Introduction to Differential Power Analysis

With Correlation Coefficients

Jiří Buček, Martin Novotný

CTU in Prague, FIT

Training School on Trustworthy Manufacturing and Utilization of Secure Devices

TRUDEVICE 2014, 14-18 July, Lisbon Portugal

Lecture Outline

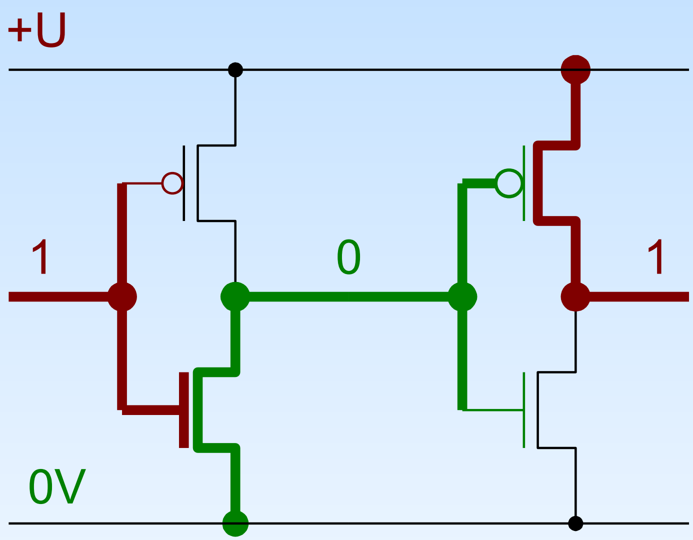
- Power side channel principle (recap)
- Differential Power Analysis
- Intermediate value
- Measured traces
- Power models
- Statistical evaluation
- Number of measurements needed
- Defences

Power Side Channel – CMOS

- CMOS power consumption depends on logic logic circuit activity – rate of logic transitions
- In other words – the more the circuits computes, the more it draws
- Consumption depends also on the values in the computation → side channel
- (CMOS logic has also a static consumption, which we disregard here.)

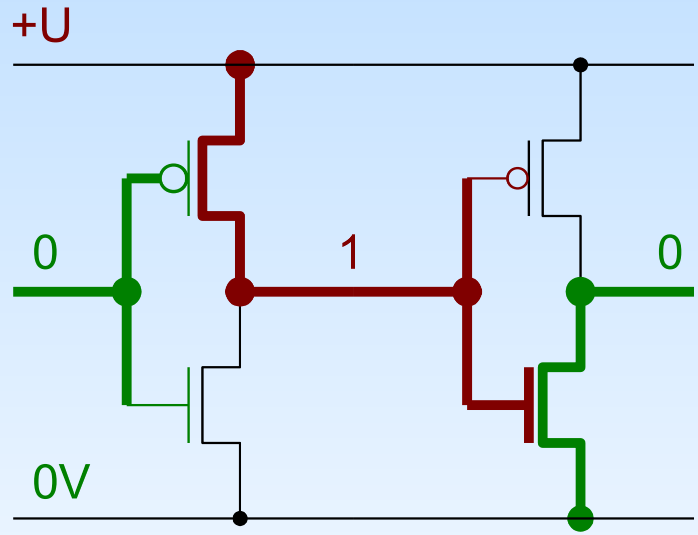
CMOS logic circuit

- How a CMOS inverter works
- Example: Two inverters in series, input log. 1:



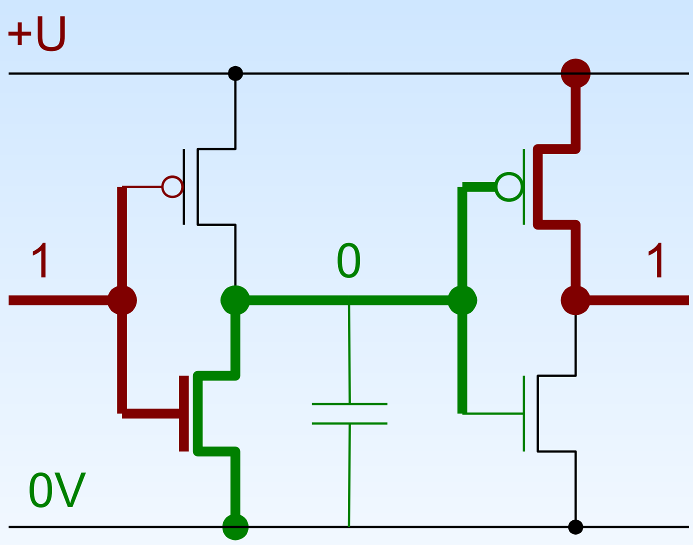
CMOS logic circuit

- How a CMOS inverter works
- Example: Two inverters in series, input log. 0:



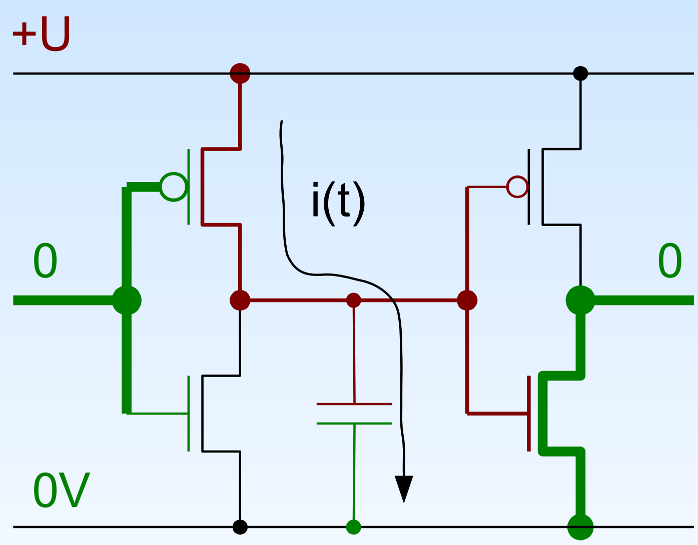
CMOS power consumption

- Real circuit – neither the transistors nor the wires are ideal
- Particularly the resistance and capacity of the transistors and the capacitance of the wires



