Simple Power Analysis (SPA)

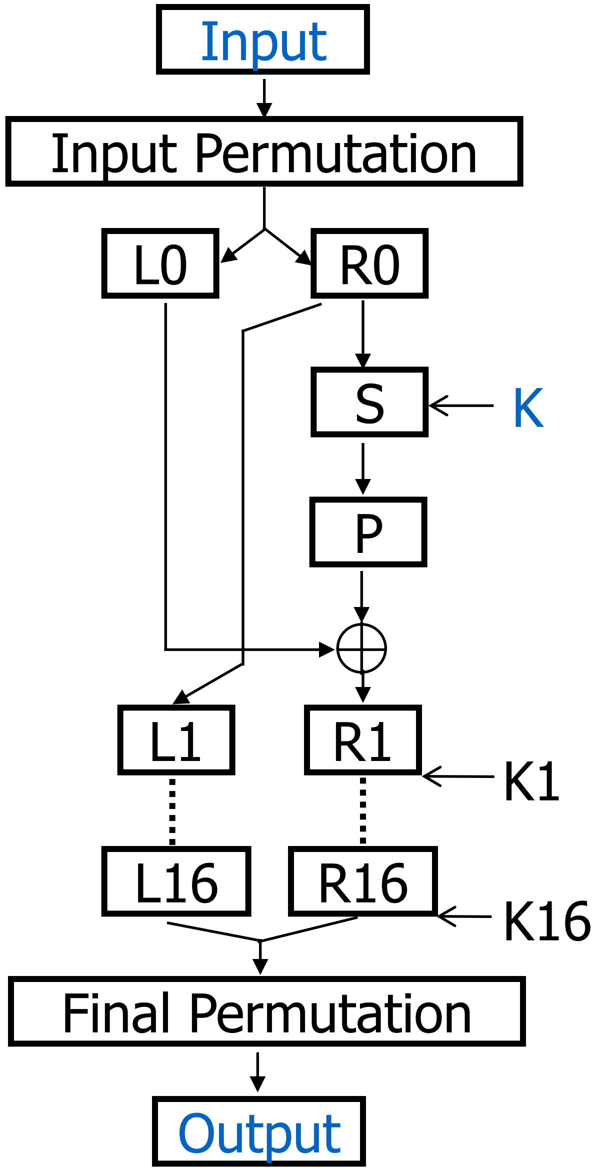
- Originally proposed by Paul Kocher, 1996
- Monitor the device’s power consumption to deduce information about data and operation
- Example: SPA on DES – smart cards
  - The internal structure is shown on the next slide
- Summary of DES – a block cipher
  - a product cipher
  - 16 rounds iterations
    - substitutions (for confusion)
    - permutations (for diffusion)
  - Each round has a *round key*
    - Generated from the user-supplied key

Known

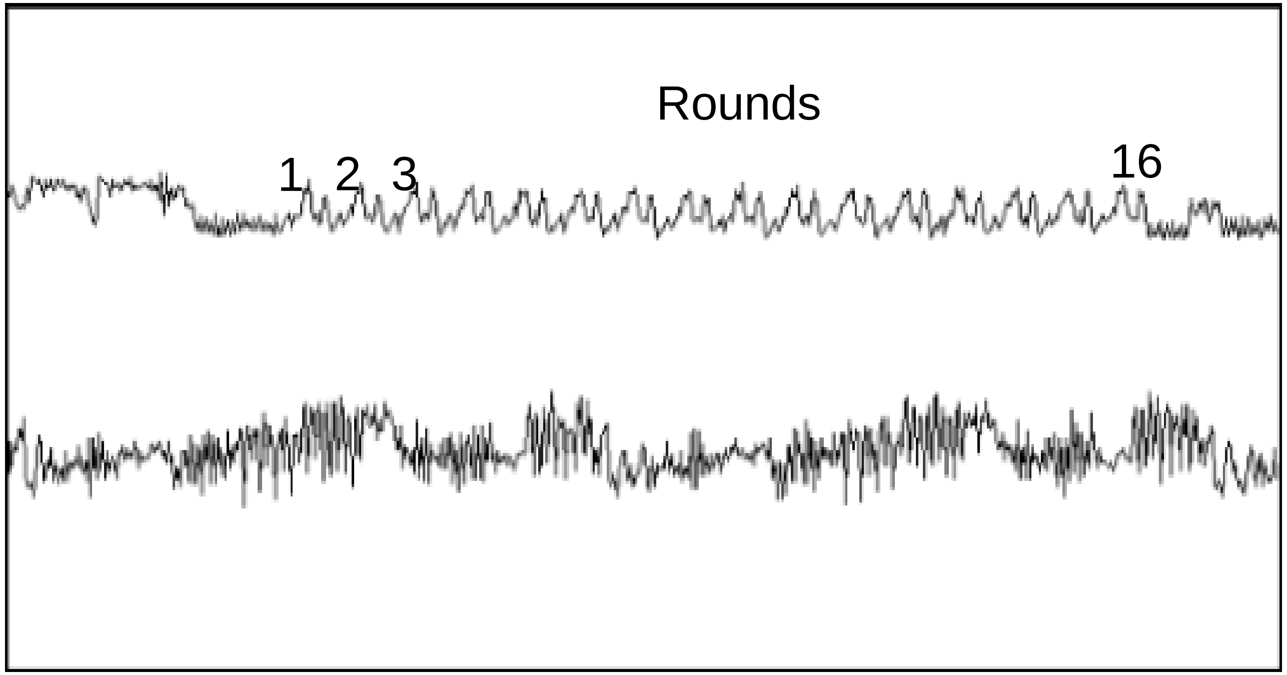
DES Basic Structure

- Input: 64 bits (a block)
- Li/Ri– left/right half (32 bits) of the input block for iteration i– subject to substitution S and permutation P
- K - user-supplied key
- Ki - round key:
  - 56 bits used +8 unused

    (unused for encryption but often used for error checking)
- Output: 64 bits (a block)
- Note: Ri becomes L(i+1)
- All basic op's are simple logical ops
  - Left shift / XOR



SPA on DES (cont'd)



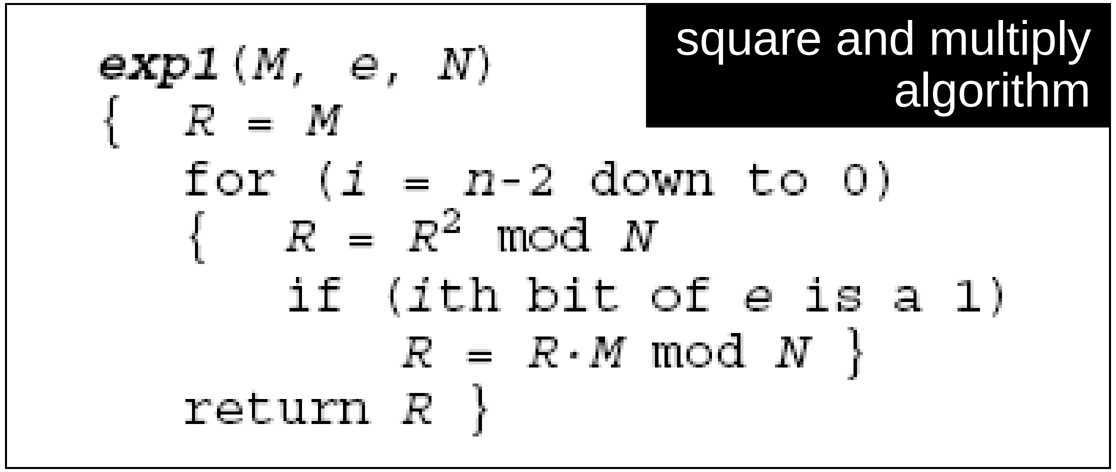
- The upper trace – entire encryption, including the initial phase, 16 DES rounds, and the final permutation
- The lower trace – detailed view of the second and third rounds
- **The power trace can reveal the instruction sequence**

