Side-Channel Attacks and Countermeasures

Sonia Belaïd

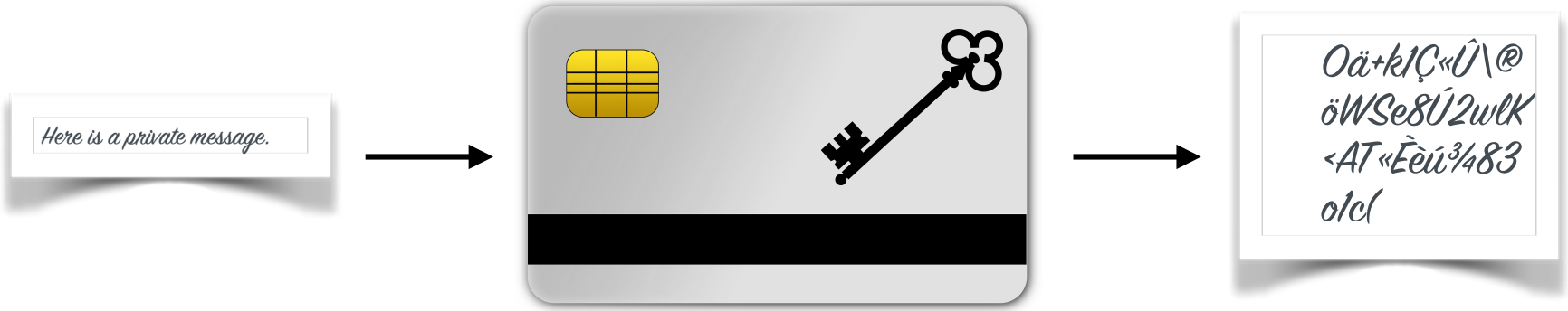
ASCrypto 2019

Overview

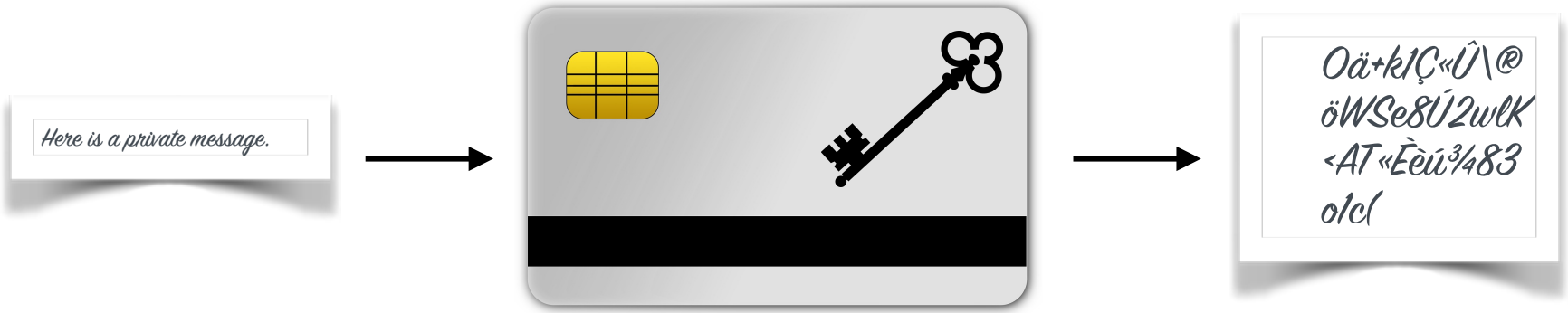
- What are side-channel attacks?
  - Definition, examples
- How to thwart side-channel attacks?
  - Countermeasures
- How to make sure that you did it?
  - Proofs, automatic tools

Side-Channel Attacks

Side-Channel Attacks

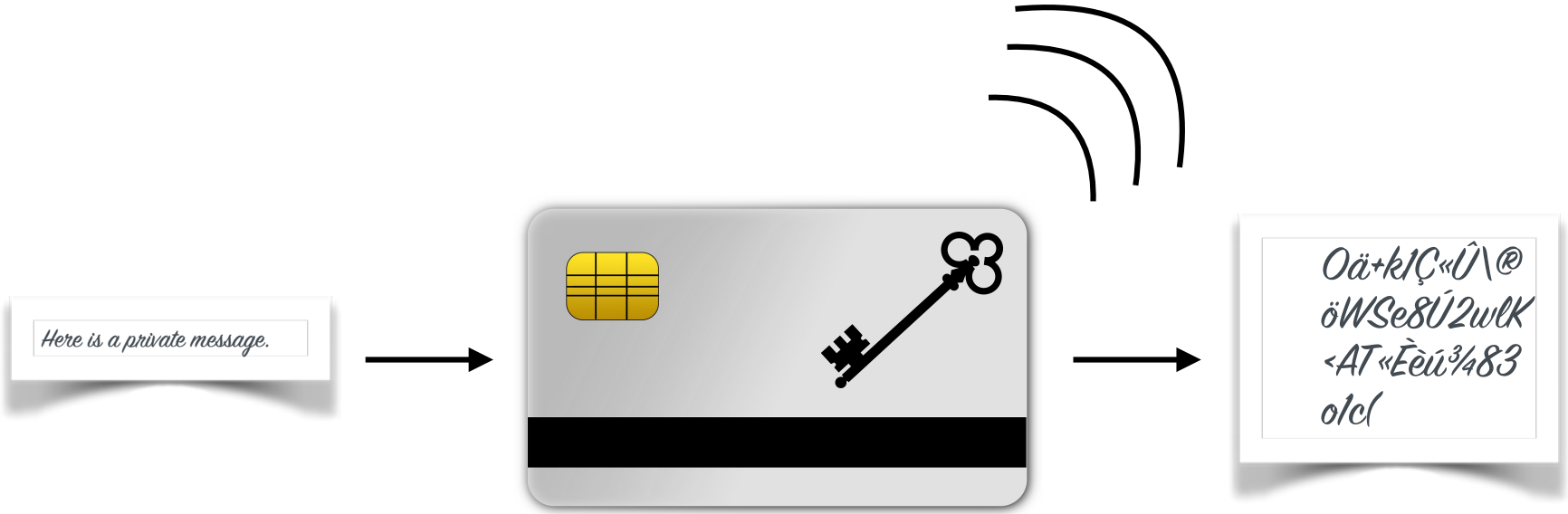


Side-Channel Attacks



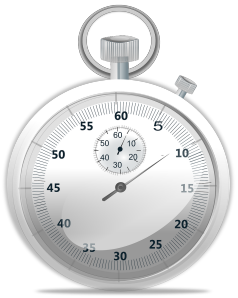
- Black-box cryptanalysis: $\mathcal{A} \leftarrow (m, c)$

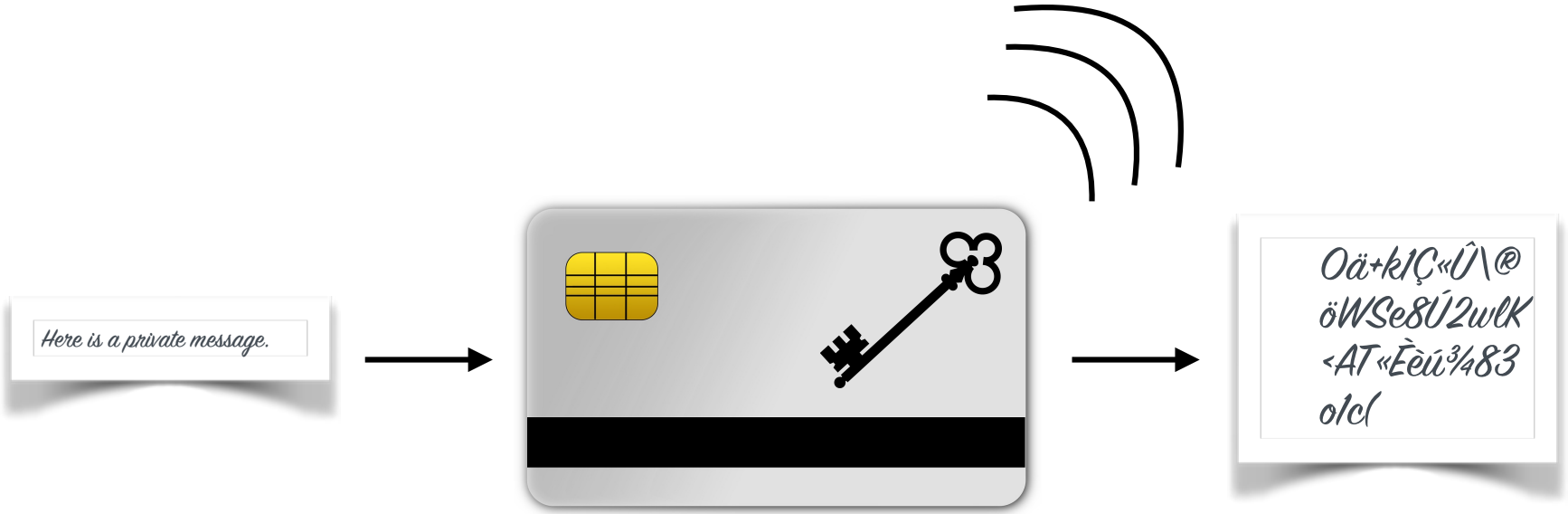
Side-Channel Attacks



- Black-box cryptanalysis: $\mathscr{A} \leftarrow (m, c)$
- Side-channel analysis: $\mathscr{A} \leftarrow (m, c, \mathscr{L})$

