pmatrix}$$

☞ For n shares: requires $n(n-1)/2$ fresh random values

Ishai-Sahai-Wagner (ISW) Scheme

Decomposition of the c_i

$$\bigoplus_i c_i = \left(\bigoplus_i a_i\right)\left(\bigoplus_i b_i\right) = \bigoplus_{i,j} a_i b_j$$

Example for $n = 3$

$$\begin{pmatrix} a_1b_1 & 0 & 0 \\ a_2b_1 & a_2b_2 & 0 \\ a_3b_1 & a_3b_2 & a_3b_3 \end{pmatrix} \oplus \begin{pmatrix} 0 & 0 & 0 \\ a_1b_2 & 0 & 0 \\ a_1b_3 & a_2b_3 & 0 \end{pmatrix}$$

☞ For n shares: requires $n(n-1)/2$ fresh random values

Ishai-Sahai-Wagner (ISW) Scheme

Decomposition of the c_i

$$\bigoplus_i c_i = \left(\bigoplus_i a_i\right)\left(\bigoplus_i b_i\right) = \bigoplus_{i,j} a_i b_j$$

Example for $n = 3$

$$\begin{pmatrix} a_1 b_1 & 0 & 0 \\ a_2 b_1 \oplus a_1 b_2 & a_2 b_2 & 0 \\ a_3 b_1 \oplus a_1 b_3 & a_3 b_2 \oplus a_2 b_3 & a_3 b_3 \end{pmatrix}$$

☞ For n shares: requires $n(n-1)/2$ fresh random values

Ishai-Sahai-Wagner (ISW) Scheme

Decomposition of the c_i

$$\bigoplus_i c_i = \left(\bigoplus_i a_i\right)\left(\bigoplus_i b_i\right) = \bigoplus_{i,j} a_i b_j$$

Example for $n = 3$

$$\begin{pmatrix} a_1 b_1 & 0 & 0 \\ a_2 b_1 \oplus a_1 b_2 & a_2 b_2 & 0 \\ a_3 b_1 \oplus a_1 b_3 & a_3 b_2 \oplus a_2 b_3 & a_3 b_3 \end{pmatrix}$$

☞ For n shares: requires $n(n-1)/2$ fresh random values

Ishai-Sahai-Wagner (ISW) Scheme

Decomposition of the c_i

$$\bigoplus_i c_i = \left(\bigoplus_i a_i\right)\left(\bigoplus_i b_i\right) = \bigoplus_{i,j} a_i b_j$$

Example for $n = 3$

$$\begin{pmatrix} a_1b_1 & 0 & 0 \\ a_2b_1 \oplus a_1b_2 & a_2b_2 & 0 \\ a_3b_1 \oplus a_1b_3 & a_3b_2 \oplus a_2b_3 & a_3b_3 \end{pmatrix} \oplus \begin{pmatrix} 0 & 0 & 0 \\ r_{1,2} & 0 & 0 \\ r_{1,3} & r_{2,3} & 0 \end{pmatrix}$$

☞ For n shares: requires $n(n-1)/2$ fresh random values

Ishai-Sahai-Wagner (ISW) Scheme

Decomposition of the c_i

$$\bigoplus_i c_i = \left(\bigoplus_i a_i\right)\left(\bigoplus_i b_i\right) = \bigoplus_{i,j} a_i b_j$$

Example for $n = 3$

$$\begin{pmatrix} a_1b_1 & 0 & 0 \\ a_2b_1 \oplus a_1b_2 & a_2b_2 & 0 \\ a_3b_1 \oplus a_1b_3 & a_3b_2 \oplus a_2b_3 & a_3b_3 \end{pmatrix} \oplus \begin{pmatrix} 0 & r_{1,2} & r_{1,3} \\ r_{1,2} & 0 & r_{2,3} \\ r_{1,3} & r_{2,3} & 0 \end{pmatrix}$$

☞ For n shares: requires $n(n-1)/2$ fresh random values

Ishai-Sahai-Wagner (ISW) Scheme

Decomposition of the c_i

$$\bigoplus_i c_i = \left(\bigoplus_i a_i\right)\left(\bigoplus_i b_i\right) = \bigoplus_{i,j} a_i b_j$$