SPA

- SPA can be used to break cryptographic implementations (execution path, instruction, key change, etc.)
  - **DES key schedule:** Involves rotating 28-bit key registers
  - **DES permutation:** involves conditional branching
  - The DES structure and 16 rounds are known
  - Instruction flow depends on data → power signature
  - **Comparison:** Involves string and memory comparison operations performing a conditional branch when a mismatch is found
- SPA Countermeasure:
  - Avoid procedures that use secret intermediates or keys for conditional branching operation

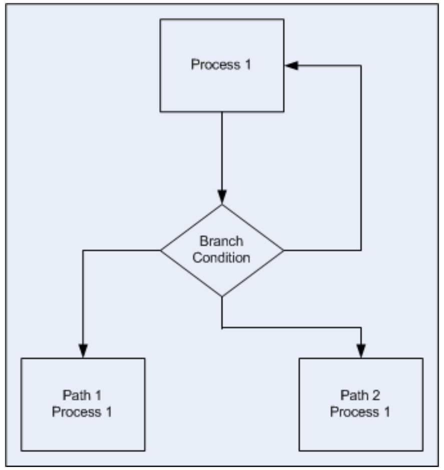
SPA for other encryption techniques

- AES is another private encryption technique that includes a data mixing step.
- RSA is a public key encryption technique that involves modulo exponents.
- Example: Modular exponentiation in DES is often implemented by square and multiply algorithm
- Then, the power trace of the exponentiation can directly yields the corresponding value
- All programs involving conditional branching based on the key values are at risk!



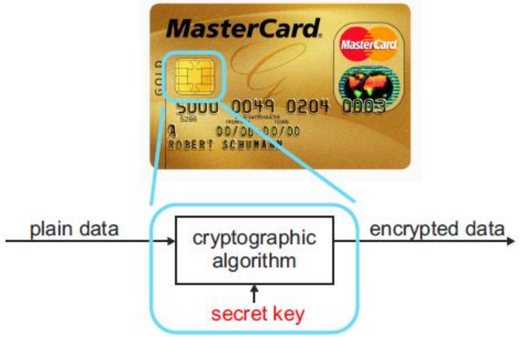
square and multiply algorithm

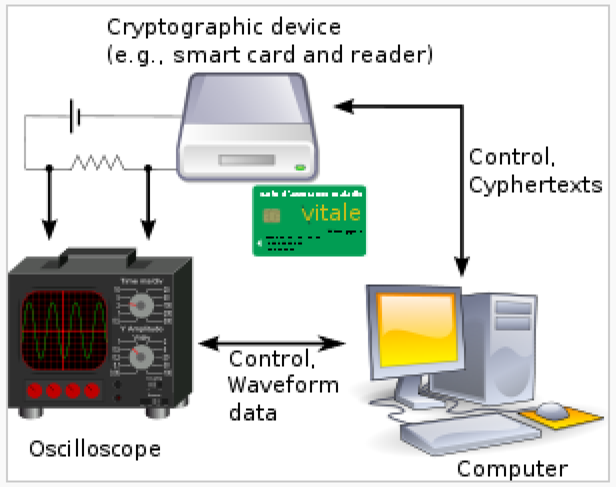
```
exp1($M$, $e$, $N$)
{   $R = M$
    for ($i = n-2$ down to 0)
    {   $R = R^2 \bmod N$
        if (ith bit of $e$ is a 1)
            $R = R \cdot M \bmod N$ }
    return $R$ }
```



Differential Power Analysis (DPA)

- SPA targets variable instruction flow
- DPA targets data-dependence
  - Different operands present different power
- Difference between smart cards and FPGAs
  - In smart cards, one operation running at a time
    - → Simple power tracing is possible
  - In FPGAs, typically parallel computations prevent visual SPA inspection → DPA

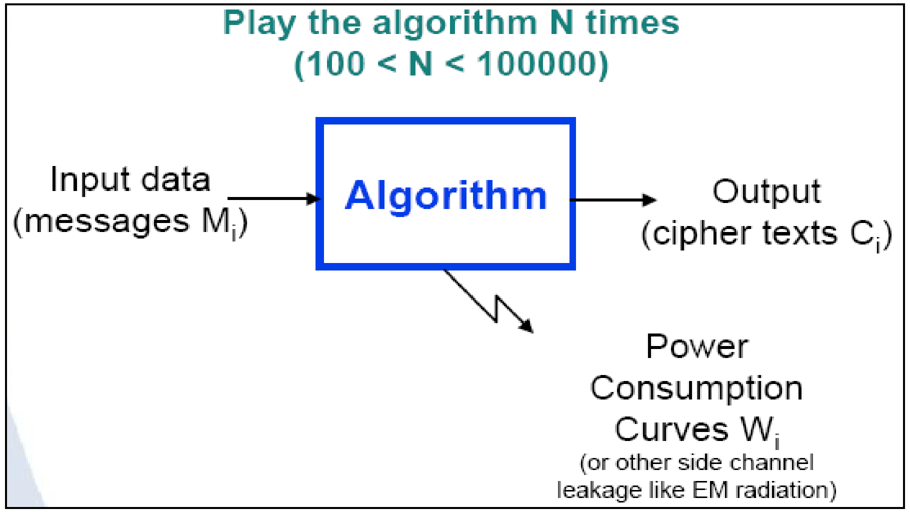




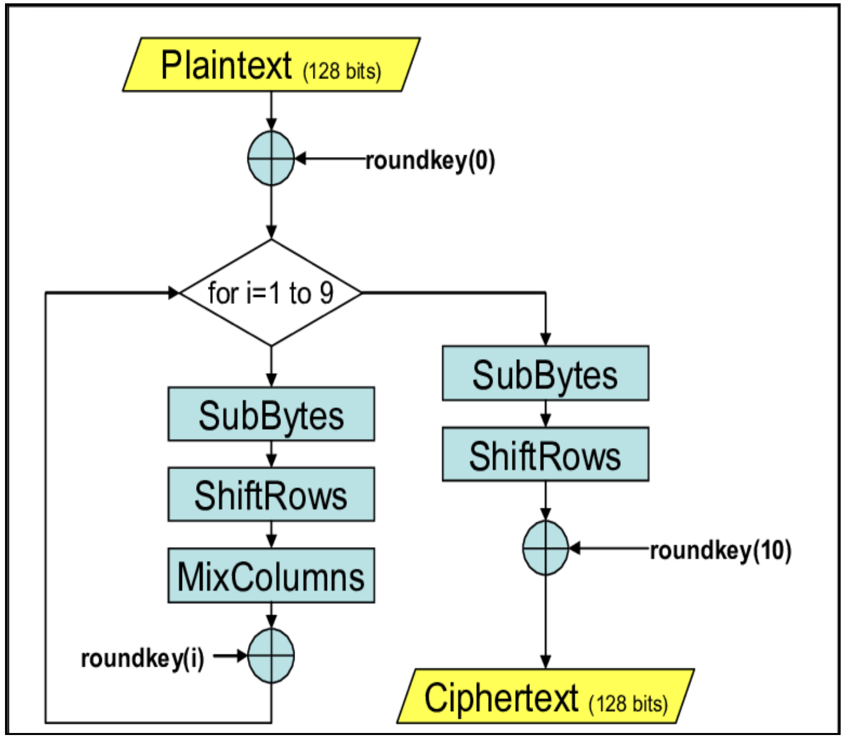
A diagram of differential power analysis.