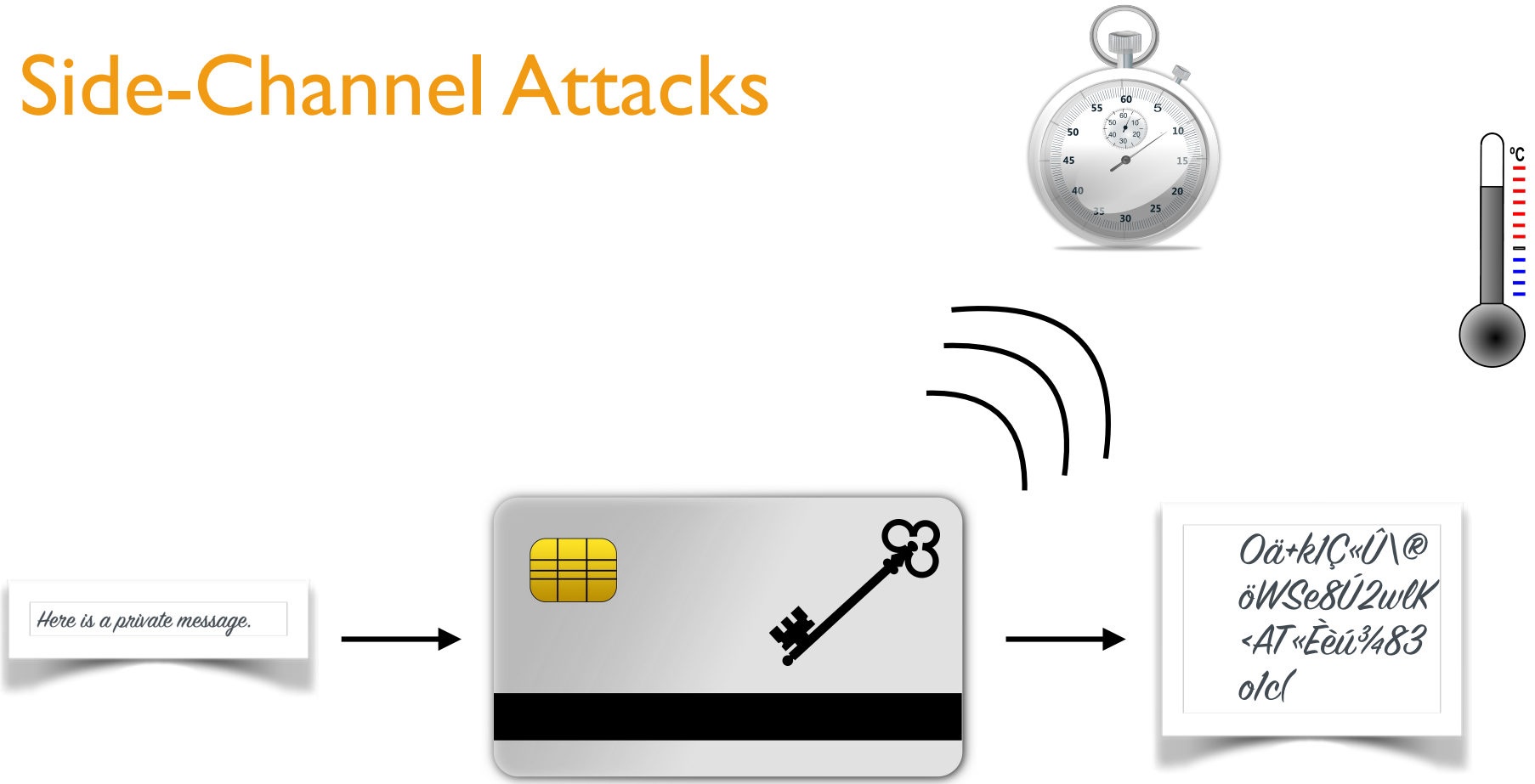
Side-Channel Attacks





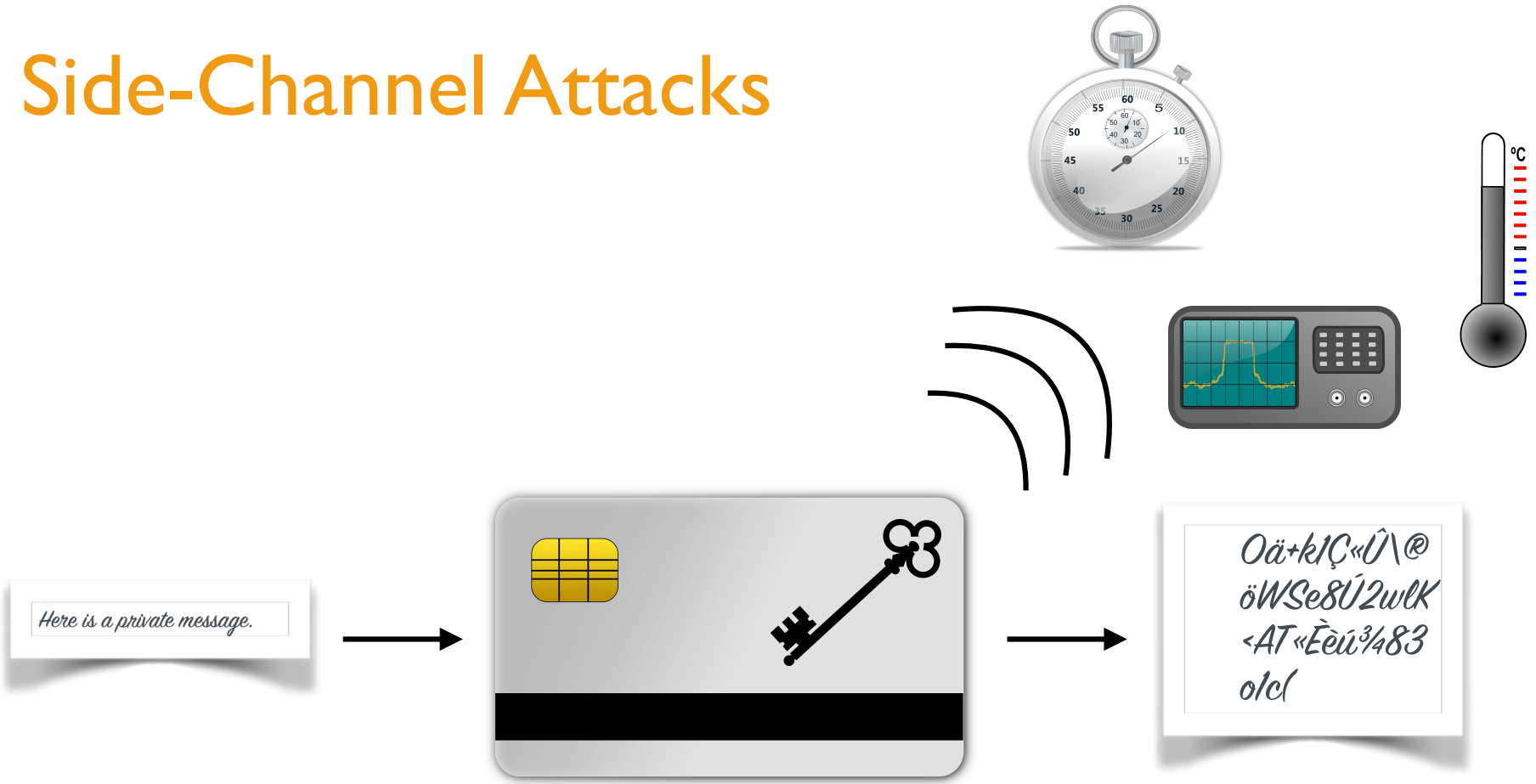
- Black-box cryptanalysis: $\mathscr{A} \leftarrow (m, c)$
- Side-channel analysis: $\mathscr{A} \leftarrow (m, c, \mathscr{L})$

Side-Channel Attacks



- Black-box cryptanalysis: $\mathscr{A} \leftarrow (m, c)$
- Side-channel analysis: $\mathscr{A} \leftarrow (m, c, \mathscr{L})$

Side-Channel Attacks

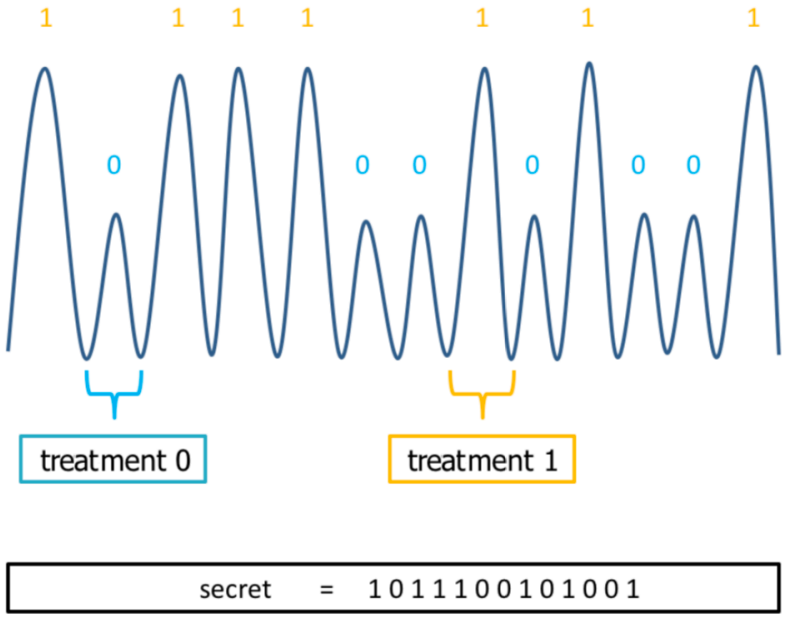


- Black-box cryptanalysis: $\mathscr{A} \leftarrow (m, c)$
- Side-channel analysis: $\mathscr{A} \leftarrow (m, c, \mathscr{L})$

Example of SPA

Algorithm 1 Example

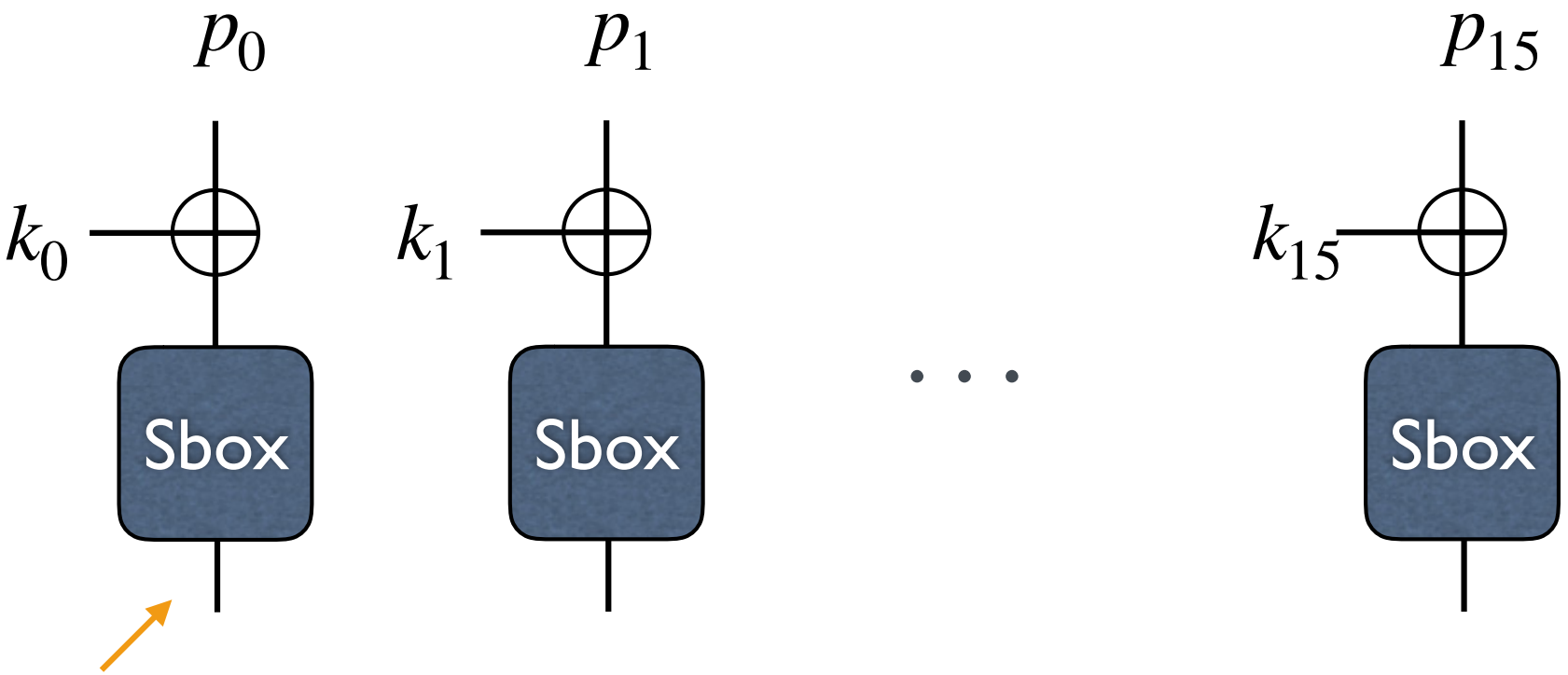
```
for i = 1 to n do
    if key[i] = 0 then
        do treatment 0
    else
        do treatment 1
    end if
end for
```