Example for $n = 3$

$$\begin{pmatrix} a_1 b_1 & r_{1,2} & r_{1,3} \\ (r_{1,2} \oplus a_2 b_1) \oplus a_1 b_2 & a_2 b_2 & r_{2,3} \\ (r_{1,3} \oplus a_3 b_1) \oplus a_1 b_3 & (r_{2,3} \oplus a_3 b_2) \oplus a_2 b_3 & a_3 b_3 \end{pmatrix}$$

☞ For n shares: requires $n(n-1)/2$ fresh random values

Ishai-Sahai-Wagner (ISW) Scheme

Decomposition of the c_i

$$\bigoplus_i c_i = \left(\bigoplus_i a_i\right)\left(\bigoplus_i b_i\right) = \bigoplus_{i,j} a_i b_j$$

Example for $n = 3$

$$\begin{pmatrix} a_1 b_1 & r_{1,2} & r_{1,3} \\ (r_{1,2} \oplus a_2 b_1) \oplus a_1 b_2 & a_2 b_2 & r_{2,3} \\ (r_{1,3} \oplus a_3 b_1) \oplus a_1 b_3 & (r_{2,3} \oplus a_3 b_2) \oplus a_2 b_3 & a_3 b_3 \end{pmatrix} \begin{matrix} \to c_1 \\ \to c_2 \\ \to c_3 \end{matrix}$$

☞ For n shares: requires $n(n-1)/2$ fresh random values

Ishai-Sahai-Wagner (ISW) Scheme

Decomposition of the c_i

$$\bigoplus_i c_i = \left(\bigoplus_i a_i\right)\left(\bigoplus_i b_i\right) = \bigoplus_{i,j} a_i b_j$$

Example for $n = 3$

$$\begin{pmatrix} a_1 b_1 & r_{1,2} & r_{1,3} \\ (r_{1,2} \oplus a_2 b_1) \oplus a_1 b_2 & a_2 b_2 & r_{2,3} \\ (r_{1,3} \oplus a_3 b_1) \oplus a_1 b_3 & (r_{2,3} \oplus a_3 b_2) \oplus a_2 b_3 & a_3 b_3 \end{pmatrix} \begin{matrix} \longrightarrow c_1 \\ \longrightarrow c_2 \\ \longrightarrow c_3 \end{matrix}$$

☞ For n shares: requires $n(n-1)/2$ fresh random values

The secure multiplication [ISW03]

Algo. SecMult

Input: $\bigoplus_i a_i = a$ and $\bigoplus_i b_i = b$
Output: shares c_i satisfying $\bigoplus_i c_i = a\,b$

for $i = 1$ **to** n
  for $j = i + 1$ **to** n
    $r_{i,j} \leftarrow \{0,1\}$
    $r_{j,i} \leftarrow (r_{i,j} \oplus a_i b_j) \oplus a_j b_i$
for $i = 1$ **to** n
  $c_i \leftarrow a_i b_i$
  for $j = 1$ **to** $n, j \neq i$ **do** $c_i \leftarrow c_i \oplus r_{i,j}$
return $(c_1, c_1, \ldots, c_n)$

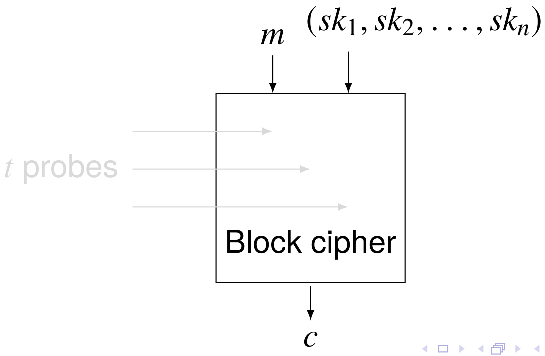
$$\begin{pmatrix} a_1 b_1 & r_{1,2} & r_{1,3} \\ (r_{1,2} \oplus a_2 b_1) \oplus a_1 b_2 & a_2 b_2 & r_{2,3} \\ (r_{1,3} \oplus a_3 b_1) \oplus a_1 b_3 & (r_{2,3} \oplus a_3 b_2) \oplus a_2 b_3 & a_3 b_3 \end{pmatrix} \begin{matrix} \longrightarrow & c_1 \\ \longrightarrow & c_2 \\ \longrightarrow & c_3 \end{matrix}$$

ISW security model

- The t-probing model
  - Protected block-cipher takes as input $n = 2t + 1$ shares sk_i of the secret key sk, with

$$sk = sk_1 \oplus \cdots \oplus sk_n$$

  - Prove that even if the attacker probes t variables in the block-cipher, he learns nothing about the secret-key sk.