DPA

- DPA can be performed on any algorithm that has the operation $\beta = S(\alpha \oplus K)$,
  - α is known and K is the segment key

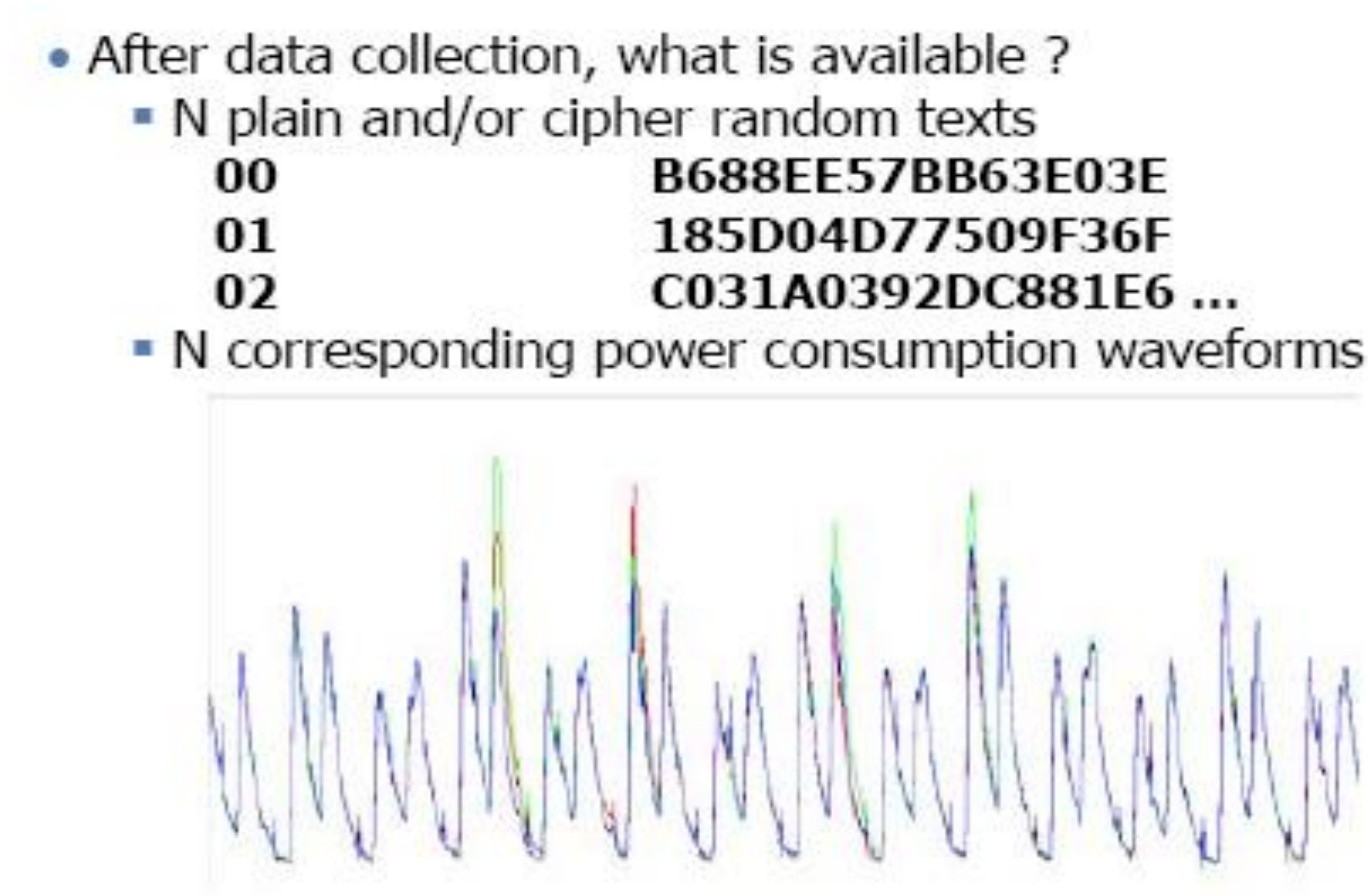


The waveforms are captured by a scope and sent to a computer for analysis



Assumption: Either Plaintext or Cipher is known

What is available after acquisition?

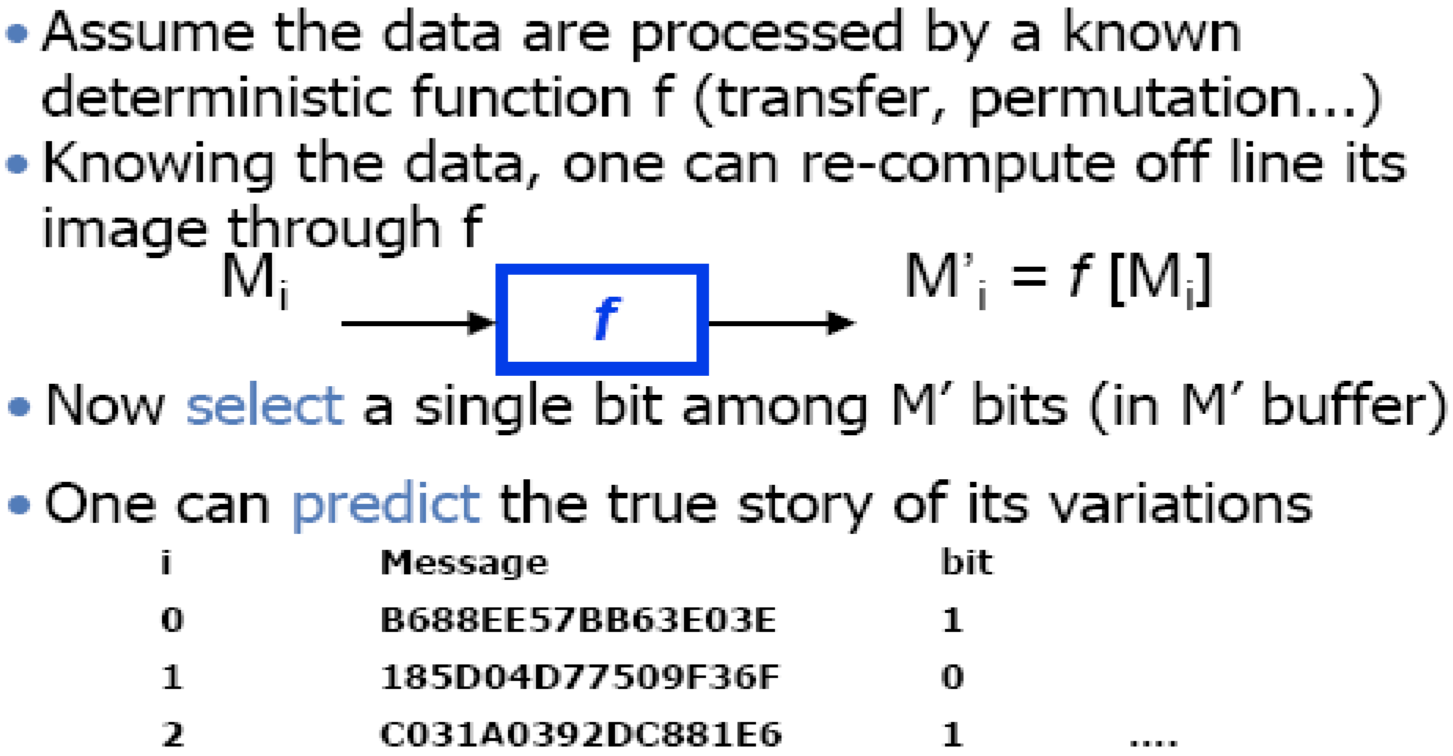


- After data collection, what is available ?
  - N plain and/or cipher random texts