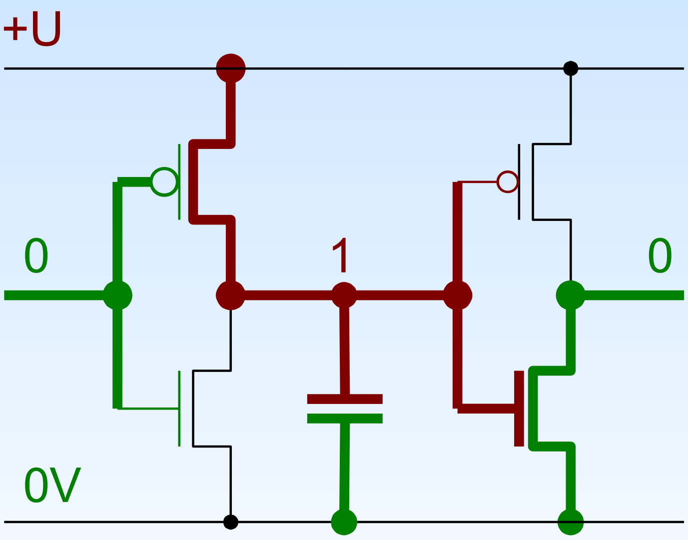
CMOS power consumption

- Input logic change causes current flow through the on-resistance of the transistor to charge the capacitance of the wire and subsequent transistor gates



CMOS power consumption

- After the value settles, the current stops flowing
  - Except for leakage currents, which we neglect here – but leakage is higher in smaller technology (tens of nanometers) so beware

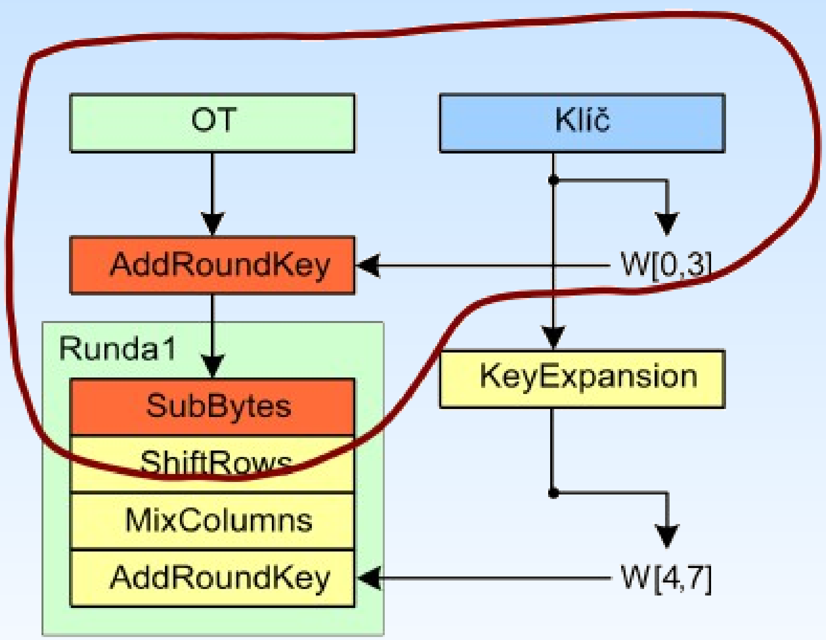


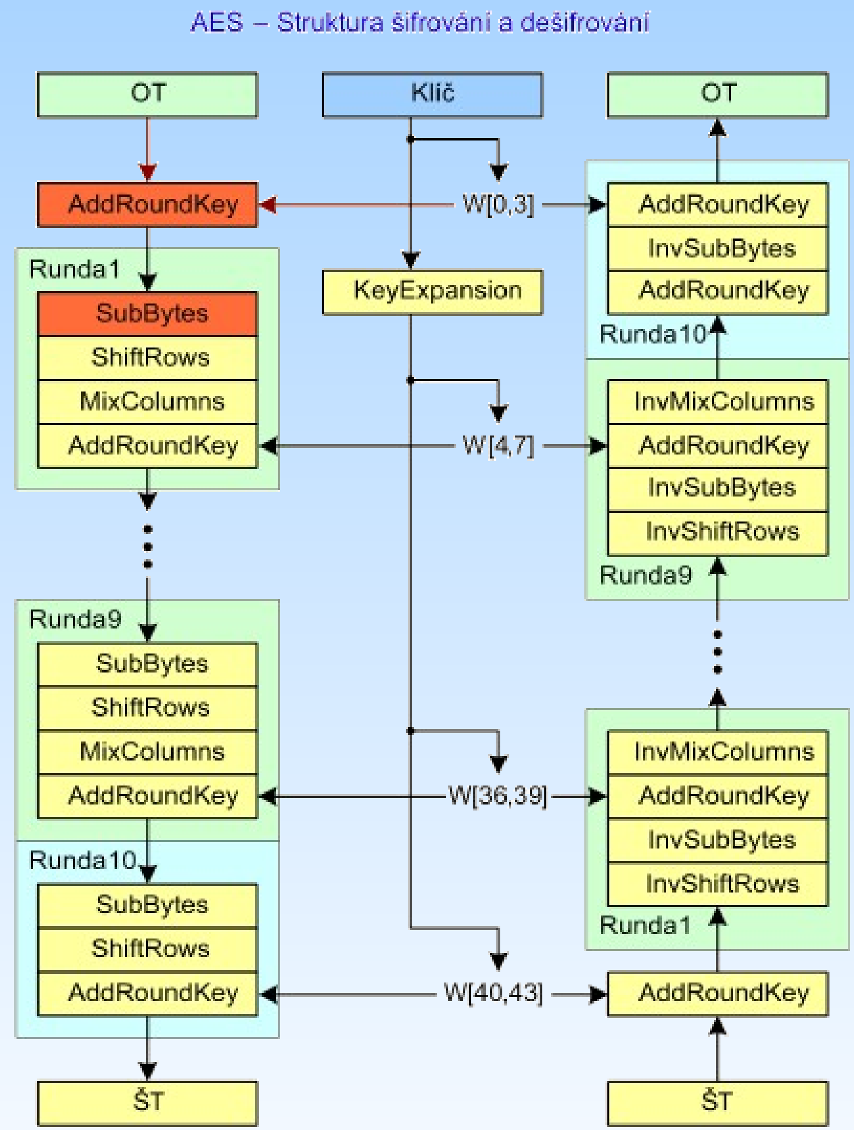
Differential Power Analysis (DPA)

- Choose an **intermediate value** that depends on data and key $v_{i,k} = f(d_i, k)$
- Measure power **traces** $t_{i,j}$ while encrypting data d_i
- Build a matrix of **hypothetical intermediate values** inside the cipher for all possible keys and traces $v_{i,k}$
- Using a power model, compute the matrix of **hypothetical power consumption** for all keys and traces $h_{i,k}$
- Statistically evaluate which key hypothesis best matches the measured power at each individual time