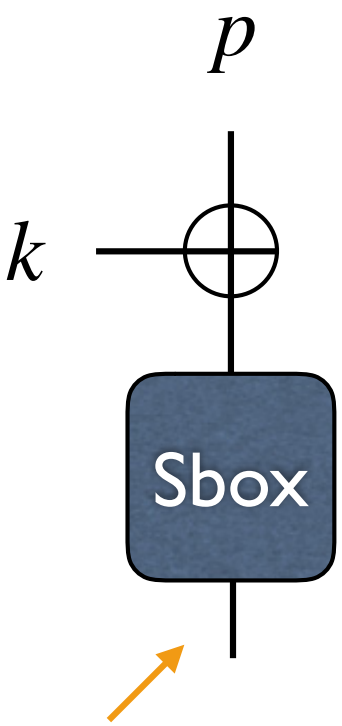
SPA: one single trace to recover the secret key

Example of DPA

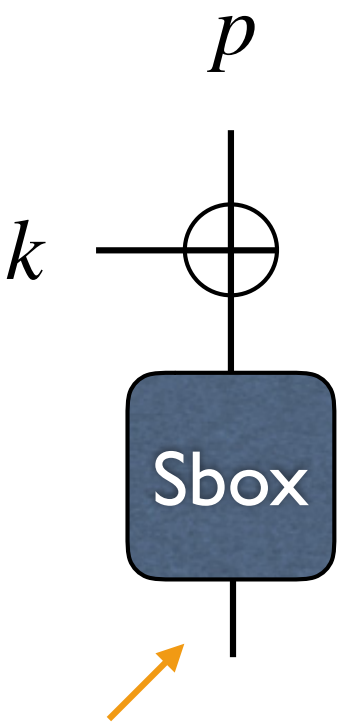
- AES
  - plaintext and key on 16 bytes
  - First round: 16 S-boxes

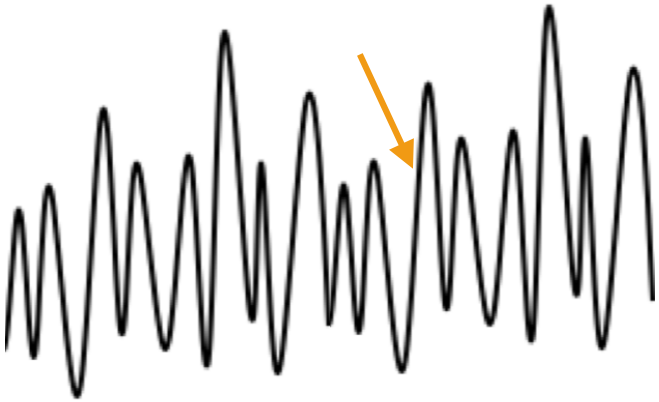


Example of DPA

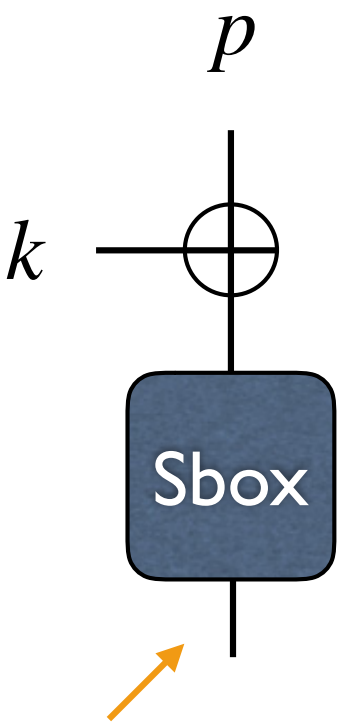


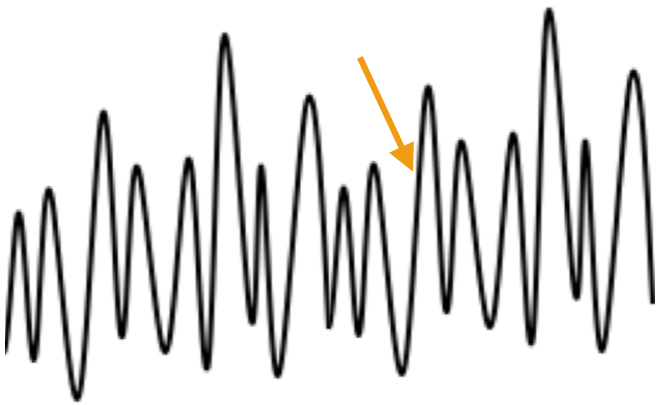
Example of DPA





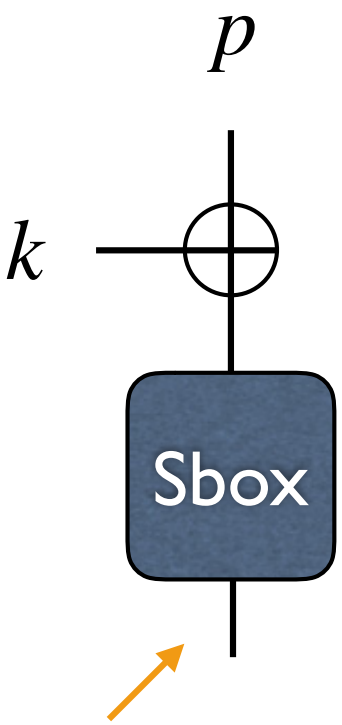
Example of DPA

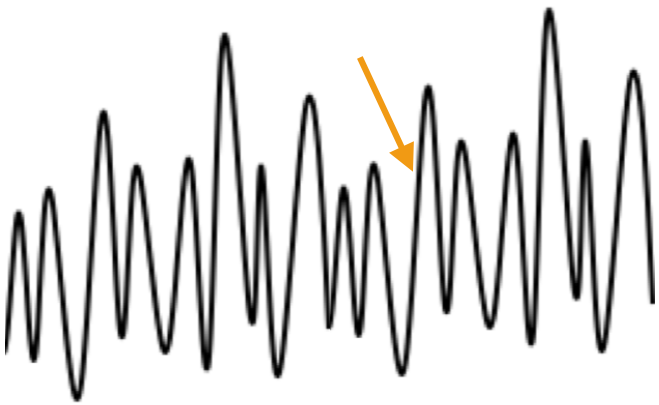




$$\begin{pmatrix} v_0 = \mathcal{L}(p_0) \\ v_1 = \mathcal{L}(p_1) \\ v_2 = \mathcal{L}(p_2) \\ \cdots \\ v_{n-1} = \mathcal{L}(p_{n-1}) \end{pmatrix}$$

Example of DPA

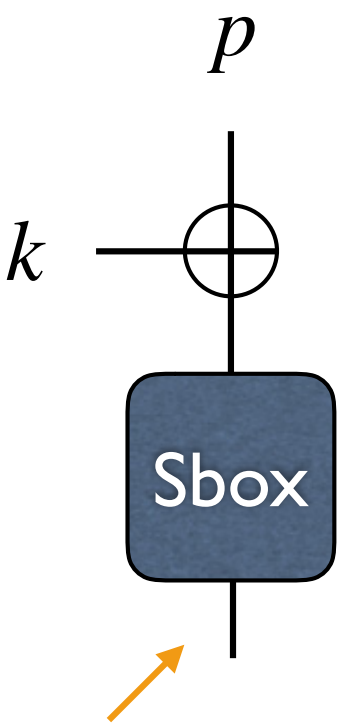


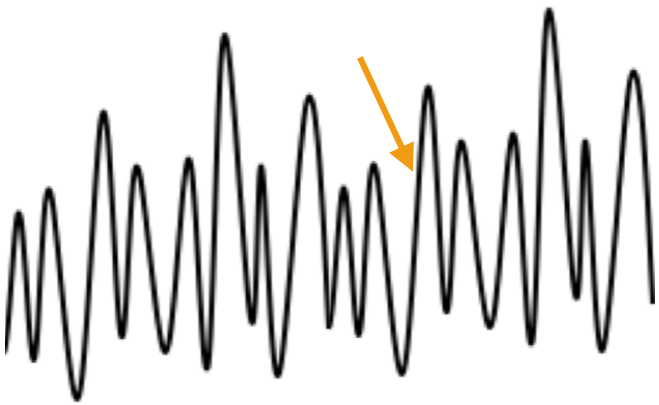


$$\begin{pmatrix} v_0 = \mathcal{L}(p_0) \\ v_1 = \mathcal{L}(p_1) \\ v_2 = \mathcal{L}(p_2) \\ \cdots \\ v_{n-1} = \mathcal{L}(p_{n-1}) \end{pmatrix}$$

$$\begin{pmatrix} \text{Sbox}(p_0 \oplus \mathbf{0}) \\ \text{Sbox}(p_1 \oplus \mathbf{0}) \\ \text{Sbox}(p_2 \oplus \mathbf{0}) \\ \cdots \\ \text{Sbox}(p_{n-1} \oplus \mathbf{0}) \end{pmatrix}$$

Example of DPA

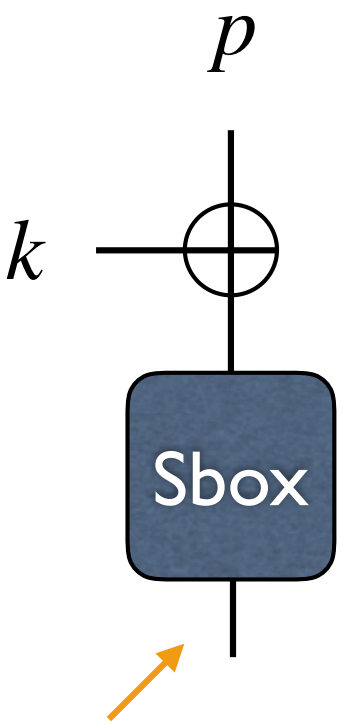


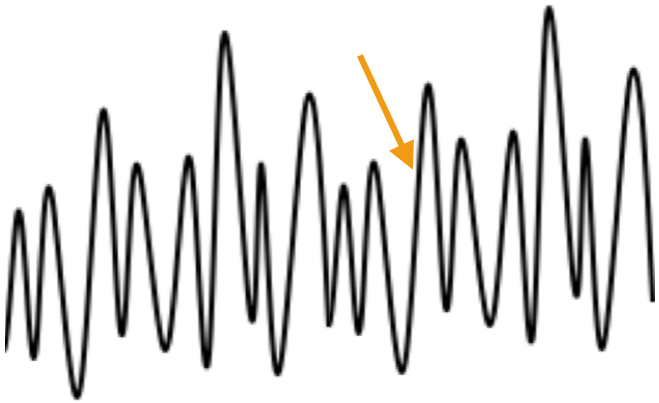


$$\begin{pmatrix} v_0 = \mathcal{L}(p_0) \\ v_1 = \mathcal{L}(p_1) \\ v_2 = \mathcal{L}(p_2) \\ \cdots \\ v_{n-1} = \mathcal{L}(p_{n-1}) \end{pmatrix}$$

$$\begin{pmatrix} \text{Sbox}(p_0 \oplus \mathbf{0}) \\ \text{Sbox}(p_1 \oplus \mathbf{0}) \\ \text{Sbox}(p_2 \oplus \mathbf{0}) \\ \cdots \\ \text{Sbox}(p_{n-1} \oplus \mathbf{0}) \end{pmatrix} \qquad \begin{pmatrix} \text{HW}(\text{Sbox}(p_0 \oplus \mathbf{0})) \\ \text{HW}(\text{Sbox}(p_1 \oplus \mathbf{0})) \\ \text{HW}(\text{Sbox}(p_2 \oplus \mathbf{0})) \\ \cdots \\ \text{HW}(\text{Sbox}(p_{n-1} \oplus \mathbf{0})) \end{pmatrix}$$