00	B688EE57BB63E03E
01	185D04D77509F36F
02	C031A0392DC881E6 ...

  - N corresponding power consumption waveforms

DPA (cont'd)



- Assume the data are processed by a known deterministic function f (transfer, permutation…)
- Knowing the data, one can re-compute off line its image through f

$$M_i \longrightarrow \boxed{f} \longrightarrow M'_i = f\,[M_i]$$

- Now select a single bit among M' bits (in M' buffer)
- One can predict the true story of its variations

i	Message	bit	
0	B688EE57BB63E03E	1	
1	185D04D77509F36F	0	
2	C031A0392DC881E6	1	….

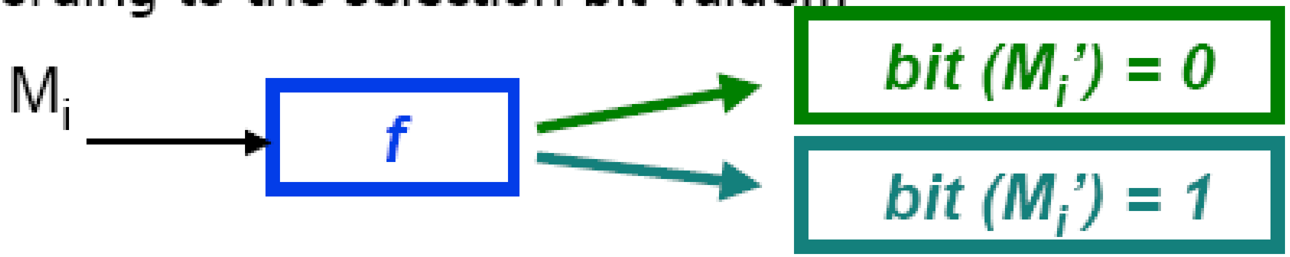
The bit will classify the wave w_i

- Hypothesis 1: bit is zero
- Hypothesis 2: bit is one
- A differential trace will be calculated for each bit!

Assumption: Attacker knows the algorithm well

DPA (cont’d)

- Partition the data and related curves into two packs, according to the selection bit value...



0	B688EE57BB63E03E	1
1	185D04D77509F36F	0
2	C031A0392DC881E6	1

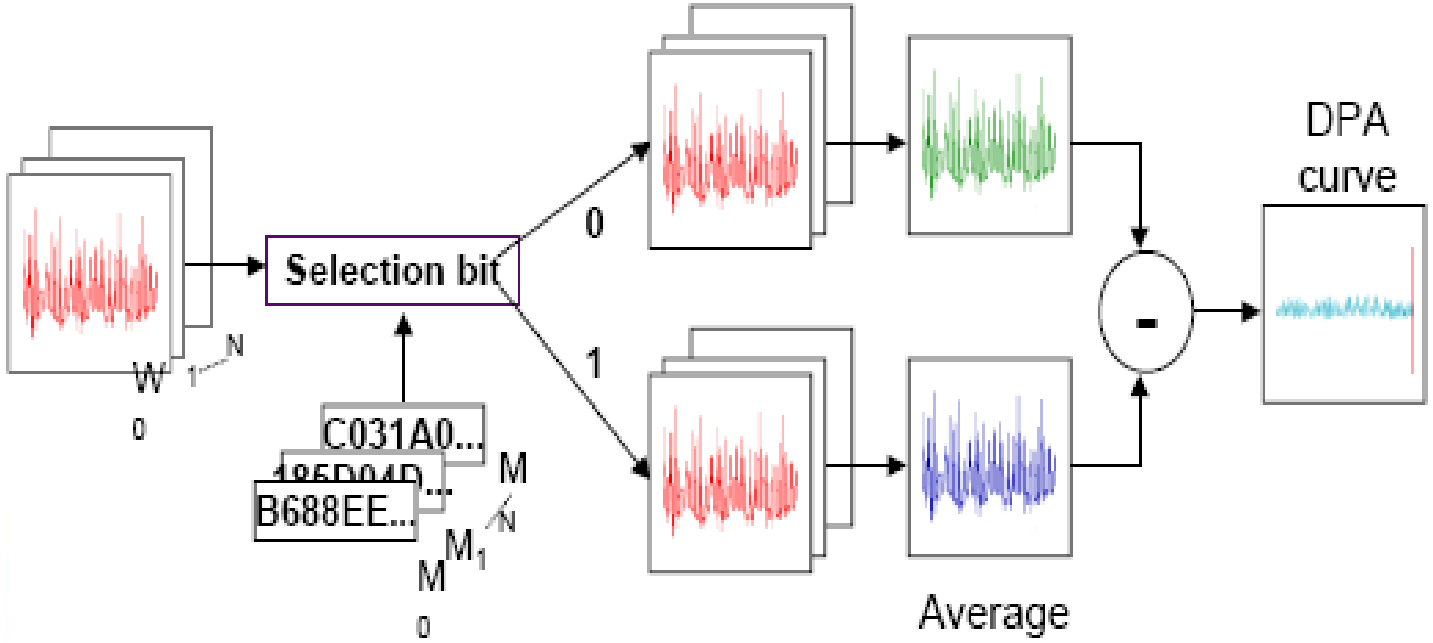
...