Cipher intermediate value – AES

- Intermediate value must depend on known data and a subkey that can be guessed



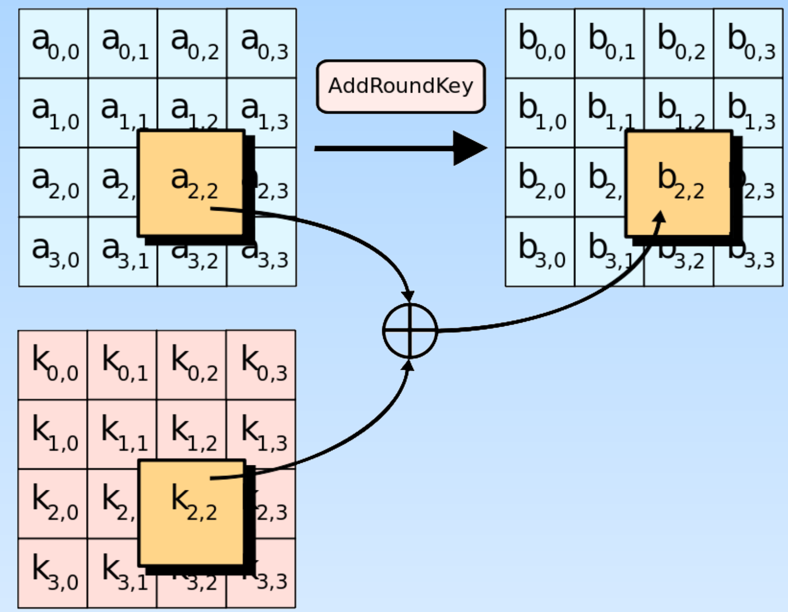


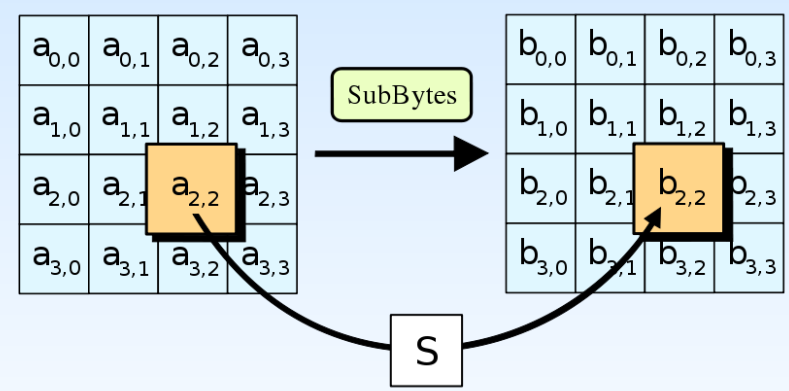
© 2006 R. Lórencz – APK

Intermediate value – AES (2)

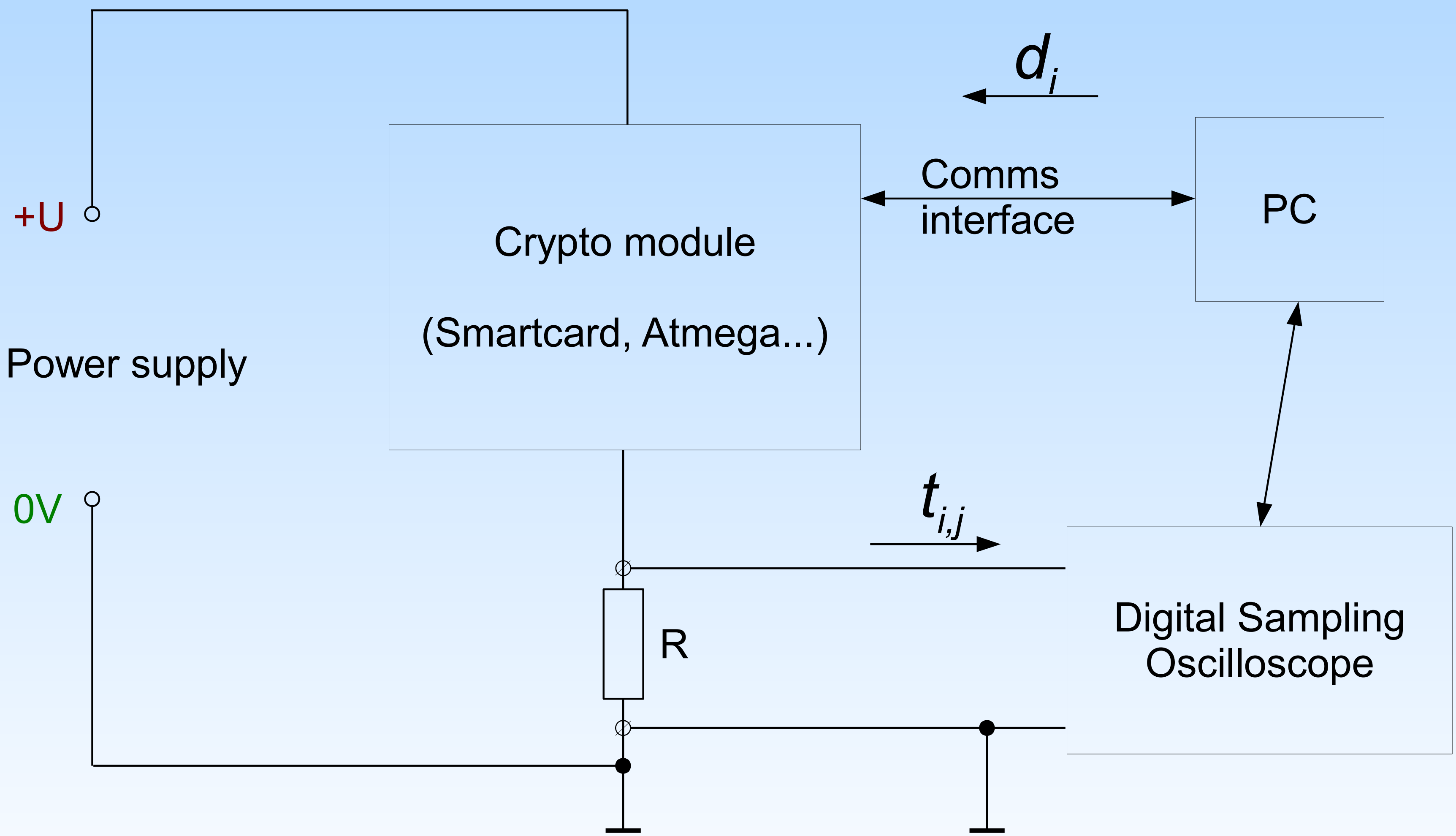
- Input – data (plaintext)
  - known, *a*
- AddRoundKey – XOR
  - „adding“ round key
  - per key byte *k*
- SubBytes – substitution table S (Sbox)
  - again byte by byte