Example of DPA



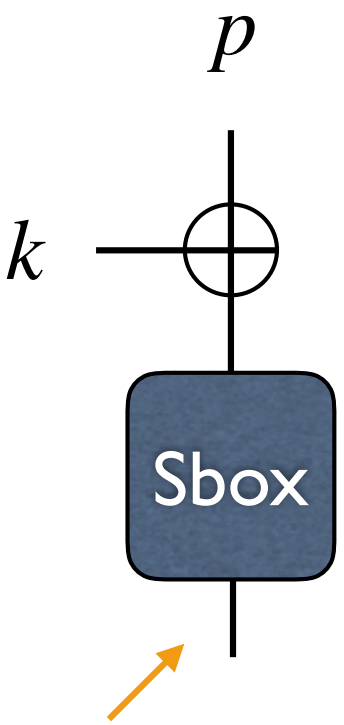


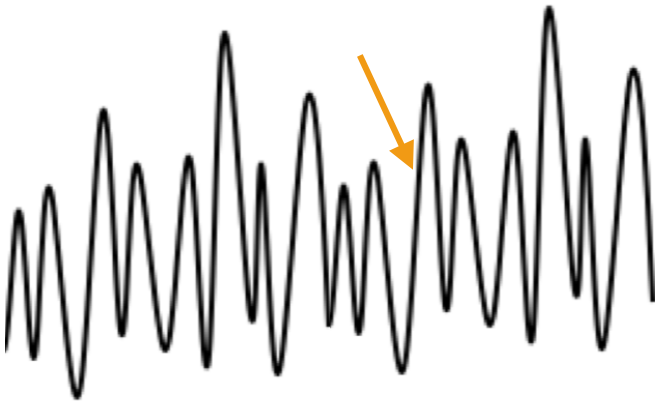
$$\begin{pmatrix} v_0 = \mathscr{L}(p_0) \\ v_1 = \mathscr{L}(p_1) \\ v_2 = \mathscr{L}(p_2) \\ \cdots \\ v_{n-1} = \mathscr{L}(p_{n-1}) \end{pmatrix}$$

$\mathscr{C}_0$

$$\begin{pmatrix} \mathsf{Sbox}(p_0 \oplus \mathbf{0}) \\ \mathsf{Sbox}(p_1 \oplus \mathbf{0}) \\ \mathsf{Sbox}(p_2 \oplus \mathbf{0}) \\ \cdots \\ \mathsf{Sbox}(p_{n-1} \oplus \mathbf{0}) \end{pmatrix} \qquad \begin{pmatrix} \mathsf{HW}(\mathsf{Sbox}(p_0 \oplus \mathbf{0})) \\ \mathsf{HW}(\mathsf{Sbox}(p_1 \oplus \mathbf{0})) \\ \mathsf{HW}(\mathsf{Sbox}(p_2 \oplus \mathbf{0})) \\ \cdots \\ \mathsf{HW}(\mathsf{Sbox}(p_{n-1} \oplus \mathbf{0})) \end{pmatrix}$$

Example of DPA



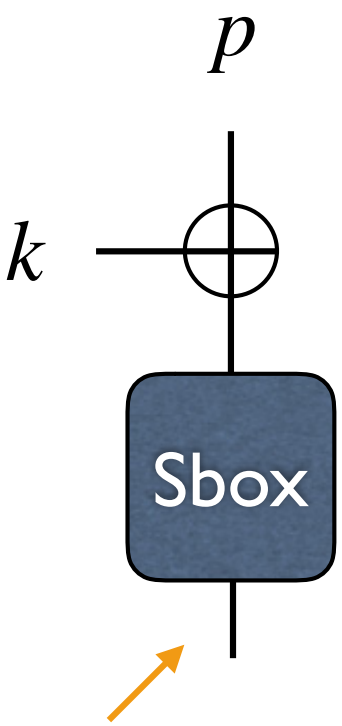


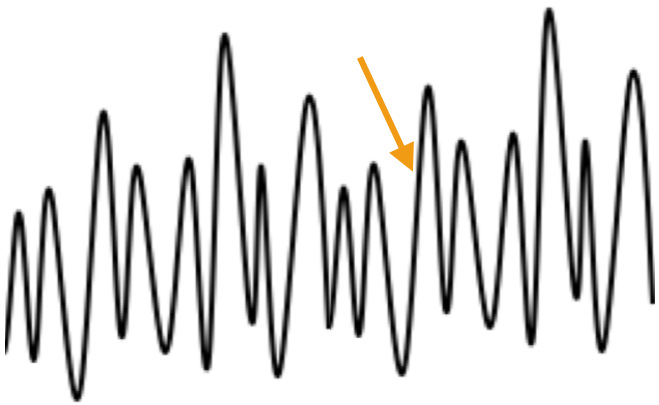
$$\begin{pmatrix} v_0 = \mathcal{L}(p_0) \\ v_1 = \mathcal{L}(p_1) \\ v_2 = \mathcal{L}(p_2) \\ \cdots \\ v_{n-1} = \mathcal{L}(p_{n-1}) \end{pmatrix}$$

$$(\mathscr{C}_0, \mathscr{C}_1)$$

$$\begin{pmatrix} \mathsf{Sbox}(p_0 \oplus \mathbf{1}) \\ \mathsf{Sbox}(p_1 \oplus \mathbf{1}) \\ \mathsf{Sbox}(p_2 \oplus \mathbf{1}) \\ \cdots \\ \mathsf{Sbox}(p_{n-1} \oplus \mathbf{1}) \end{pmatrix} \qquad \begin{pmatrix} \mathsf{HW}(\mathsf{Sbox}(p_0 \oplus \mathbf{1})) \\ \mathsf{HW}(\mathsf{Sbox}(p_1 \oplus \mathbf{1})) \\ \mathsf{HW}(\mathsf{Sbox}(p_2 \oplus \mathbf{1})) \\ \cdots \\ \mathsf{HW}(\mathsf{Sbox}(p_{n-1} \oplus \mathbf{1})) \end{pmatrix}$$

Example of DPA



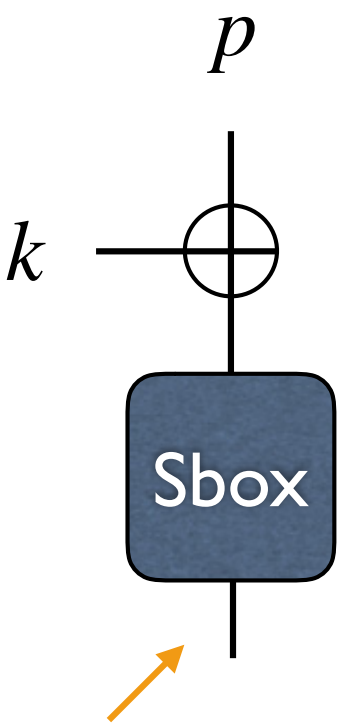


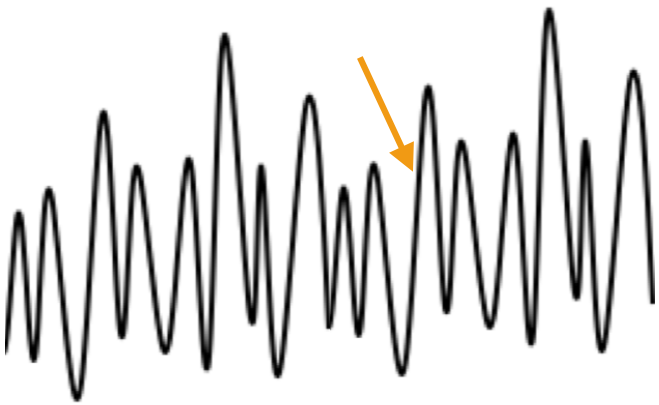
$$\begin{pmatrix} v_0 = \mathcal{L}(p_0) \\ v_1 = \mathcal{L}(p_1) \\ v_2 = \mathcal{L}(p_2) \\ \cdots \\ v_{n-1} = \mathcal{L}(p_{n-1}) \end{pmatrix}$$

$$(\mathscr{C}_0, \mathscr{C}_1, \mathscr{C}_2)$$

$$\begin{pmatrix} \mathsf{Sbox}(p_0 \oplus \mathbf{2}) \\ \mathsf{Sbox}(p_1 \oplus \mathbf{2}) \\ \mathsf{Sbox}(p_2 \oplus \mathbf{2}) \\ \cdots \\ \mathsf{Sbox}(p_{n-1} \oplus \mathbf{2}) \end{pmatrix} \qquad \begin{pmatrix} \mathsf{HW}(\mathsf{Sbox}(p_0 \oplus \mathbf{2})) \\ \mathsf{HW}(\mathsf{Sbox}(p_1 \oplus \mathbf{2})) \\ \mathsf{HW}(\mathsf{Sbox}(p_2 \oplus \mathbf{2})) \\ \cdots \\ \mathsf{HW}(\mathsf{Sbox}(p_{n-1} \oplus \mathbf{2})) \end{pmatrix}$$

Example of DPA





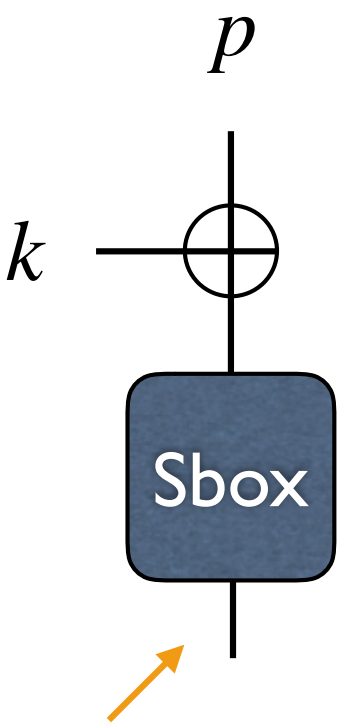
$$\begin{pmatrix} v_0 = \mathcal{L}(p_0) \\ v_1 = \mathcal{L}(p_1) \\ v_2 = \mathcal{L}(p_2) \\ \cdots \\ v_{n-1} = \mathcal{L}(p_{n-1}) \end{pmatrix}$$