- Sum the signed consumption curves and normalise
- <=> Difference of averages

$(N_0 + N_1 = N)$

$$\boxed{DPA = \frac{\sum W_1}{N_1} - \frac{\sum W_0}{N_0}}$$

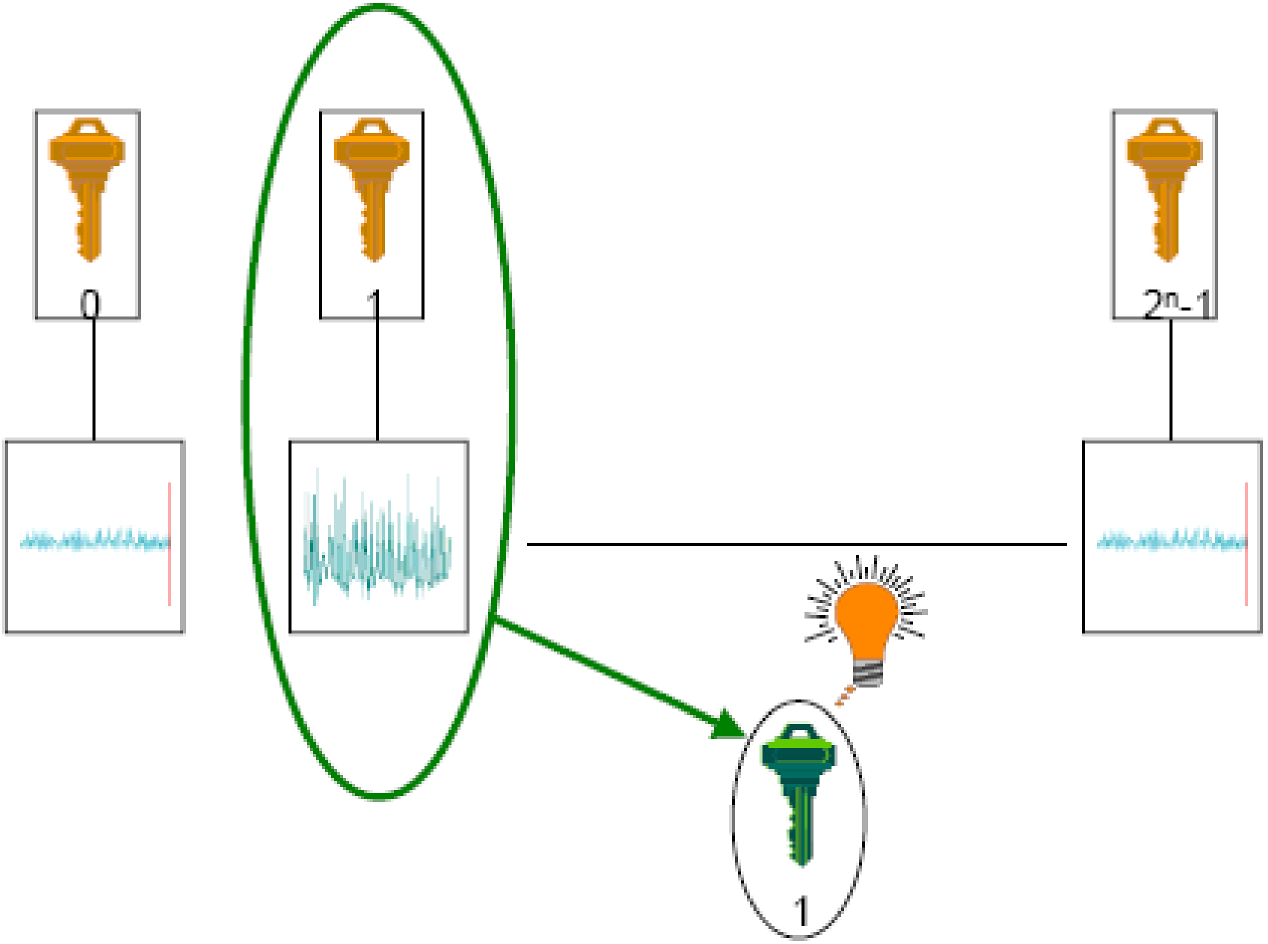
DPA (cont’d)



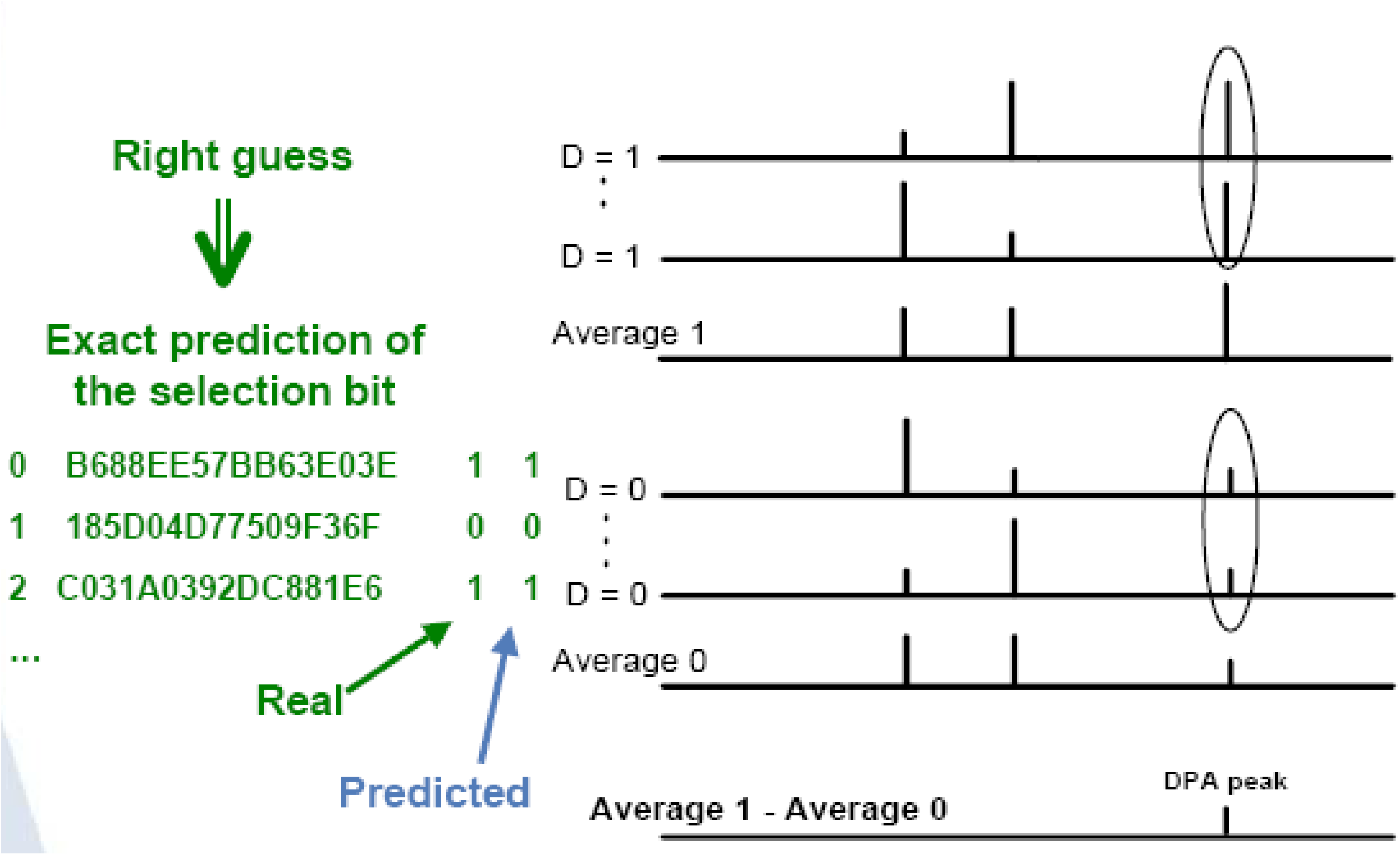
$$\Delta_n = \frac{\sum_{w_i \in S_0} w_i}{|S_0|} - \frac{\sum_{w_i \in S_1} w_i}{|S_1|}$$

DPA -- testing

- The right guess provides the highest spikes !