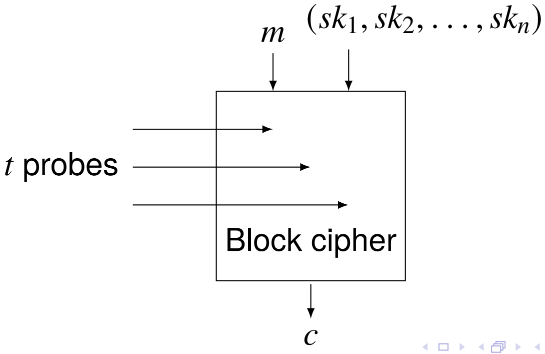


ISW security model

- The t-probing model
  - Protected block-cipher takes as input $n = 2t + 1$ shares sk_i of the secret key sk, with

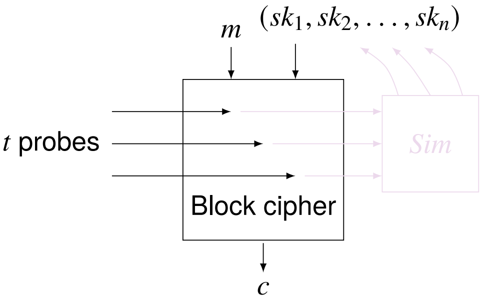
$$sk = sk_1 \oplus \cdots \oplus sk_n$$

  - Prove that even if the attacker probes t variables in the block-cipher, he learns nothing about the secret-key sk.



ISW security model

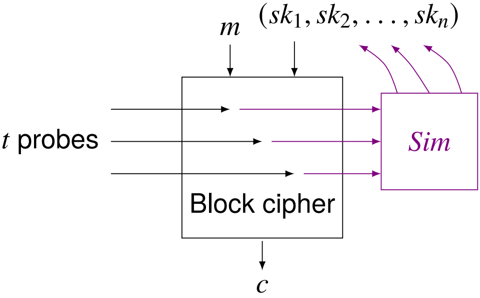
- Simulation framework of [ISW03]:



- Show that any t probes can be perfectly simulated from at most $n-1$ of the sk_i's.
- Those $n-1$ shares sk_i are initially uniformly and independently distributed.
- $\Longrightarrow$ the adversary learns nothing from the t probes, since he could simulate those t probes by himself.

ISW security model

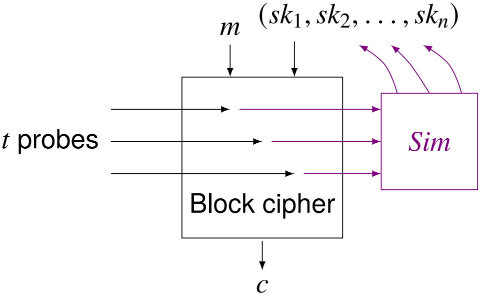
- Simulation framework of [ISW03]:



- Show that any t probes can be perfectly simulated from at most $n-1$ of the sk_i's.
- Those $n-1$ shares sk_i are initially uniformly and independently distributed.
- $\Rightarrow$ the adversary learns nothing from the t probes, since he could simulate those t probes by himself.

ISW security model

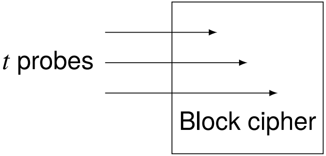
- Simulation framework of [ISW03]:



  - Show that any t probes can be perfectly simulated from at most $n-1$ of the sk_i's.
  - Those $n-1$ shares sk_i are initially uniformly and independently distributed.
  - $\Rightarrow$ the adversary learns nothing from the t probes, since he could simulate those t probes by himself.