$$v = S(a \oplus k)$$



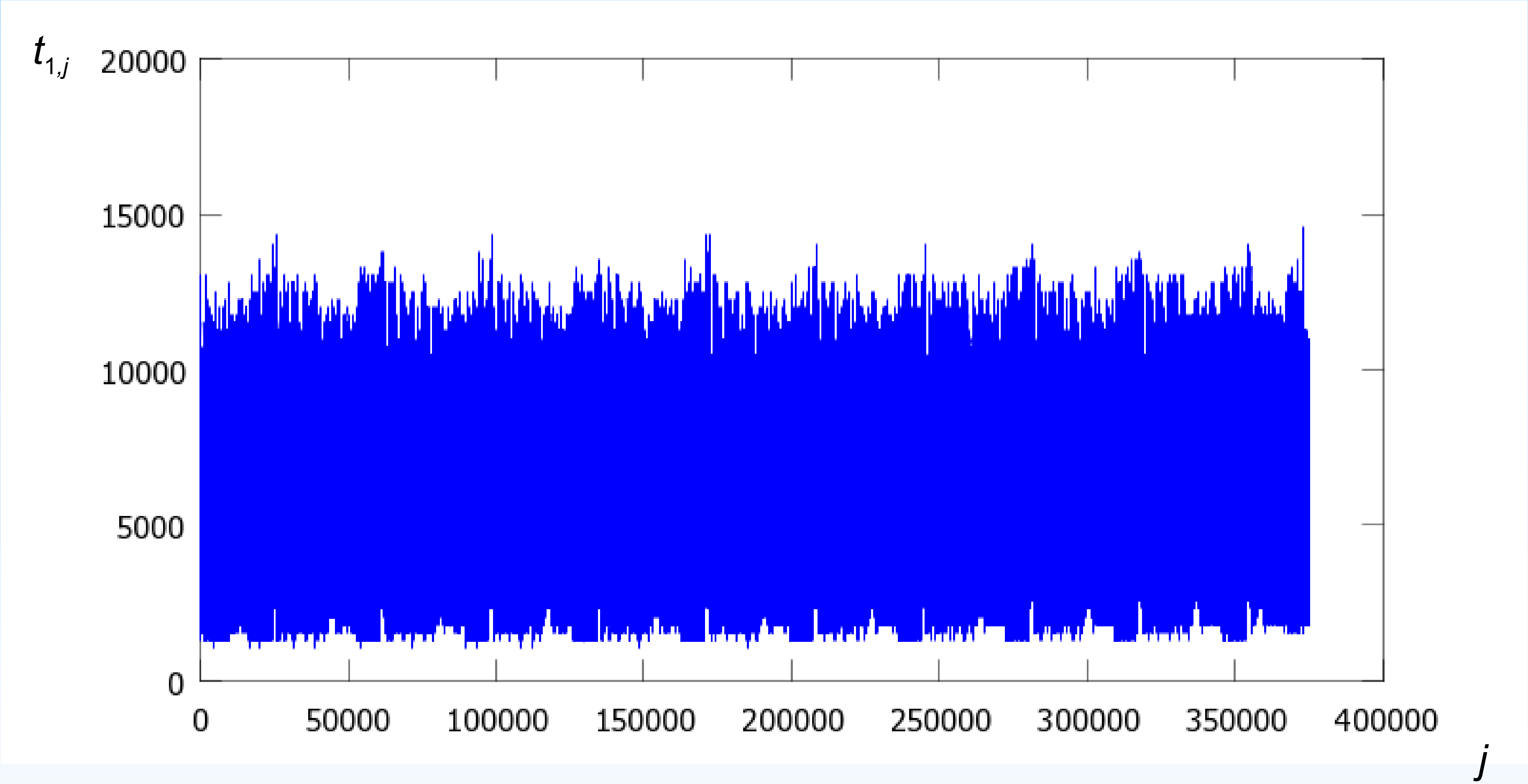


Power Trace Measurement



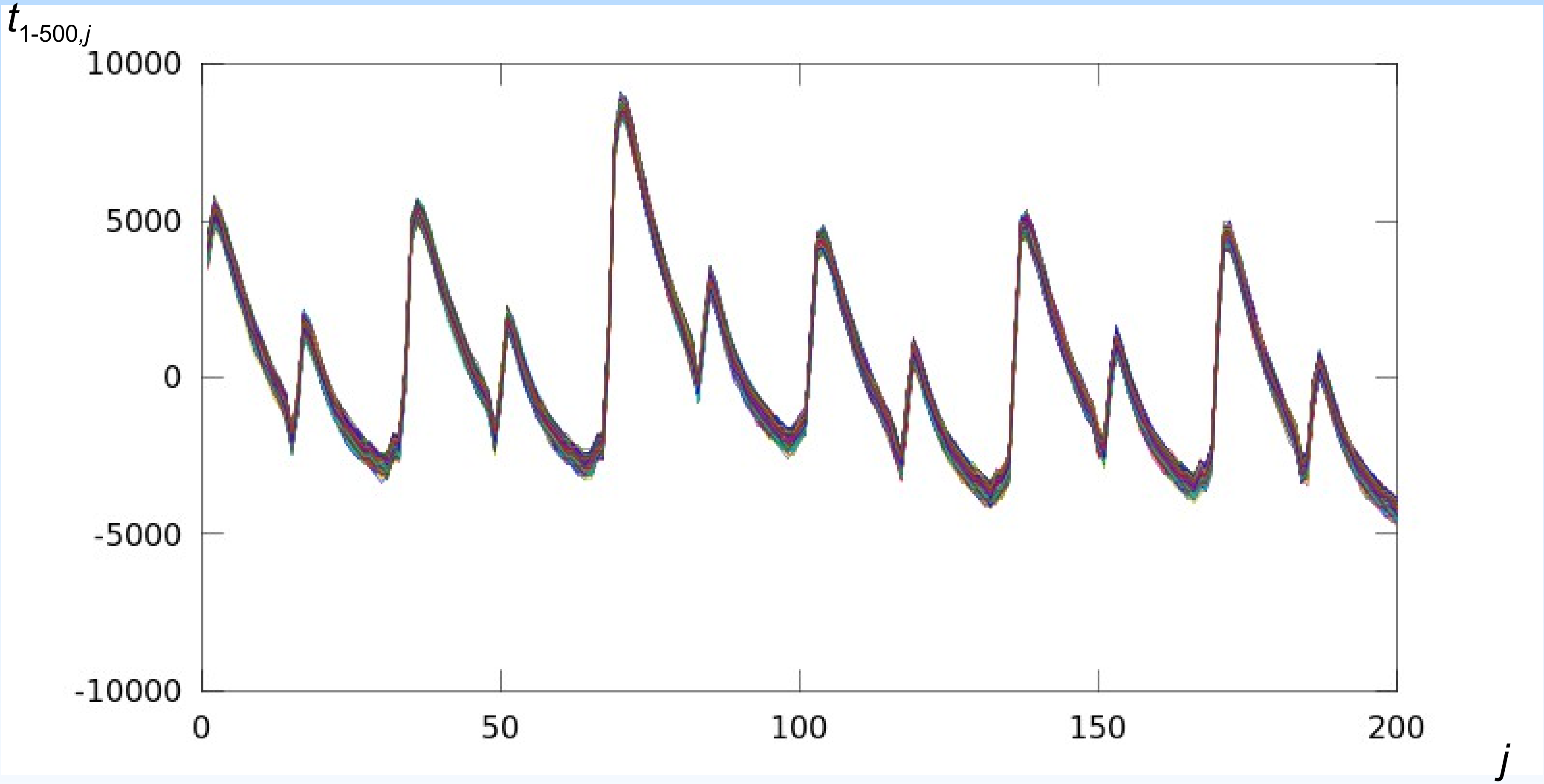
Measured Traces (1)