DPA -- testing



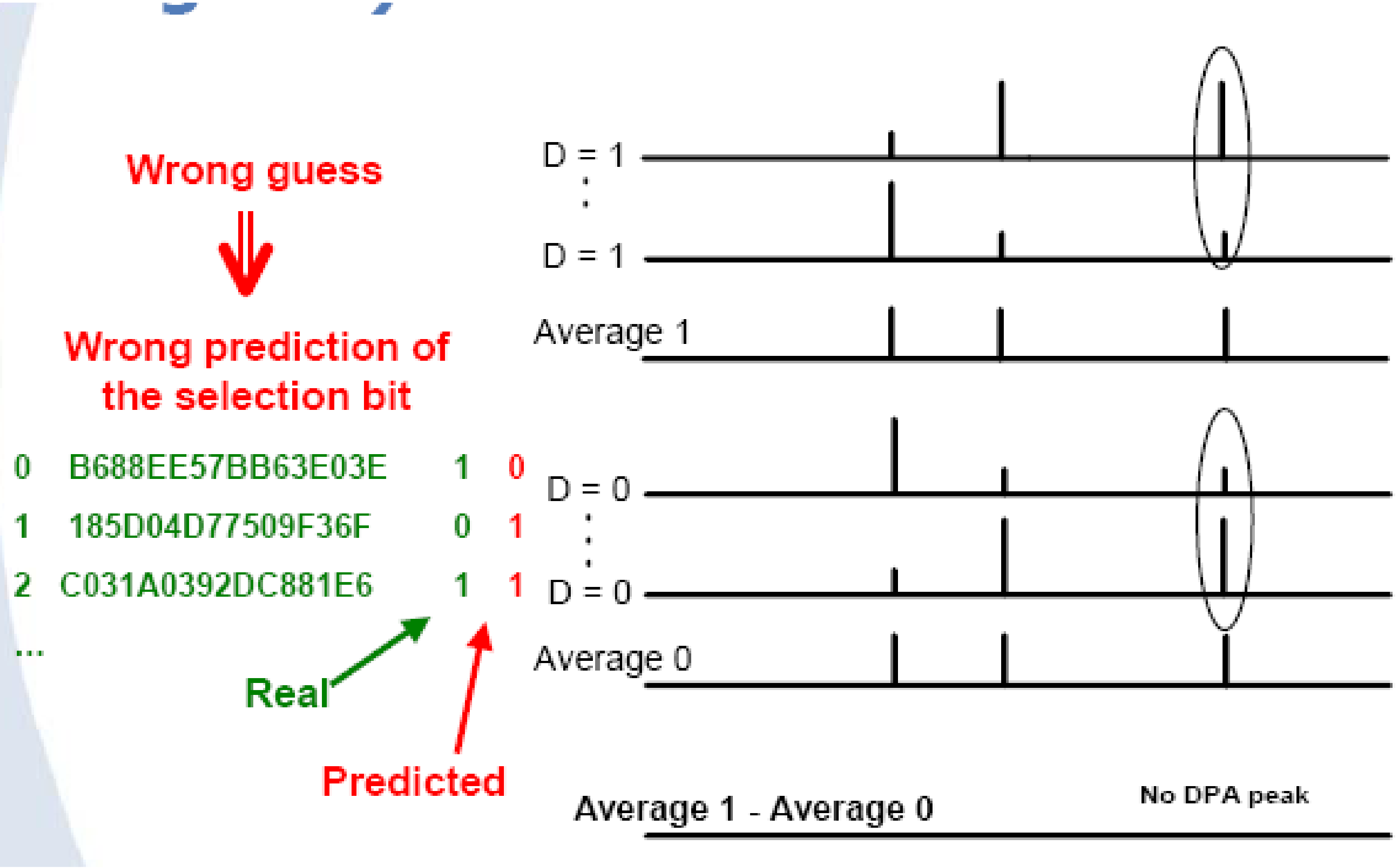
Right guess

⇓

Exact prediction of the selection bit

0	B688EE57BB63E03E	1	1
1	185D04D77509F36F	0	0
2	C031A0392DC881E6	1	1
...			

DPA – the wrong guess



Wrong guess

⇓

Wrong prediction of the selection bit

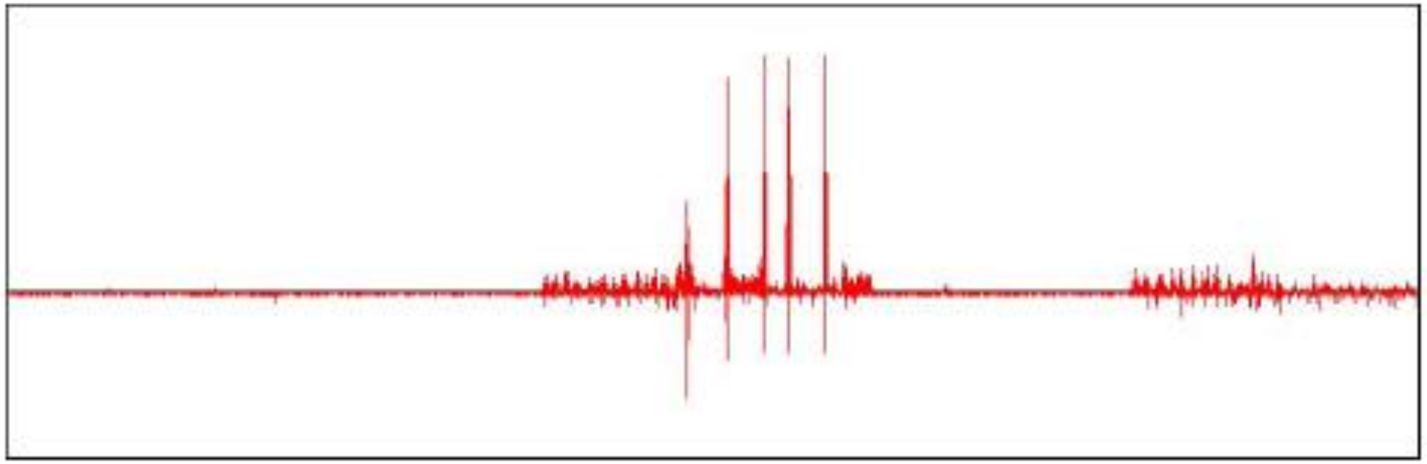
0	B688EE57BB63E03E	1	0
1	185D04D77509F36F	0	1
2	C031A0392DC881E6	1	1
...			

Real

Predicted

DPA (cont’d)

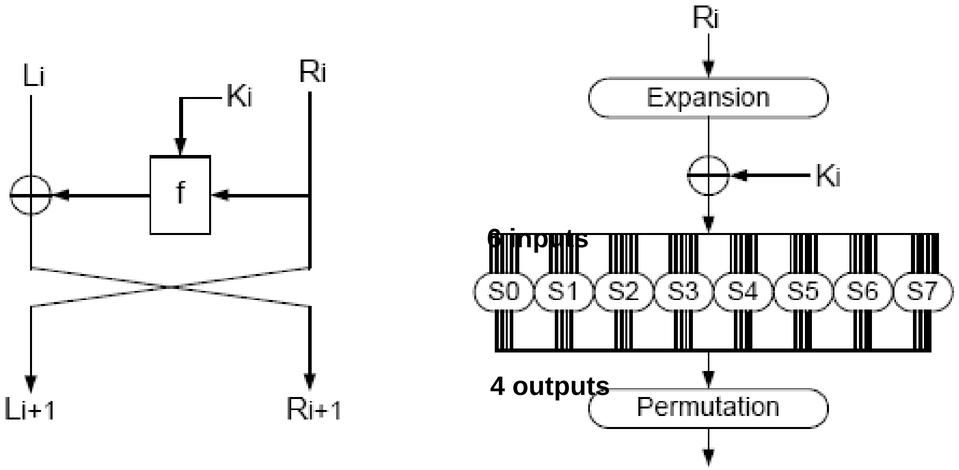
- The DPA waveform with the highest peak will validate the hypothesis