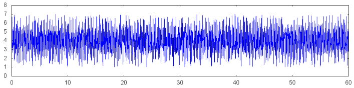
Probing Model vs. Reality

- Probing model
  - The attacker can choose at most t variables
  - He learns the value of those t variables.
- Reality with power attack
  - The attacker gets a sequence of power consumptions correlated to the variables.
  - Noisy leakage but not limited to t variables



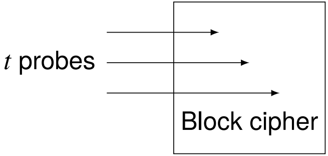
Probing model

Real life leakage



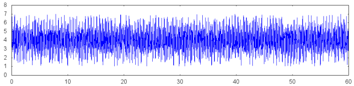
Relevance of probing model

- t-probing model
  - With security against t probes, combining t power consumption points as in a t-th order DPA will reveal no information to the adversary.
  - To recover the key, attacker must perform an attack of order at least $t + 1 \Rightarrow$ more complex.



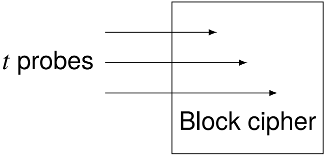
Probing model

Real life leakage



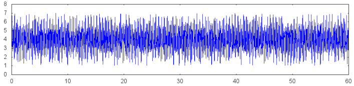
Probing Model vs. Reality

- Noisy leakage model
  - All variables leak independently with noise
  - Closer to reality
- Probing model vs noisy leakage model
  - Security in probing model $\Rightarrow$ security in noisy leakage model [DDF14]



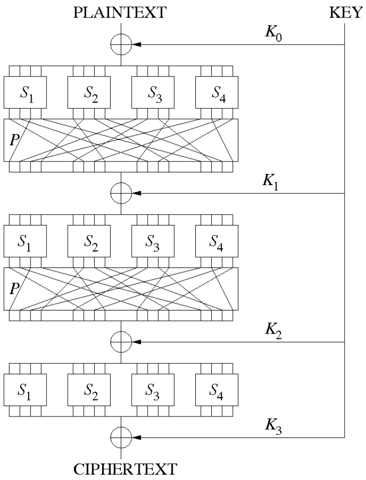
Probing model

Real life leakage



Application to masking AES

- AES: Substitution-permutation network (SPN)
  - Several rounds of SBoxes and linear layer.



High-order masking of AES

Ishai-Sahai-Wagner private circuit [ISW03]

- Transform any Boolean circuit C into a circuit C' of size $\mathcal{O}(|C| \cdot t^2)$ perfectly secure against t probes, using $n = 2t + 1$ shares.

- Masking AES: generic approach
  - First write AES as a Boolean circuit C and apply [ISW03], with complexity $\mathcal{O}(t^2)$.
  - too inefficient.
- Masking linear operations (MixColumns):
  - Easy: compute the $f(x_i)$ separately

$$x = x_1 \oplus x_2 \oplus \cdots \oplus x_n$$

$$f(x) = f(x_1) \oplus f(x_2) \oplus \cdots \oplus f(x_n)$$

High-order masking of AES

Ishai-Sahai-Wagner private circuit [ISW03]

- Transform any Boolean circuit C into a circuit C' of size $\mathcal{O}(|C| \cdot t^2)$ perfectly secure against t probes, using $n = 2t + 1$ shares.

- Masking AES: generic approach
  - First write AES as a Boolean circuit C and apply [ISW03], with complexity $\mathcal{O}(t^2)$.
  - too inefficient.
- Masking linear operations (MixColumns):
  - Easy: compute the $f(x_i)$ separately

$$x = x_1 \oplus x_2 \oplus \cdots \oplus x_n$$
$$f(x) = f(x_1) \oplus f(x_2) \oplus \cdots \oplus f(x_n)$$

Secure SBox Computation

Secure Computation of $S(x)$

- **Inputs:** $(x_i)_i$ such that
  - $x_1 \oplus x_2 \oplus \cdots \oplus x_n \;=\; x$
- **Output:** $(y_i)_i$ such that
  - $y_1 \oplus y_2 \oplus \cdots \oplus y_n \;=\; S(x)$