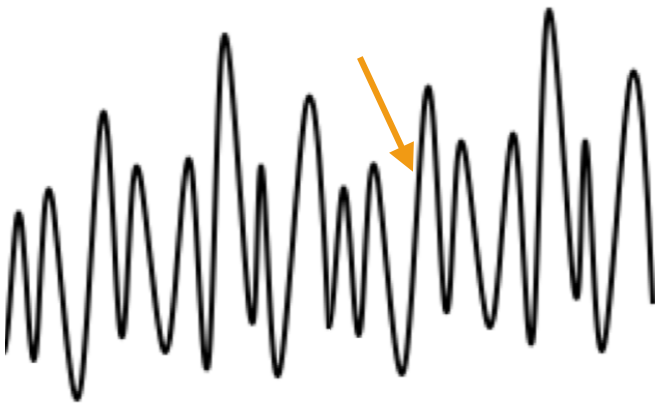
$$(\mathscr{C}_0, \mathscr{C}_1, \mathscr{C}_2, \ldots, \mathscr{C}_{255})$$

$$\begin{pmatrix} \mathsf{Sbox}(p_0 \oplus \mathbf{255}) \\ \mathsf{Sbox}(p_1 \oplus \mathbf{255}) \\ \mathsf{Sbox}(p_2 \oplus \mathbf{255}) \\ \cdots \\ \mathsf{Sbox}(p_{n-1} \oplus \mathbf{255}) \end{pmatrix}$$

$$\begin{pmatrix} \mathsf{HW}(\mathsf{Sbox}(p_0 \oplus \mathbf{255})) \\ \mathsf{HW}(\mathsf{Sbox}(p_1 \oplus \mathbf{255})) \\ \mathsf{HW}(\mathsf{Sbox}(p_2 \oplus \mathbf{255})) \\ \cdots \\ \mathsf{HW}(\mathsf{Sbox}(p_{n-1} \oplus \mathbf{255})) \end{pmatrix}$$

Example of DPA





$$\begin{pmatrix} v_0 = \mathcal{L}(p_0) \\ v_1 = \mathcal{L}(p_1) \\ v_2 = \mathcal{L}(p_2) \\ \cdots \\ v_{n-1} = \mathcal{L}(p_{n-1}) \end{pmatrix}$$

$$\text{argmax}\ (\mathscr{C}_0, \mathscr{C}_1, \mathscr{C}_2, \ldots, \mathscr{C}_{255})$$

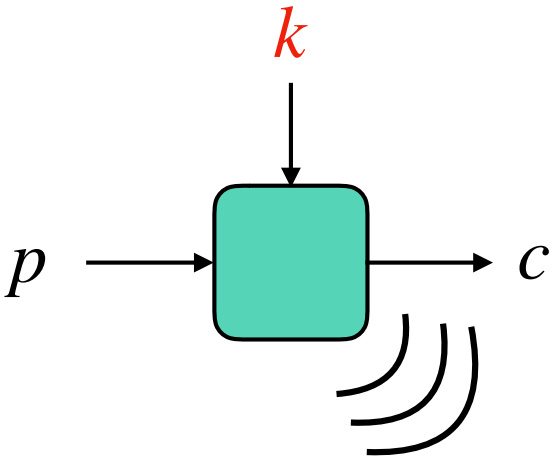
$$\begin{pmatrix} \text{Sbox}(p_0 \oplus \mathbf{255}) \\ \text{Sbox}(p_1 \oplus \mathbf{255}) \\ \text{Sbox}(p_2 \oplus \mathbf{255}) \\ \cdots \\ \text{Sbox}(p_{n-1} \oplus \mathbf{255}) \end{pmatrix} \qquad \begin{pmatrix} \text{HW}(\text{Sbox}(p_0 \oplus \mathbf{255})) \\ \text{HW}(\text{Sbox}(p_1 \oplus \mathbf{255})) \\ \text{HW}(\text{Sbox}(p_2 \oplus \mathbf{255})) \\ \cdots \\ \text{HW}(\text{Sbox}(p_{n-1} \oplus \mathbf{255})) \end{pmatrix}$$

Side-Channel Attacks

- Cheap equipment
  - Basic oscilloscope is enough
- Few traces
  - Less than a hundred traces to recover secrets in software
  - A few hundreds/thousands traces in hardware
- Fast
  - A few minutes to get the traces
  - A few seconds to mount the attack

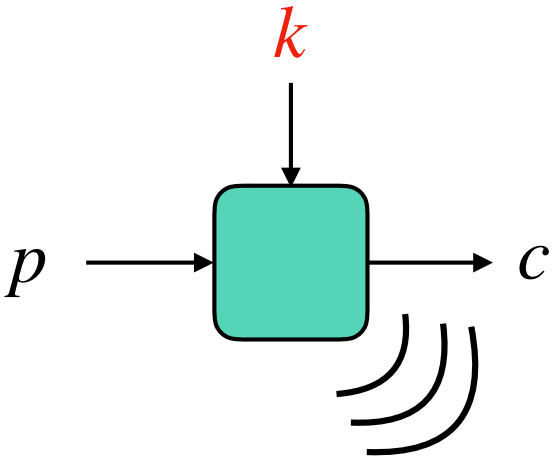
Countermeasures

How to thwart SCA?



Problem: the leakage is key-dependent

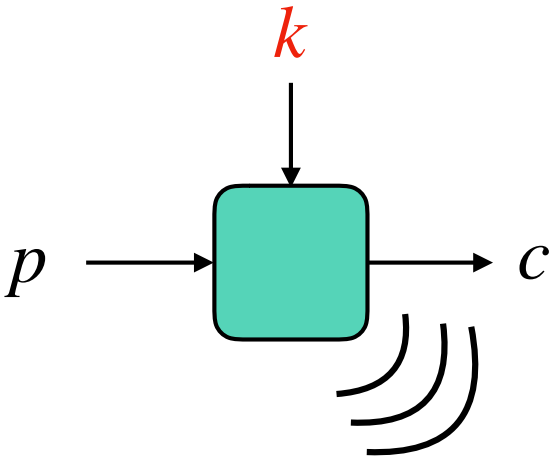
How to thwart SCA?



Problem: the leakage is key-dependent

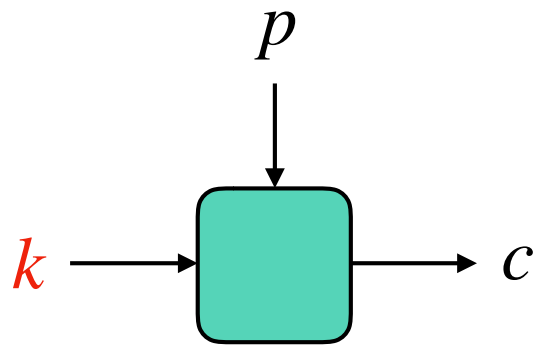
Solution 1: Fresh Re-Keying (regularly change the key)

How to thwart SCA?

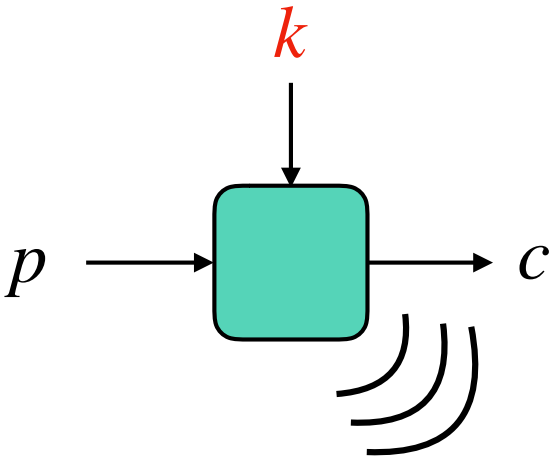


Problem: the leakage is key-dependent

Solution 1: Fresh Re-Keying (regularly change the key)

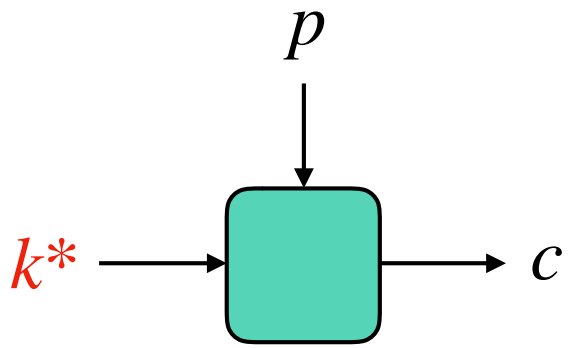


How to thwart SCA?

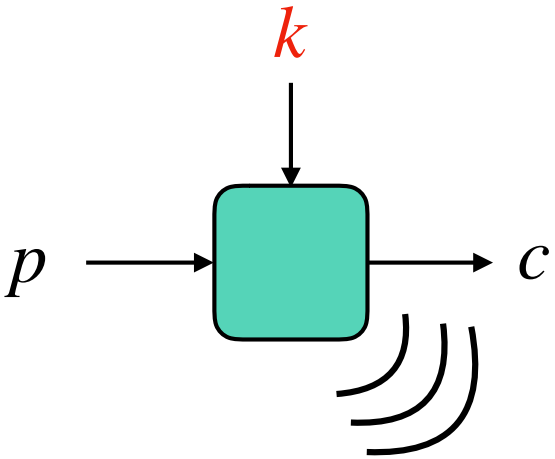


Problem: the leakage is key-dependent

Solution 1: Fresh Re-Keying (regularly change the key)

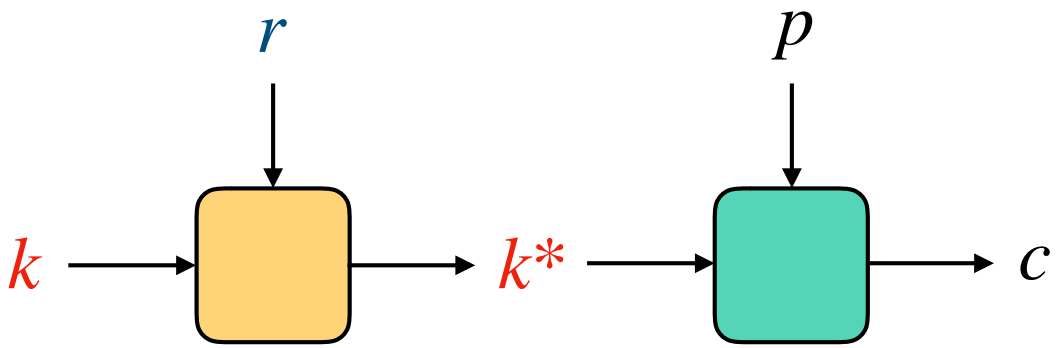


How to thwart SCA?

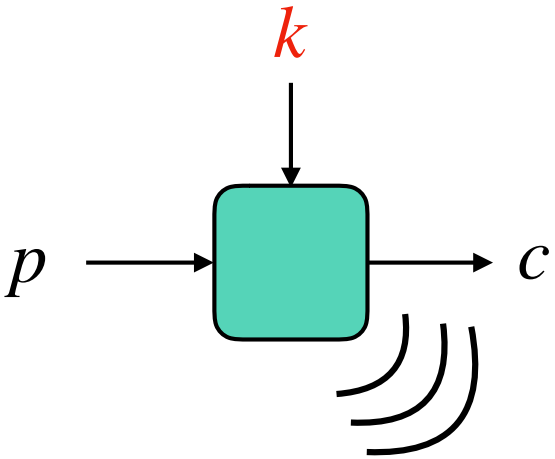


Problem: the leakage is key-dependent

Solution 1: Fresh Re-Keying (regularly change the key)



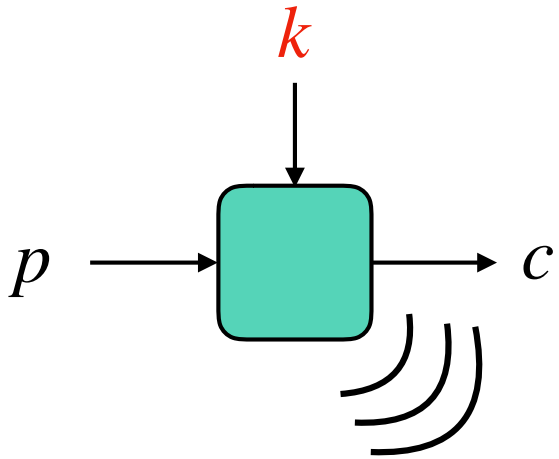
How to thwart SCA?