Example: DPA on DES

- Assumption: Attacker presumes detailed knowledge of the DES
- Divide-and-conquer strategy, comparing powers for different inputs
  - Record large number of inputs and record the corresponding power consumption
  - Start with round 15 -- We have access to R_{15}, that entered the last round operation, since it is equal to L_{16}
  - Take this output bit (called M'_i) at the last round and classify the curves based on the bit
    - 6 specific bits of R_{15} will be XOR'd with 6 bits of the key, before entering the S-box
    - By guessing the 6-bit key value, we can predict the bit b, or an arbitrary output bit of an arbitrary S-box output
  - Thus, with 16 partitions, one for each possible key, we can break the cipher much faster

A closer look at HW Implementation of DES

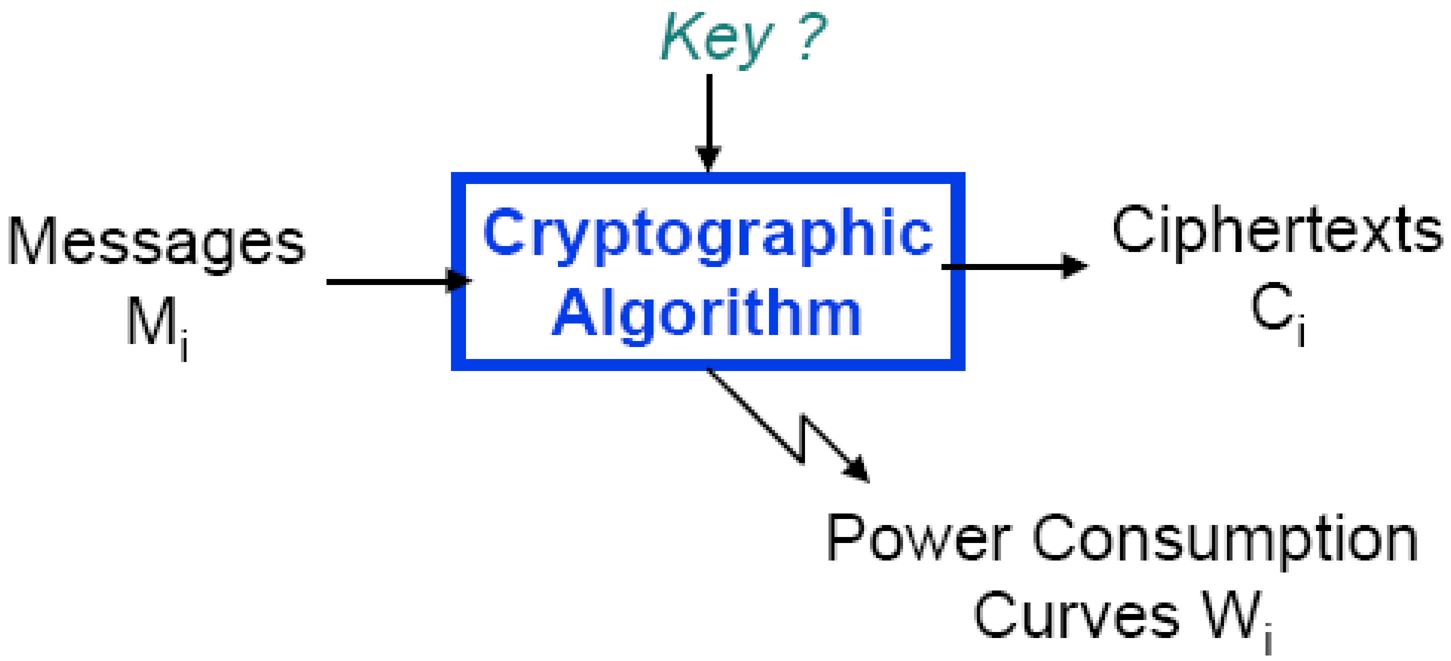


(a) DES round

(b) f function

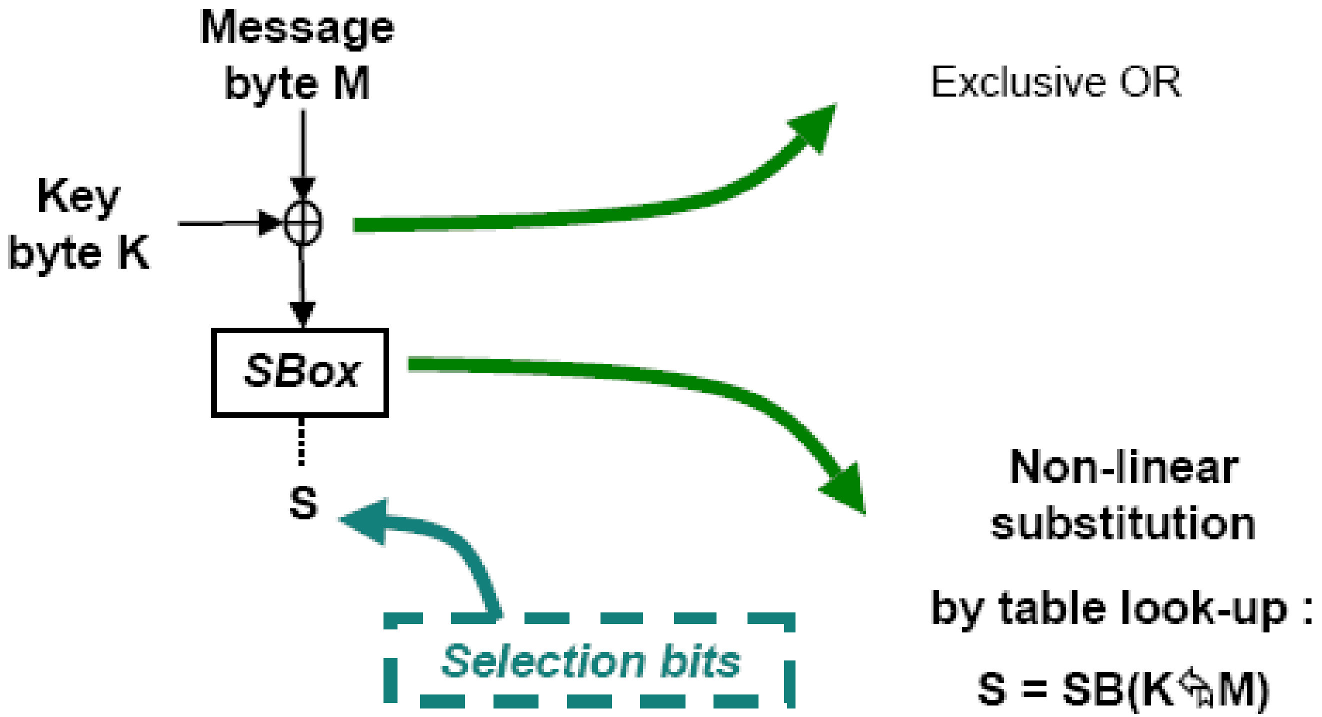
Attacking a secret key algorithm

- DPA works thanks to the perfect prediction of the selection bit
- How to break a key ?