[RP10] countermeasure **for AES**: compute $S(x) = x^{254}$

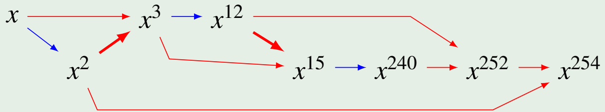
- 4 multiplications over $\mathbb{F}_{2^8}$ with ISW
- 7 linear squarings

Secure SBox Computation

Secure Computation of $S(x)$

- **Inputs:** $(x_i)_i$ such that
  - $x_1 \oplus x_2 \oplus \cdots \oplus x_n = x$
- **Output:** $(y_i)_i$ such that
  - $y_1 \oplus y_2 \oplus \cdots \oplus y_n = S(x)$

[RP10] countermeasure **for AES**: compute $S(x) = x^{254}$



- 4 multiplications over $\mathbb{F}_{2^8}$ with ISW
- 7 linear squarings

Secure multiplication over $\mathbb{F}_{2^8}$: ISW

- Goal: compute $c = a \cdot b$ securely over $\mathbb{F}_{2^8}$

Decomposition of the c_i over $\mathbb{F}_{2^8}$

$$\bigoplus_i c_i = \left(\bigoplus_i a_i\right)\left(\bigoplus_i b_i\right) = \bigoplus_{i,j} a_i b_j$$

Example of ISW over $\mathbb{F}_{2^8}$ for $n = 3$

$$\begin{pmatrix} a_1 b_1 & r_{1,2} & r_{1,3} \\ (r_{1,2} \oplus a_2 b_1) \oplus a_1 b_2 & a_2 b_2 & r_{2,3} \\ (r_{1,3} \oplus a_3 b_1) \oplus a_1 b_3 & (r_{2,3} \oplus a_3 b_2) \oplus a_2 b_3 & a_3 b_3 \end{pmatrix} \begin{matrix} \longrightarrow & c_1 \\ \longrightarrow & c_2 \\ \longrightarrow & c_3 \end{matrix}$$

Summary: high-order masking of AES

- High-order masking of AES
  - Input: n shares $sk = sk_1 \oplus sk_2 \oplus \cdots \oplus sk_n$, and a message m
  - First encode $m = m_1 \oplus \cdots \oplus m_n$
  - Process linear operations with n shares (easy)
  - For SBoxes, write $x^3 = x \times x^2$ and $S(x) = x^{254} = (x)^2 \times (x^3)^4 \times (x^3 \times (x^3)^4)^{16} \in \mathbb{F}_{2^8}$
  - Apply ISW for secure multiplication over $\mathbb{F}_{2^8}$
  - Output: decode $c = c_1 \oplus \cdots \oplus c_n$
- Complexity: $\mathcal{O}(n^2)$

Security

- Provably secure against t probes with $n = 2t + 1$ shares
  - Possible with $n = t + 1$ shares using mask refreshing

Extension to any SBOX

- Use Lagrange interpolation over $\mathbb{F}_{2^k}$ [CGP12]

$$S(x) = \sum_{i=0}^{2^k-1} \alpha_i \cdot x^i$$

over $\mathbb{F}_{2^k}$, for constant coefficients $\alpha_i \in \mathbb{F}_{2^k}$.

  - One can evaluate the polynomial with only $\mathcal{O}(2^{k/2})$ multiplications.
  - Asymptotic complexity is therefore $\mathcal{O}(2^{k/2} \cdot n^2)$.

Proof of security for ISW multiplication

- Input: a_i and b_i
- Output: c_i such that $\bigoplus_i c_i = (\bigoplus_i a_i) \cdot (\bigoplus_i b_i)$
- Algorithm: for each $1 \leqslant i < j \leqslant n$, let $r_{ij} \leftarrow \{0, 1\}$ and

$$z_{ij} \leftarrow r_{ij}$$
$$z_{ji} \leftarrow (z_{ij} \oplus a_i b_j) \oplus a_j b_i$$
$$c_i \leftarrow a_i b_i \oplus \bigoplus_{j \neq i} z_{ij}$$

Security property

- Any set of t probes can be perfectly simulated with the knowledge of $a_{|I}$ and $b_{|I}$, for some subset I with $|I| \leqslant 2t$
- where $a_{|I} = (a_i)_{i \in I}$.