Problem: the leakage is key-dependent

Solution 2: Masking (make the leakage random)

How to thwart SCA?

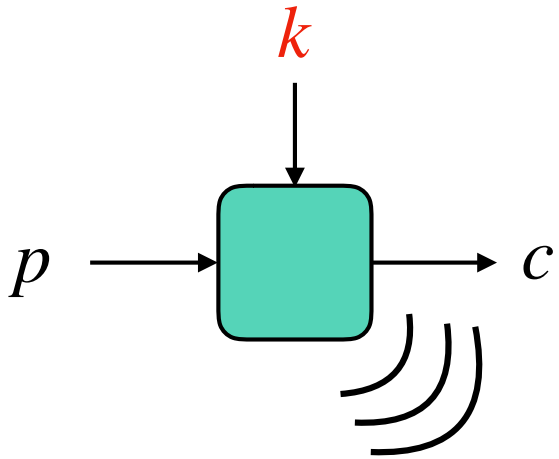


Problem: the leakage is key-dependent

Solution 2: Masking (make the leakage random)

for each sensitive value $v \leftarrow f(p, k)$

How to thwart SCA?



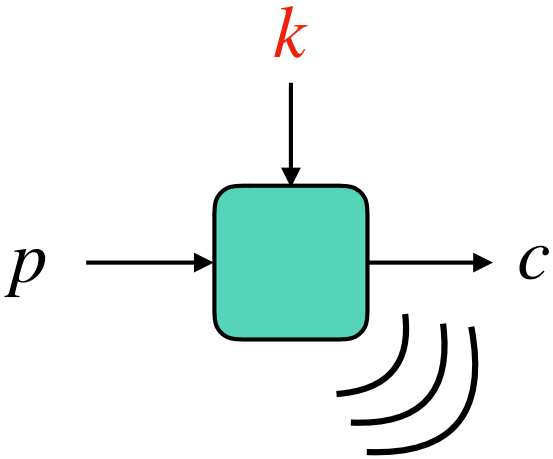
Problem: the leakage is key-dependent

Solution 2: Masking (make the leakage random)

for each sensitive value $v \leftarrow f(p, k)$

$$v_1 \leftarrow \$ \qquad v_2 \leftarrow \$ \qquad \cdots \qquad v_{n-1} \leftarrow \$$$

How to thwart SCA?



Problem: the leakage is key-dependent

Solution 2: Masking (make the leakage random)

for each sensitive value $v \leftarrow f(p, k)$

$$v_0 \leftarrow v \oplus \left(\bigoplus_{i=1}^{n-1} v_i \right) \qquad v_1 \leftarrow \$ \qquad v_2 \leftarrow \$ \qquad \cdots \qquad v_{n-1} \leftarrow \$$$

Masking in Practice

- Masking linear operations

$$z \leftarrow x \oplus y$$

$$x = x_0 \oplus x_1 \oplus \ldots \oplus x_{n-1}$$
$$y = y_0 \oplus y_1 \oplus \ldots \oplus y_{n-1}$$

Masking in Practice

- Masking linear operations

$$z \leftarrow x \oplus y$$

$$x = x_0 \oplus x_1 \oplus \ldots \oplus x_{n-1}$$
$$y = y_0 \oplus y_1 \oplus \ldots \oplus y_{n-1}$$

$$\mathbf{z} = (x_0 \oplus y_0, x_1 \oplus y_1, \ldots, x_{n-1} \oplus y_{n-1})$$

Masking in Practice

- Masking linear operations

$$z \leftarrow x \oplus y$$

$$x = x_0 \oplus x_1 \oplus \ldots \oplus x_{n-1}$$
$$y = y_0 \oplus y_1 \oplus \ldots \oplus y_{n-1}$$

$$\mathbf{z} = (x_0 \oplus y_0, x_1 \oplus y_1, \ldots, x_{n-1} \oplus y_{n-1})$$

- Masking non linear operations
  - Cannot be done share by share
  - Example of multiplication for $n = 2$

Masking in Practice

- Masking linear operations

$$z \leftarrow x \oplus y$$

$$x = x_0 \oplus x_1 \oplus \ldots \oplus x_{n-1}$$
$$y = y_0 \oplus y_1 \oplus \ldots \oplus y_{n-1}$$

$$\mathbf{z} = (x_0 \oplus y_0, x_1 \oplus y_1, \ldots, x_{n-1} \oplus y_{n-1})$$

- Masking non linear operations
  - Cannot be done share by share
  - Example of multiplication for $n = 2$

$$x = x_0 \oplus x_1$$
$$y = y_0 \oplus y_1$$

Masking in Practice

- Masking linear operations

$$z \leftarrow x \oplus y$$

$$x = x_0 \oplus x_1 \oplus \ldots \oplus x_{n-1}$$
$$y = y_0 \oplus y_1 \oplus \ldots \oplus y_{n-1}$$

$$\mathbf{z} = (x_0 \oplus y_0, x_1 \oplus y_1, \ldots, x_{n-1} \oplus y_{n-1})$$

- Masking non linear operations
  - Cannot be done share by share
  - Example of multiplication for $n = 2$

$$x = x_0 \oplus x_1$$
$$y = y_0 \oplus y_1$$

$$z_0 \leftarrow x_0 y_0 \oplus x_0 y_1$$
$$z_1 \leftarrow x_1 y_1 \oplus x_1 y_0$$

Masking in Practice

- Masking linear operations

$$z \leftarrow x \oplus y$$

$$x = x_0 \oplus x_1 \oplus \ldots \oplus x_{n-1}$$
$$y = y_0 \oplus y_1 \oplus \ldots \oplus y_{n-1}$$

$$\mathbf{z} = (x_0 \oplus y_0, x_1 \oplus y_1, \ldots, x_{n-1} \oplus y_{n-1})$$

- Masking non linear operations
  - Cannot be done share by share
  - Example of multiplication for $n = 2$

$$x = x_0 \oplus x_1$$
$$y = y_0 \oplus y_1$$

$$z_0 \leftarrow x_0 y_0 \oplus r \oplus x_0 y_1$$
$$z_1 \leftarrow x_1 y_1 \oplus r \oplus x_1 y_0$$

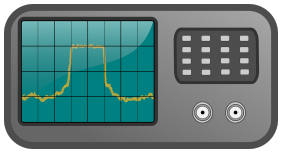
Leakage Models

Security of an implementation

- How to evaluate the security of an implementation?

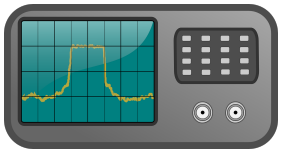
Security of an implementation

- How to evaluate the security of an implementation?
  - Integrate it on a device and try to attack it
    - Not always possible



Security of an implementation

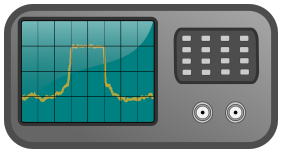
- How to evaluate the security of an implementation?
  - Integrate it on a device and try to attack it
    - Not always possible



  - Model the leakage and prove its security or exhibit an attack

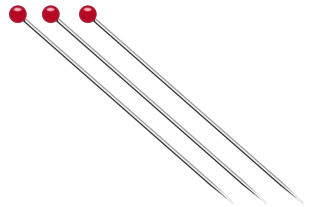
Security of an implementation

- How to evaluate the security of an implementation?
  - Integrate it on a device and try to attack it
    - Not always possible



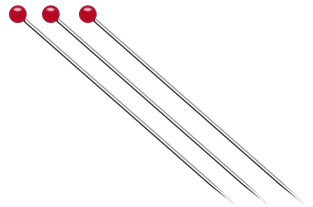
  - Model the leakage and prove its security or exhibit an attack

Probing Model



- Leakage
  - Only t variables leak in the implementation
  - Leakage = exact value

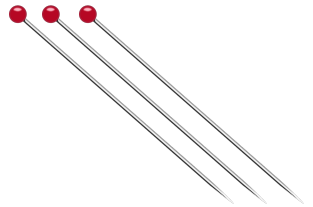
Probing Model



- Leakage
  - Only t variables leak in the implementation
  - Leakage = exact value

- Security in the t-probing model
  - Implementation such that any set of t intermediate variables is independent from the secret

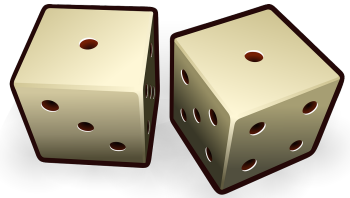
Probing Model



- Leakage
  - Only *t* variables leak in the implementation
  - Leakage = exact value
- Security in the *t*-probing model
  - Implementation such that any set of *t* intermediate variables is independent from the secret
- Pros and Cons
  - Easy to make security proofs
  - Not that close to the reality…

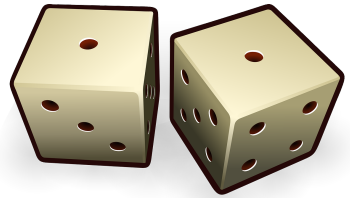
Random Probing Model

- Leakage
  - Every variable leaks with probability p
  - Leakage = exact value



Random Probing Model

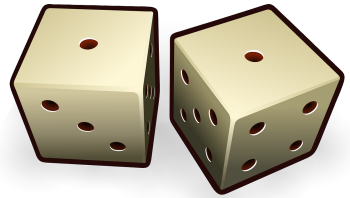
- Leakage
  - Every variable leaks with probability p
  - Leakage = exact value



- Security in the p-random probing model
  - Given p, the probability to recover information on the secret is negligible

Random Probing Model

- Leakage
  - Every variable leaks with probability p
  - Leakage = exact value

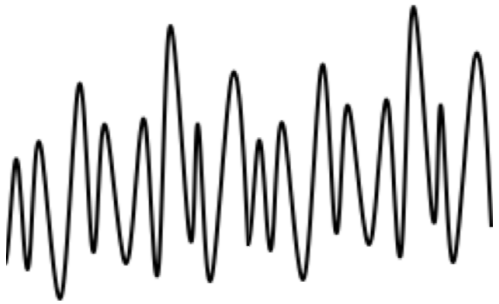


- Security in the p-random probing model
  - Given p, the probability to recover information on the secret is negligible

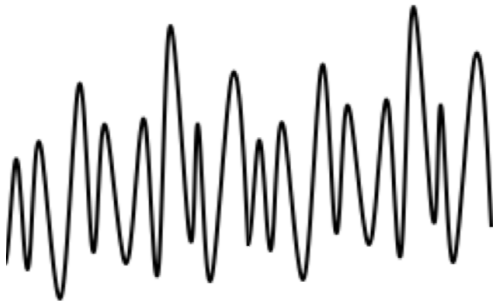
- Pros and Cons
  - A bit more complicated to make security proofs
  - Closer to the reality