- One Block (10 rounds of AES)



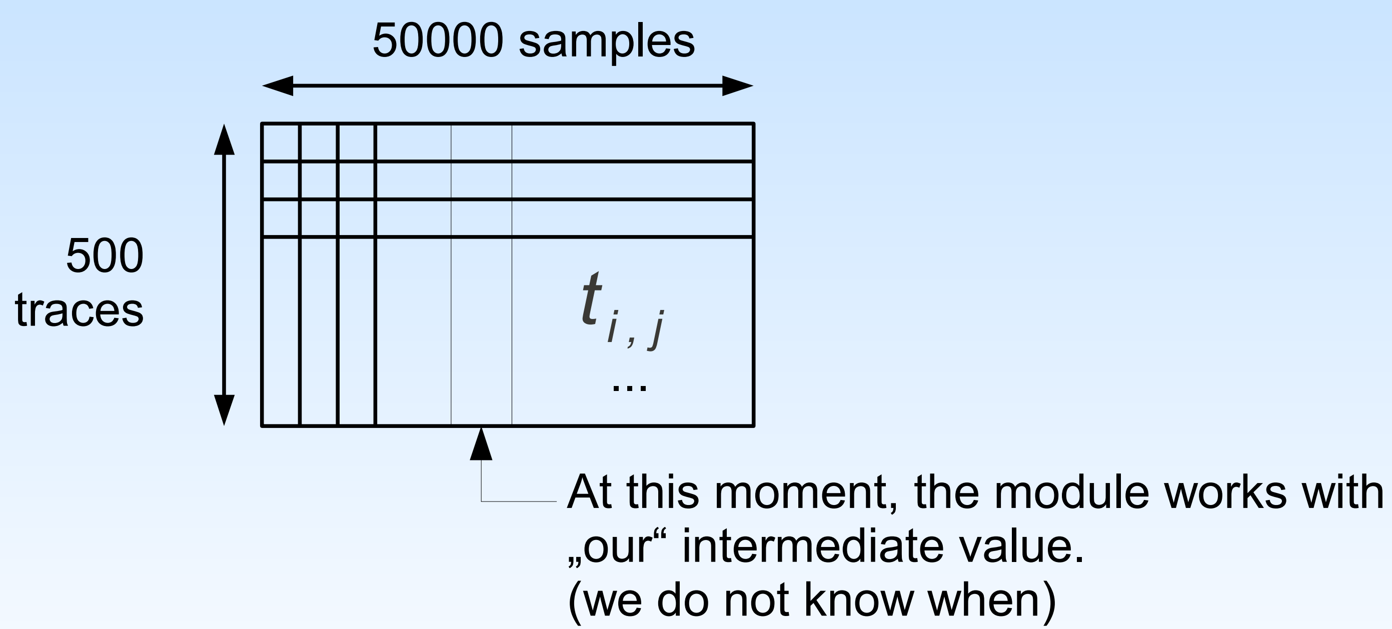
Measured Traces (2)

- Encryption of 500 random blocks (200 samp.)



Measured Traces (3)

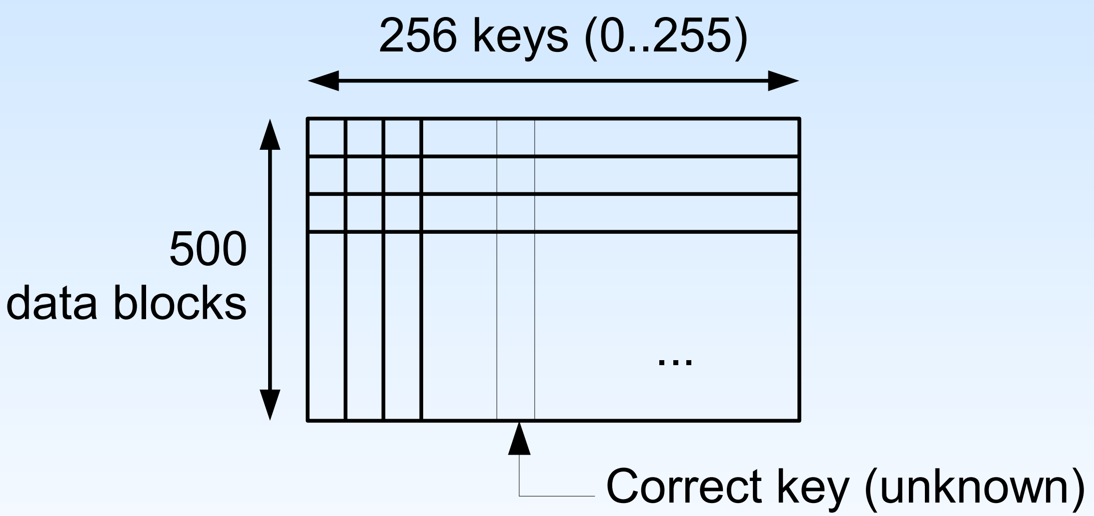
- Store measured trace samples into matrix
- Example: 500 traces by 375000 samples
  - Guessed that 1st round occurs in the first 50000 samples. Take only this section of all traces