Proof of security for ISW multiplication

- Construction of the set I.
  - Initially $I \leftarrow \emptyset$.
  - If a wire a_i, b_i, $a_i b_i$, z_{ij} (for $i \neq j$) is probed, add i to I.
  - Same for a sum of values of the above form, including c_i.
  - For the wires $a_i b_j$ or $z_{ij} \oplus a_i b_j$ for $i \neq j$, add both i, j to I
  - We have $|I| \leqslant 2t$

$$\begin{pmatrix} a_1 b_1 & \cdots & z_{1,i} & \cdots & z_{1,n} \\ \vdots & \ddots & & & \vdots \\ z_{i,1} & \cdots & a_i b_i & \cdots & z_{i,n} \\ \vdots & & & \ddots & \vdots \\ z_{n,1} & \cdots & z_{n,i} & \cdots & a_n b_n \end{pmatrix} \begin{matrix} c_1 \\ \vdots \\ c_i \\ \vdots \\ c_n \end{matrix}$$

Simulation of the probes

- We must show that all probes can be perfectly simulated using only $a_{|I}$ and $b_{|I}$
  - Simulation of probed a_i, b_i, $a_i b_i$: obvious since $i \in I$
  - Same for probed $a_i b_j$ and $z_{ij} \oplus a_i b_j$, since $i, j \in I$
  - There remains the probed z_{ij}'s and sums of z_{ij}'s, including c_i. We must have $i \in I$.
- We would like to show that if $i \in I$, we can simulate all z_{ij} for $i \neq j$.

$$\begin{pmatrix} a_1 b_1 & \cdots & z_{1,i} & \cdots & z_{1,n} \\ \vdots & \ddots & & & \vdots \\ z_{i,1} & \cdots & a_i b_i & \cdots & z_{i,n} \\ \vdots & & & \ddots & \vdots \\ z_{n,1} & \cdots & z_{n,i} & \cdots & a_n b_n \end{pmatrix} \begin{matrix} c_1 \\ \vdots \\ c_i \\ \vdots \\ c_n \end{matrix}$$

Simulation of row i for $i \in I$

- Goal: show that in row i for $i \in I$, we can simulate all $z_{i,j}$ for $i \neq j$.
  - Therefore we can also simulate the partial sums of z_{ij}, and the final sum c_i.
- Simulation of z_{ij} for $j > i$
  - Easy because $z_{ij} = r_{ij}$ where $r_{ij} \leftarrow \{0, 1\}$

$$\begin{pmatrix} a_1 b_1 & \cdots & z_{1,i} & & \cdots & & z_{1,n} \\ \vdots & \ddots & & & & & \vdots \\ z_{i,1} & \cdots & a_i b_i & \cdots & z_{i,j} & \cdots & z_{i,n} \\ \vdots & & & & \ddots & & \vdots \\ z_{n,1} & \cdots & z_{n,i} & & \cdots & & a_n b_n \end{pmatrix} \begin{matrix} c_1 \\ \vdots \\ c_i \\ \vdots \\ c_n \end{matrix}$$

Simulation of row i for $i \in I$

- Simulation of z_{ij} for $j < i$:

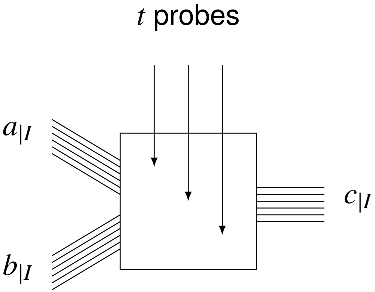
$$z_{ij} = (z_{ji} \oplus a_j b_i) \oplus a_i b_j$$

  - where $z_{ji} = r_{ji}$ with $r_{ji} \leftarrow \{0, 1\}$
- If $j \in I$, easy, since we know a_i, b_i, a_j and b_j.
- If $j \notin I$, then z_{ji} is not used in another probe.
  - Nothing in row j has been probed, otherwise $j \in I$.
  - z_{ji} is a one-time-pad, so we can simulate z_{ij} as $z_{ij} \leftarrow \{0, 1\}$, without knowing a_j and b_j.