Noisy Leakage Model

- Leakage
  - Every variable leaks
  - Leakage = noisy function of the value



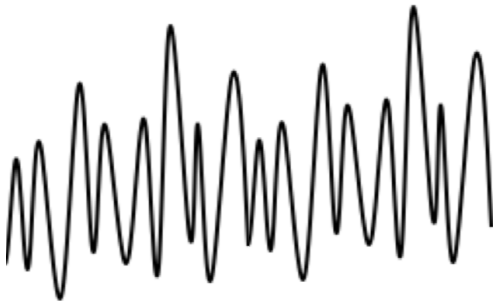
Noisy Leakage Model



- Leakage
  - Every variable leaks
  - Leakage = noisy function of the value

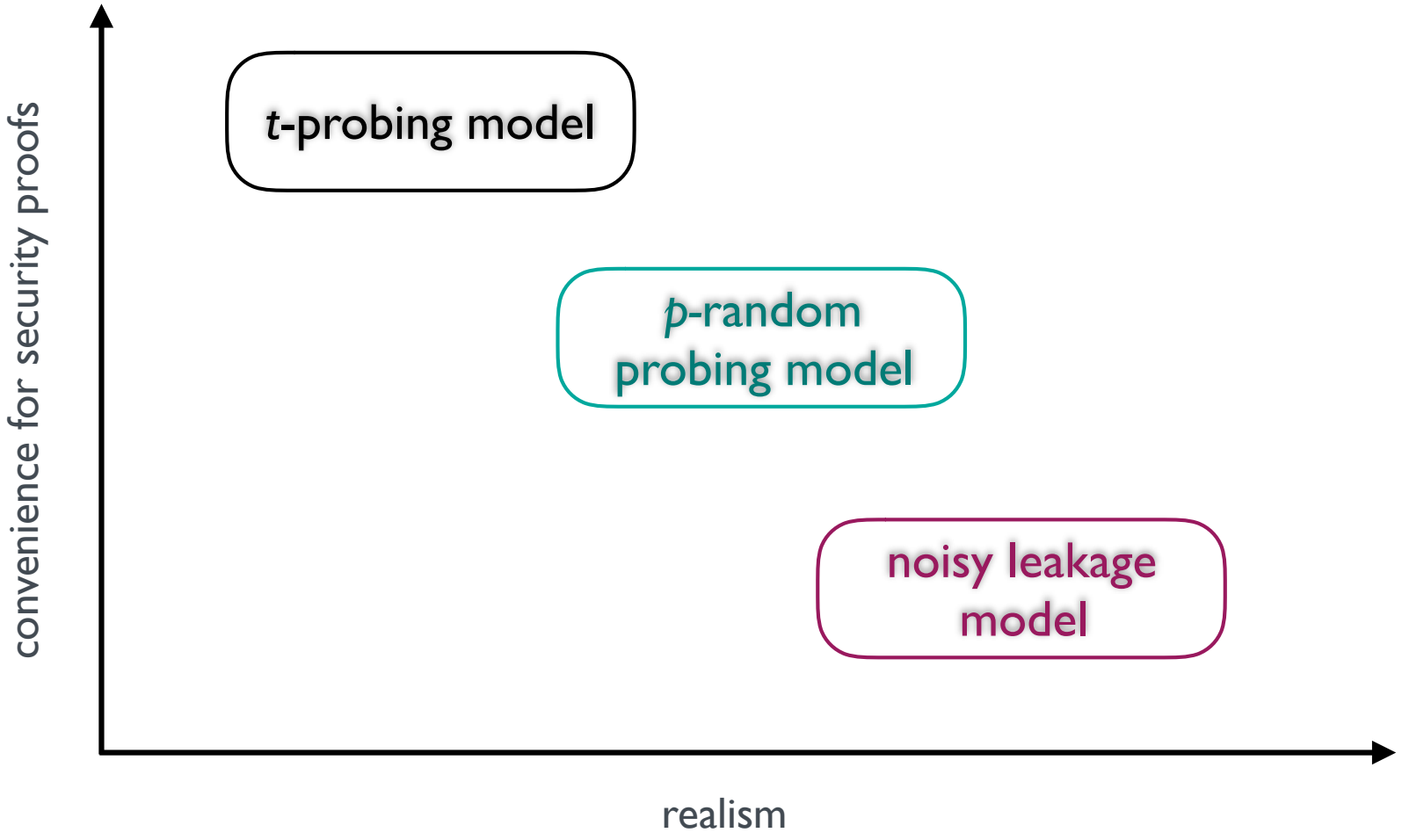
- Security in the noisy leakage model
  - Given the level of noise, the probability to recover information on the secret is negligible

Noisy Leakage Model

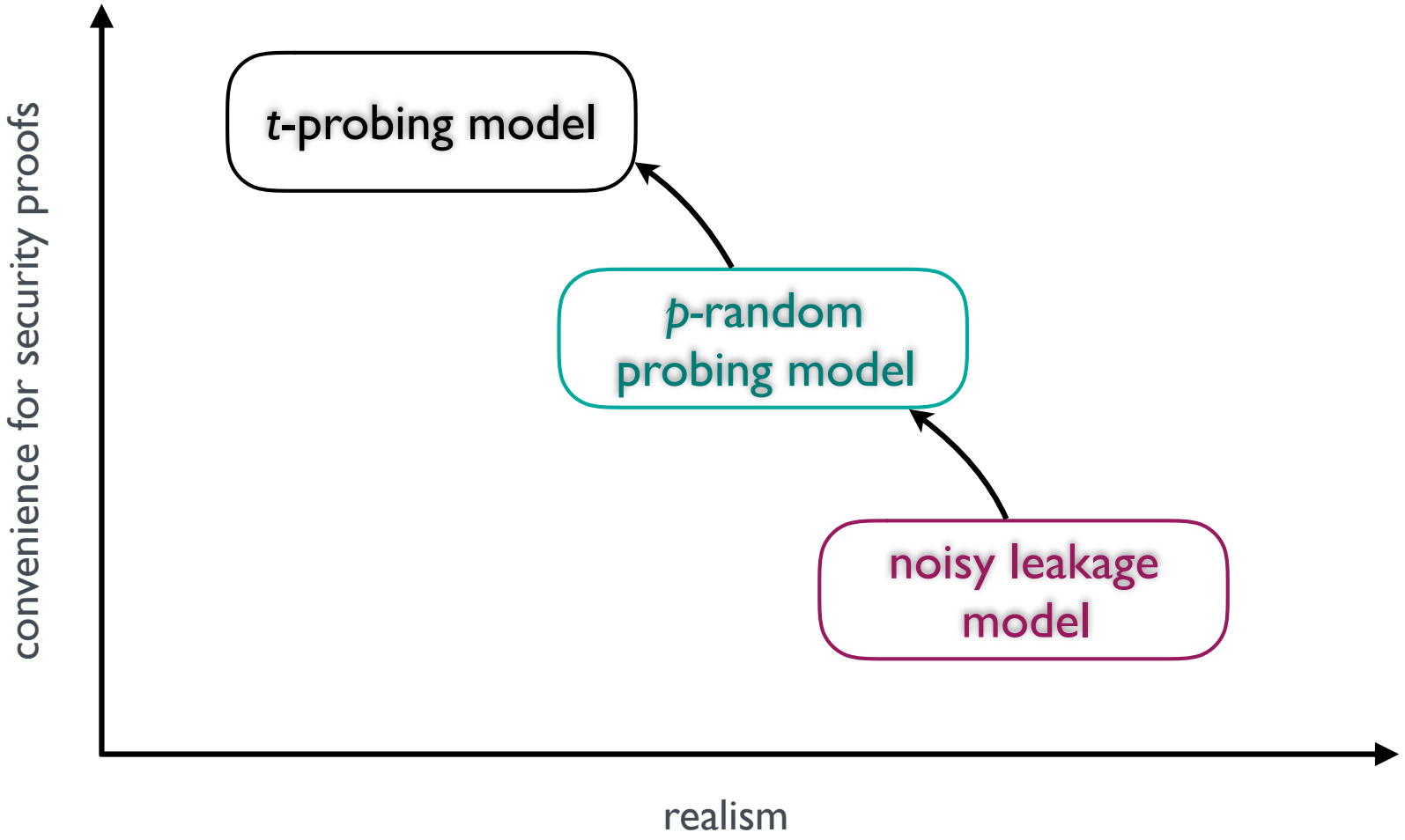


- Leakage
  - Every variable leaks
  - Leakage = noisy function of the value
- Security in the noisy leakage model
  - Given the level of noise, the probability to recover information on the secret is negligible
- Pros and Cons
  - Much more complicated to make security proofs
  - The closest to the reality

Reductions


convenience for security proofs
t-probing model
p-random probing model
noisy leakage model
realism

Reductions


t-probing model
p-random probing model
noisy leakage model
convenience for security proofs
realism

Security Proofs

Proof in the Probing Model

- Reminder: an implementation is *t*-probing secure iff any set of at most *t* variables is independent from the secret

2 shares

function example(a_0, a_1, b_0, b_1)

$r \leftarrow \$$

$u \leftarrow a_0 \cdot b_0$

$c_0 \leftarrow u \oplus r$

$v \leftarrow a_1 \cdot b_1$

$x \leftarrow a_0 \cdot b_1$

$w \leftarrow v \oplus x$

$y \leftarrow w \oplus r$

$z \leftarrow a_1 \cdot b_0$

$c_1 \leftarrow y \oplus z$

return (c_0, c_1)

Proof in the Probing Model

- Reminder: an implementation is *t*-probing secure iff any set of at most *t* variables is independent from the secret

2 shares

1-probing secure?

function example(a_0, a_1, b_0, b_1)

$r \leftarrow \$$

$u \leftarrow a_0 \cdot b_0$

$c_0 \leftarrow u \oplus r$

$v \leftarrow a_1 \cdot b_1$

$x \leftarrow a_0 \cdot b_1$

$w \leftarrow v \oplus x$

$y \leftarrow w \oplus r$

$z \leftarrow a_1 \cdot b_0$

$c_1 \leftarrow y \oplus z$

return (c_0, c_1)

Proof in the Probing Model

- Reminder: an implementation is *t*-probing secure iff any set of at most *t* variables is independent from the secret

Independent from secrets?

function example(a_0, a_1, b_0, b_1)

$r \leftarrow \$$

$u \leftarrow a_0 \cdot b_0$

$c_0 \leftarrow u \oplus r$

$v \leftarrow a_1 \cdot b_1$

$x \leftarrow a_0 \cdot b_1$

$w \leftarrow v \oplus x$

$y \leftarrow w \oplus r$

$z \leftarrow a_1 \cdot b_0$

$c_1 \leftarrow y \oplus z$

return (c_0, c_1)

Proof in the Probing Model

- Reminder: an implementation is *t*-probing secure iff any set of at most *t* variables is independent from the secret

Independent from secrets?

function example(a_0, a_1, b_0, b_1)

$r \leftarrow \$$

$u \leftarrow a_0 \cdot b_0$

$c_0 \leftarrow u \oplus r$

$v \leftarrow a_1 \cdot b_1$

$x \leftarrow a_0 \cdot b_1$

$w \leftarrow v \oplus x$

$y \leftarrow w \oplus r$

$z \leftarrow a_1 \cdot b_0$

$c_1 \leftarrow y \oplus z$

return (c_0, c_1)