Hypothetical intermediate values

- For each of 500 encryptions we get PT and CT
- For now, take only the 1st key byte (of 16)
  - This corresponds to the 1st byte of each PT block
  - There are 256 possible (sub)keys – try all
- We have 500 x 256 intermediate value hypotheses

$$v_{i,k} = S(a_i \oplus k)$$



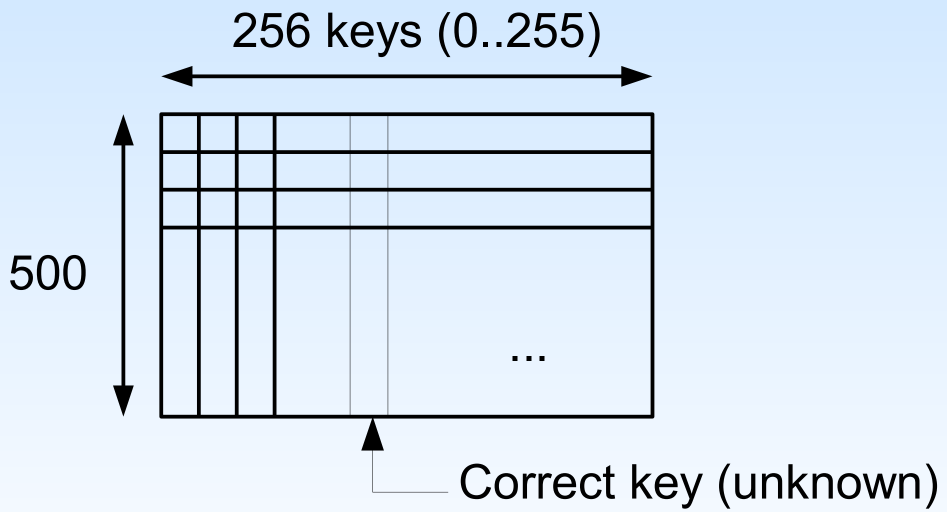
Power Model

- How power consumption depends on intermediate value
- Single bit model – eg. LSB(*v*)
- Hamming weight HW(*v*)
  - Applies mainly to processors, buses with pullups etc.
- Hamming distance $\mathrm{HD}(v, v') = \mathrm{HW}(v \oplus v')$
  - Works also for ASIC, FPGA
  - v' is the previous value (in a register, on bus, in a gate)
  - If v' is constant or is unevenly distributed, usually HW works too

Power Hypotheses

- Apply power model on matrix of intermediate values
- Example: Hamming weight
- Matrix dimension stays the same

$$h_{i,k} = \mathrm{HW}(v_{i,k})$$



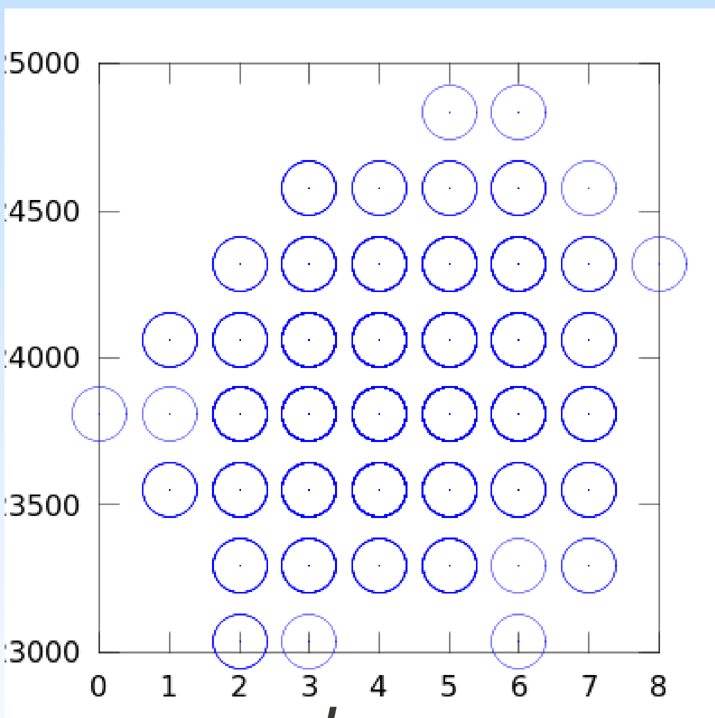
How do we tell the correct key?

- Assume the time is known.
  - We know, when the module works with our intermediate value (j = 26927)