Typical DPA Target

- Basic mechanism in Secret Key algorithms (AES, DES…)



Message byte M

Key byte K

SBox

S

Selection bits

Exclusive OR

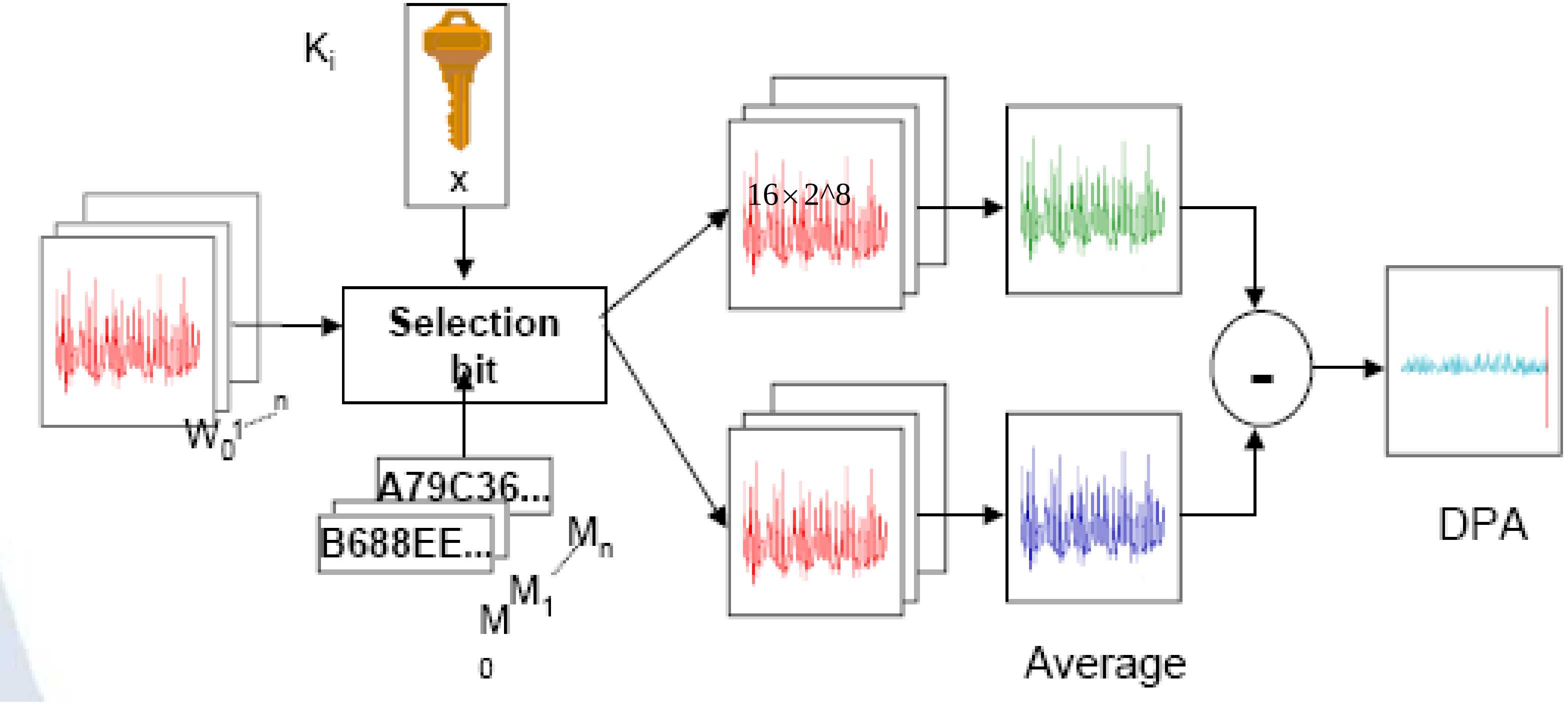
Non-linear substitution

by table look-up :

$S = SB(K \oplus M)$

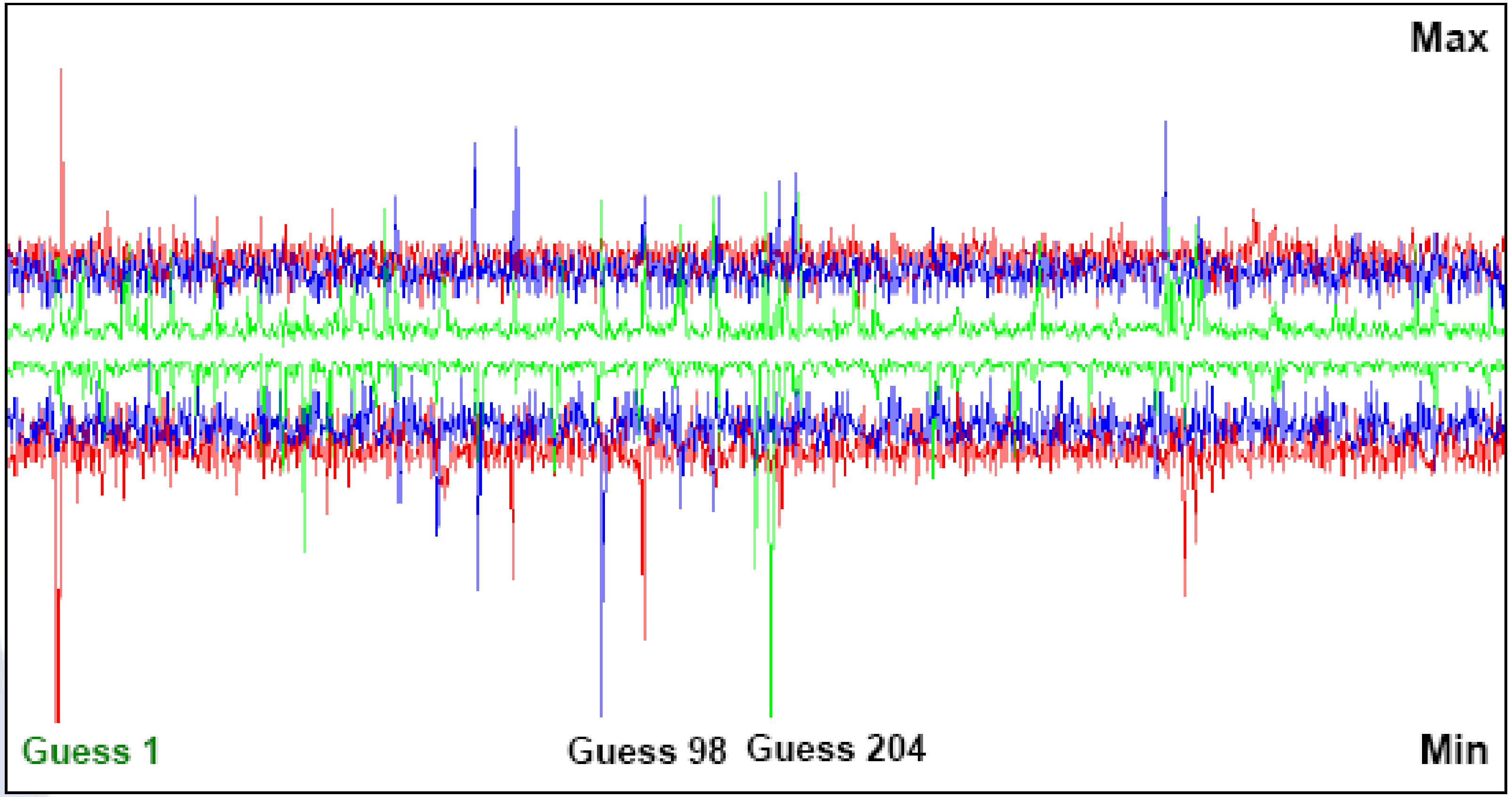
Example – DPA on AES

- Example : AES 128 bits key = 16 bytes K_i (i = 1 to 16)
  - Test 256 guesses per K_i with 256 DPA
  - 128 key bits disclosed with 16 x 256 = 4096 DPA (<< 2^{128} !)



Example – hypothesis testing

DPA on AES : 1st round and 1st byte (right guess = 1)


Max
Guess 1
Guess 98
Guess 204
Min