Proof in the Probing Model

- Reminder: an implementation is *t*-probing secure iff any set of at most *t* variables is independent from the secret

Independent from secrets?

function example(a_0, a_1, b_0, b_1)

$r \leftarrow \$$

$u \leftarrow a_0 \cdot b_0$

$c_0 \leftarrow u \oplus r$

$v \leftarrow a_1 \cdot b_1$

$x \leftarrow a_0 \cdot b_1$

$w \leftarrow v \oplus x$

$y \leftarrow w \oplus r$

$z \leftarrow a_1 \cdot b_0$

$c_1 \leftarrow y \oplus z$

return (c_0, c_1)

Proof in the Probing Model

- Reminder: an implementation is *t*-probing secure iff any set of at most *t* variables is independent from the secret

Independent from secrets?

function example(a_0, a_1, b_0, b_1)

$r \leftarrow \$$

$u \leftarrow a_0 \cdot b_0$

$c_0 \leftarrow u \oplus r$

$v \leftarrow a_1 \cdot b_1$

$x \leftarrow a_0 \cdot b_1$

$w \leftarrow v \oplus x$

$y \leftarrow w \oplus r$

$z \leftarrow a_1 \cdot b_0$

$c_1 \leftarrow y \oplus z$

return (c_0, c_1)

Proof in the Probing Model

- Reminder: an implementation is *t*-probing secure iff any set of at most *t* variables is independent from the secret

Independent from secrets?

function example(a_0, a_1, b_0, b_1)

$r \leftarrow \$$

$u \leftarrow a_0 \cdot b_0$

$c_0 \leftarrow u \oplus r$

$v \leftarrow a_1 \cdot b_1$

$x \leftarrow a_0 \cdot b_1$

$w \leftarrow v \oplus x$

$y \leftarrow w \oplus r$

$z \leftarrow a_1 \cdot b_0$

$c_1 \leftarrow y \oplus z$

return (c_0, c_1)

Proof in the Probing Model

- Reminder: an implementation is *t*-probing secure iff any set of at most *t* variables is independent from the secret

Independent from secrets?

function example(a_0, a_1, b_0, b_1)

$r \leftarrow \$$

$u \leftarrow a_0 \cdot b_0$

$c_0 \leftarrow u \oplus r$

$v \leftarrow a_1 \cdot b_1$

$x \leftarrow a_0 \cdot b_1$

$w \leftarrow v \oplus x$

$y \leftarrow w \oplus r$

$z \leftarrow a_1 \cdot b_0$

$c_1 \leftarrow y \oplus z$

return (c_0, c_1)

Proof in the Probing Model

- Reminder: an implementation is *t*-probing secure iff any set of at most *t* variables is independent from the secret

Independent from secrets?

$$w = v \oplus x$$
$$w = a_1 \cdot b_1 \oplus a_0 \cdot b_1$$
$$w = a \cdot b_1$$

function example(a_0, a_1, b_0, b_1)

$r \leftarrow \$$

$u \leftarrow a_0 \cdot b_0$

$c_0 \leftarrow u \oplus r$

$v \leftarrow a_1 \cdot b_1$

$x \leftarrow a_0 \cdot b_1$

$w \leftarrow v \oplus x$

$y \leftarrow w \oplus r$

$z \leftarrow a_1 \cdot b_0$

$c_1 \leftarrow y \oplus z$

return (c_0, c_1)

Proof in the Probing Model

- Reminder: an implementation is *t*-probing secure iff any set of at most *t* variables is independent from the secret

3 shares

function example$(a_0, a_1, a_2, b_0, b_1, b_2)$

$r_{00}, r_{01}, r_{02}, r_{12} \leftarrow \$$
$t \leftarrow a_0 \cdot b_0$
$c_0 \leftarrow t \oplus r_{00}$
$t \leftarrow a_0 \cdot b_1$
$t \leftarrow t \oplus r_{01}$
$c_0 \leftarrow c_0 \oplus t$
$t \leftarrow a_0 \cdot b_2$
$t \leftarrow t \oplus r_{02}$
$c_0 \leftarrow c_0 \oplus t$
$t \leftarrow a_1 \cdot b_0$
$c_1 \leftarrow t \oplus r_{01}$
$t \leftarrow a_1 \cdot b_1$
$c_1 \leftarrow c_1 \oplus t$
$\ldots$

return (c_0, c_1, c_2)

Proof in the Probing Model

- Reminder: an implementation is *t*-probing secure iff any set of at most *t* variables is independent from the secret

3 shares

33 intermediate variables

$\binom{33}{2} = 528$ couples to verify

function example$(a_0, a_1, a_2, b_0, b_1, b_2)$

$r_{00}, r_{01}, r_{02}, r_{12} \leftarrow \$$
$t \leftarrow a_0 \cdot b_0$
$c_0 \leftarrow t \oplus r_{00}$
$t \leftarrow a_0 \cdot b_1$
$t \leftarrow t \oplus r_{01}$
$c_0 \leftarrow c_0 \oplus t$
$t \leftarrow a_0 \cdot b_2$
$t \leftarrow t \oplus r_{02}$
$c_0 \leftarrow c_0 \oplus t$
$t \leftarrow a_1 \cdot b_0$
$c_1 \leftarrow t \oplus r_{01}$
$t \leftarrow a_1 \cdot b_1$
$c_1 \leftarrow c_1 \oplus t$
$\ldots$
return (c_0, c_1, c_2)

Proof in the Probing Model

- Reminder: an implementation is *t*-probing secure iff any set of at most *t* variables is independent from the secret

3 shares

33 intermediate variables

$\binom{33}{2} = 528$ couples to verify

$\binom{n}{t}$ tuples to verify

function example$(a_0, a_1, a_2, b_0, b_1, b_2)$

$r_{00}, r_{01}, r_{02}, r_{12} \leftarrow \$$
$t \leftarrow a_0 \cdot b_0$
$c_0 \leftarrow t \oplus r_{00}$
$t \leftarrow a_0 \cdot b_1$
$t \leftarrow t \oplus r_{01}$
$c_0 \leftarrow c_0 \oplus t$
$t \leftarrow a_0 \cdot b_2$
$t \leftarrow t \oplus r_{02}$
$c_0 \leftarrow c_0 \oplus t$
$t \leftarrow a_1 \cdot b_0$
$c_1 \leftarrow t \oplus r_{01}$
$t \leftarrow a_1 \cdot b_1$
$c_1 \leftarrow c_1 \oplus t$
$\dots$
return (c_0, c_1, c_2)

Proof in the Probing Model

- Reminder: an implementation is *t*-probing secure iff any set of at most *t* variables is independent from the secret

- Two methods to verify *t*-probing security
  - Theoretical proof from the structure of the algorithm
  - Automatic proofs with a tool

Proof in the Probing Model

- Reminder: an implementation is *t*-probing secure iff any set of at most *t* variables is independent from the secret

- Two methods to verify *t*-probing security
  - Theoretical proof from the structure of the algorithm
  - Automatic proofs with a tool

Example of Automatic Tools

maskVerif

Example of Automatic Tools

Security order t

function example$(a_0, a_1, a_2, b_0, b_1, b_2)$

$r_{00}, r_{01}, r_{02}, r_{12} \leftarrow \$$
$t \leftarrow a_0 \cdot b_0$
$c_0 \leftarrow t \oplus r_{00}$
$t \leftarrow a_0 \cdot b_1$
$t \leftarrow t \oplus r_{01}$
$c_0 \leftarrow c_0 \oplus t$
$t \leftarrow a_0 \cdot b_2$
$t \leftarrow t \oplus r_{02}$
$c_0 \leftarrow c_0 \oplus t$
$t \leftarrow a_1 \cdot b_0$
$c_1 \leftarrow t \oplus r_{01}$
$t \leftarrow a_1 \cdot b_1$
$c_1 \leftarrow c_1 \oplus t$
...
return (c_0, c_1, c_2)

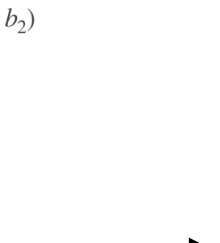
maskVerif

Example of Automatic Tools

Security order t

function example$(a_0, a_1, a_2, b_0, b_1, b_2)$

$r_{00}, r_{01}, r_{02}, r_{12} \leftarrow \$$
$t \leftarrow a_0 \cdot b_0$
$c_0 \leftarrow t \oplus r_{00}$
$t \leftarrow a_0 \cdot b_1$
$t \leftarrow t \oplus r_{01}$
$c_0 \leftarrow c_0 \oplus t$
$t \leftarrow a_0 \cdot b_2$
$t \leftarrow t \oplus r_{02}$
$c_0 \leftarrow c_0 \oplus t$
$t \leftarrow a_1 \cdot b_0$
$c_1 \leftarrow t \oplus r_{01}$
$t \leftarrow a_1 \cdot b_1$
$c_1 \leftarrow c_1 \oplus t$
...
return (c_0, c_1, c_2)



maskVerif

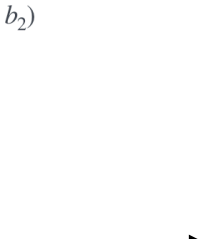
Security proof or potential attacks

Example of Automatic Tools

Security order t

function example$(a_0, a_1, a_2, b_0, b_1, b_2)$

$r_{00}, r_{01}, r_{02}, r_{12} \leftarrow \$$
$t \leftarrow a_0 \cdot b_0$
$c_0 \leftarrow t \oplus r_{00}$
$t \leftarrow a_0 \cdot b_1$
$t \leftarrow t \oplus r_{01}$
$c_0 \leftarrow c_0 \oplus t$
$t \leftarrow a_0 \cdot b_2$
$t \leftarrow t \oplus r_{02}$
$c_0 \leftarrow c_0 \oplus t$
$t \leftarrow a_1 \cdot b_0$
$c_1 \leftarrow t \oplus r_{01}$
$t \leftarrow a_1 \cdot b_1$
$c_1 \leftarrow c_1 \oplus t$
...
return (c_0, c_1, c_2)



maskVerif

Security proof or potential attacks

Rules?

Conclusion

Summary

- Side-channel attacks are very powerful
  - Few seconds to recover the key on some software devices
  - Cheap equipments

Summary

- Side-channel attacks are very powerful
  - Few seconds to recover the key on some software devices
  - Cheap equipments
- Countermeasures are mandatory for sensitive devices
  - Hardware and low cost countermeasures
  - Fresh re-keying
  - Masking

Summary

- Side-channel attacks are very powerful
  - Few seconds to recover the key on some software devices
  - Cheap equipments
- Countermeasures are mandatory for sensitive devices
  - Hardware and low cost countermeasures
  - Fresh re-keying
  - Masking
- Practical security
  - Security proofs in relevant leakage models
  - Automatic tools

Challenges

- Efficiency
  - The least possible randomness
  - The least possible operations

Challenges

- Efficiency
  - The least possible randomness
  - The least possible operations
- Security
  - Theoretical proofs of existing schemes
  - Automatic tools to verify the security of implementations

Challenges

- Efficiency
  - The least possible randomness
  - The least possible operations
- Security
  - Theoretical proofs of existing schemes
  - Automatic tools to verify the security of implementations
- Practicality
  - Security of implementations under leakage models as close as possible to the reality

Thank you

Subsidiary Question

Is probing secure ?