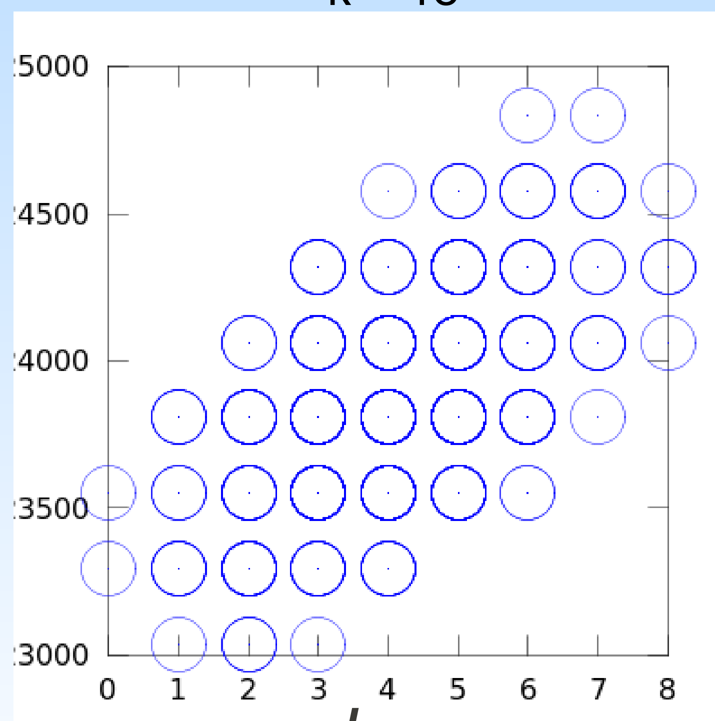
k = 5

k = 18

$t_{i,26927}$



$h_{i,5}$

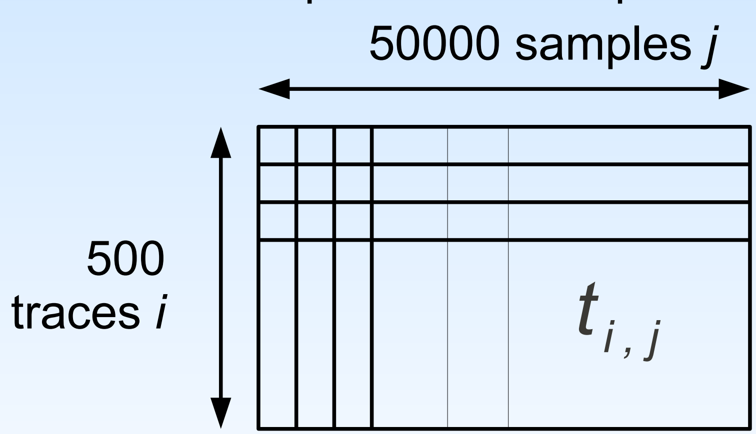


$h_{i,18}$

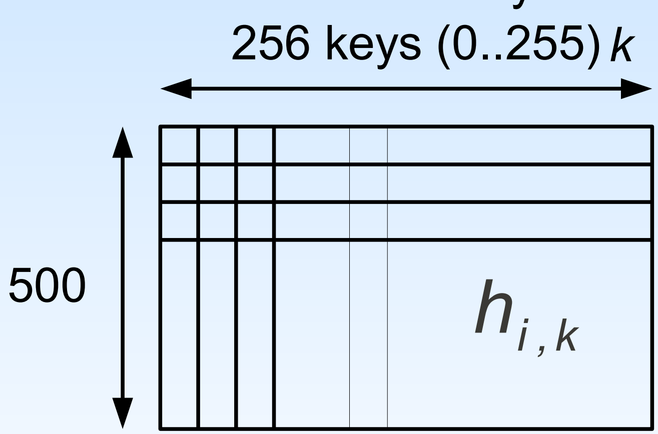
How do we tell the correct time?

- Try all times (samples)
- Evaluate the linear dependence – correlation
  - measured power for all traces in time *j*
  - and hypothetical power consumption for all data and guessed key *k*

Measured power consumption in time *j*



Hypothetical power consumption for different keys *k*



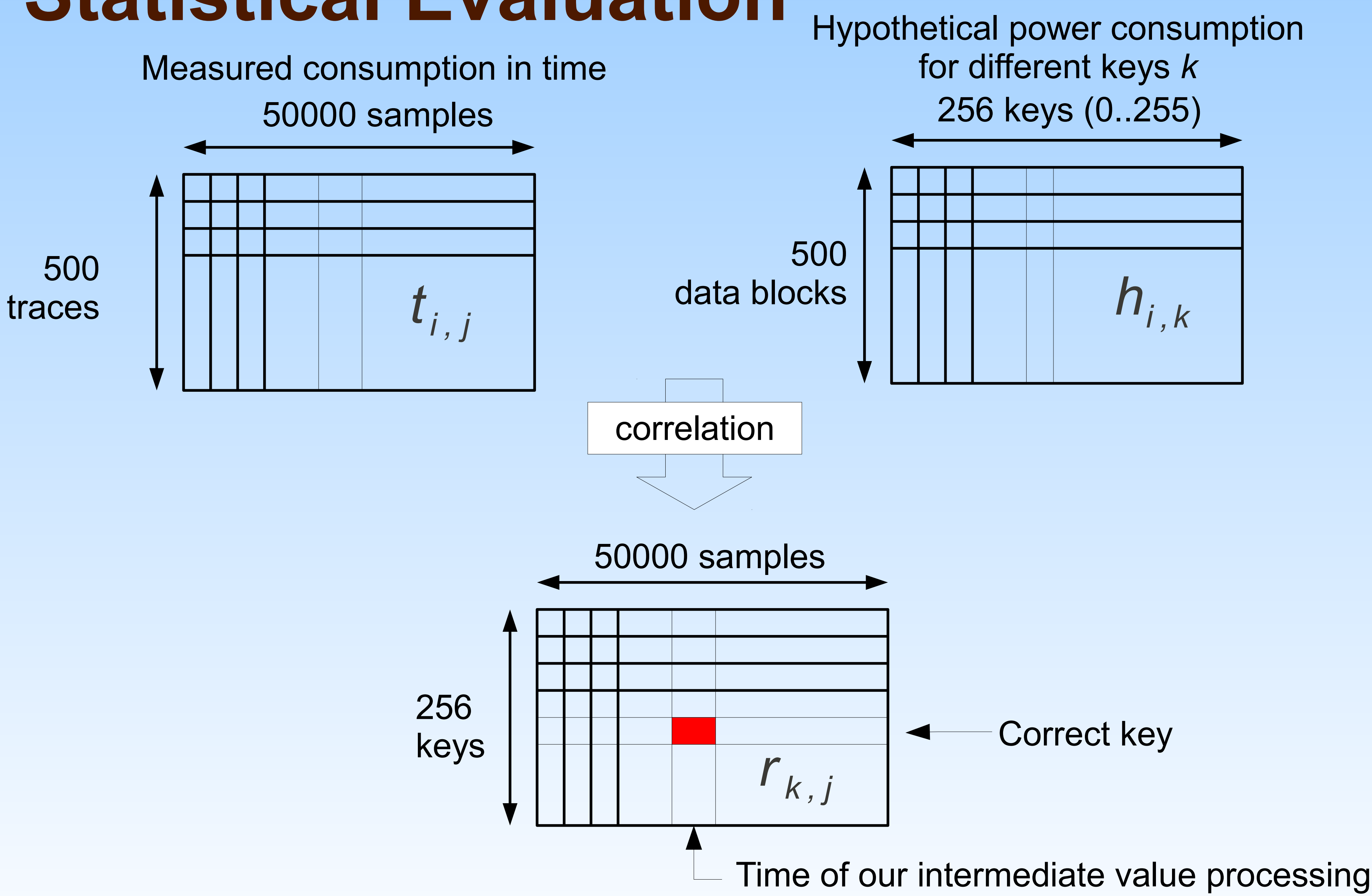
Computing Correlation

- Multiple methods exist
  - Difference of means, distance of means, correlation coefficient
- Correlation coefficient $\rho_{X,Y} = \frac{cov(X,Y)}{\sqrt{var\,X\,var\,Y}}$
- More precisely its point estimate