$$\begin{pmatrix} a_1 b_1 & & \cdots & & z_{1,i} & \cdots & z_{1,n} \\ \vdots & & \ddots & & z_{j,i} & & \vdots \\ z_{i,1} & \cdots & z_{i,j} & \cdots & a_i b_i & \cdots & z_{i,n} \\ \vdots & & & & & \ddots & \vdots \\ z_{n,1} & & \cdots & & z_{n,i} & \cdots & a_n b_n \end{pmatrix} \begin{matrix} c_1 \\ \vdots \\ c_i \\ \vdots \\ c_n \end{matrix}$$

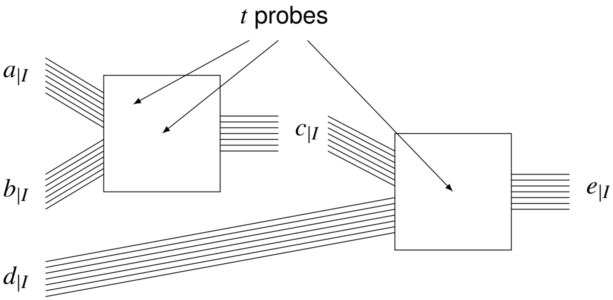
Summary: simulation for a single gate

- For a single gate, we can simulate any set of t probes
  - using a subset $a_{|I}$ and $b_{|I}$ of the input shares, for $|I| \leqslant 2t$.
  - We can also simulate the output shares $c_{|I}$



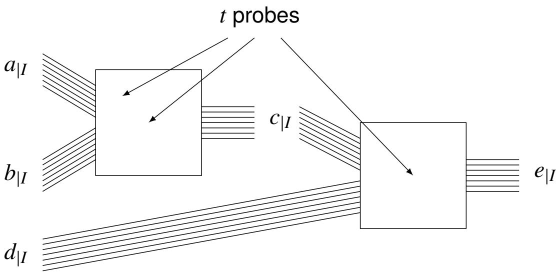
Simulation for a full circuit

- Simulation for a full circuit:
  - We examine all gadgets as previously, building a common set I, still with $|I| \leqslant 2t$
  - We can perform the simulation inductively, from input to output, using only the shares in I.



Simulation for a full circuit

- Simulation for a full circuit:
  - With $|I| \leqslant 2t < n$, the input variables in $a_{|I}$, $b_{|I}$, $d_{|I}$ can be perfectly simulated by generating random bits.



Security of ISW transform [ISW03]

- Any circuit C can be transformed into a circuit of size $O(|C| \cdot t^2)$ perfectly secure against t probes.

Conclusion

- Side-channel attacks
  - Timing attack, power attack, fault attack
- Side-channel countermeasures
  - Generic high-order Boolean masking: provable security against t probes with [ISW03], with complexity $\mathcal{O}(t^2)$
  - High-order masking of AES: ISW multiplication over $\mathbb{F}_{2^8}$ [RP10]
- New: post-quantum algorithms (Kyber, Dilithium)
  - Usually combine arithmetic and Boolean operations
  - Conversion between Boolean and arithmetic masking
  - High-order polynomial comparison for FO transform

References

CJRR99 *Towards Sound Approaches to Counteract Power-Analysis Attacks*. Suresh Chari, Charanjit S. Jutla, Josyula R. Rao, Pankaj Rohatgi. CRYPTO'99.

CGP12 *Higher-Order Masking Schemes for S-Boxes*. Claude Carlet, Louis Goubin, Emmanuel Prouff, Michaël Quisquater, Matthieu Rivain. FSE 2012.

ISW03 *Private Circuits: Securing Hardware against Probing Attacks.* Yuval Ishai, Amit Sahai, David Wagner, CRYPTO'03

RP10 *Provably Secure Higher-Order Masking of AES.* Matthieu Rivain, Emmanuel Prouff, CHES'10.

DDF14 *Unifying Leakage Models: from Probing Attacks to Noisy Leakage.* Duc, Dziembowski, Faust, EUROCRYPT'14