$$r_{X,Y} = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$$

- range $-1 \leq r \leq 1$

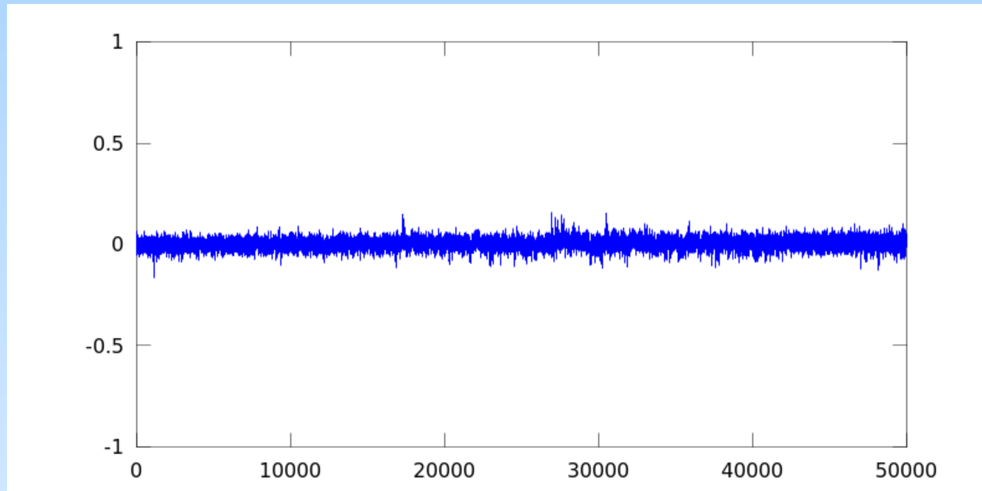
Statistical Evaluation



Key (and time) is found as maximum

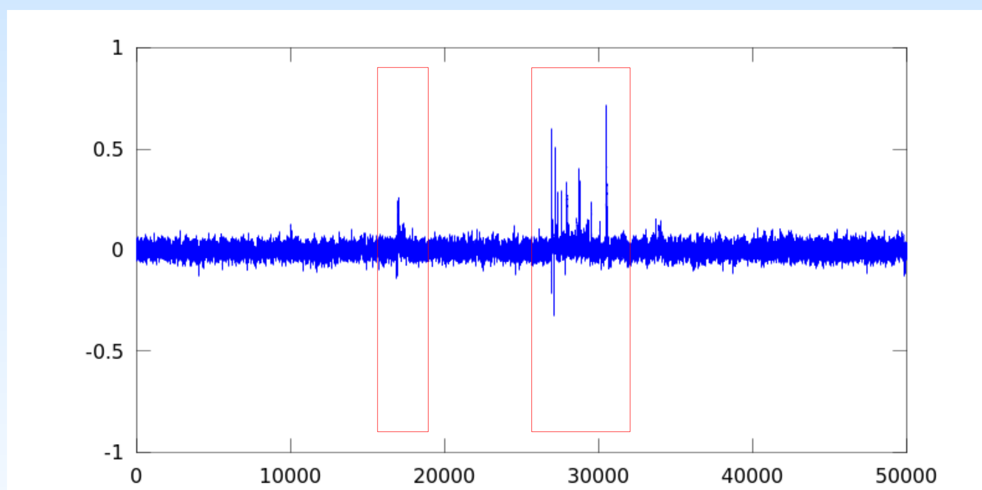
Looking …

k = 5 $r_{5,j}$



…

k = 18 $r_{18,j}$



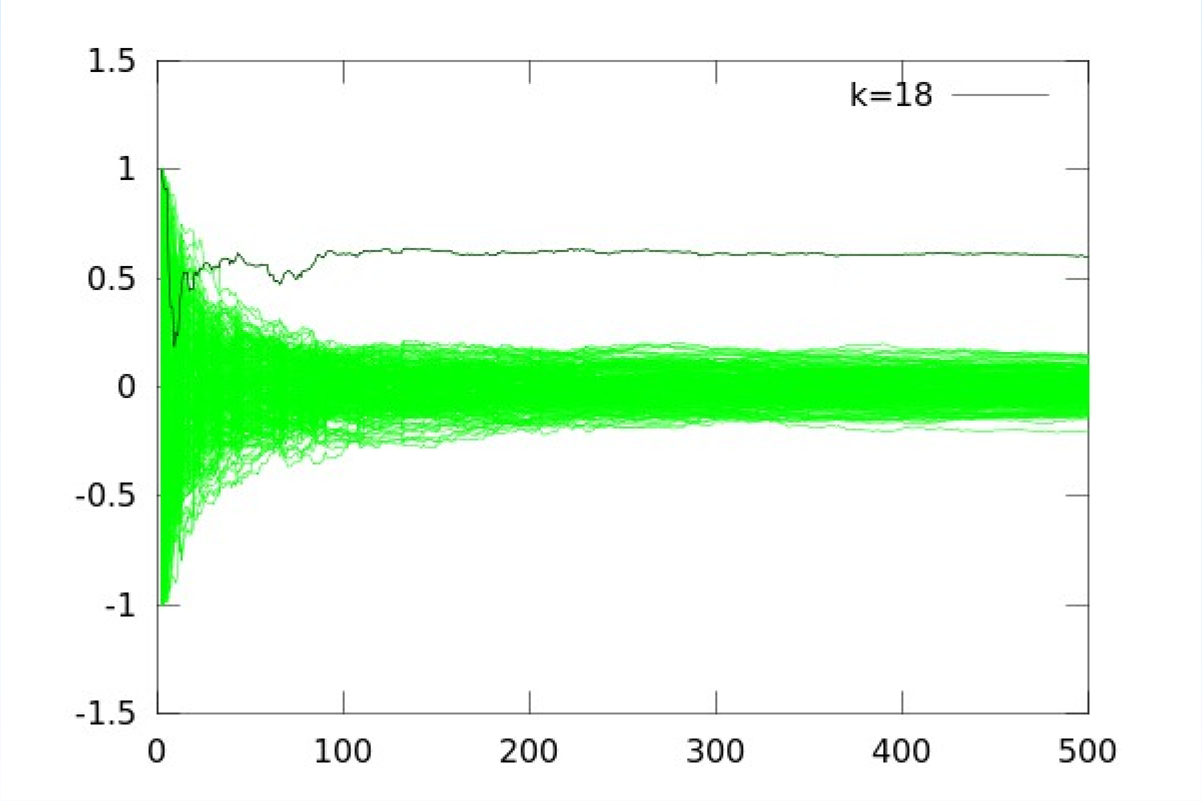
Found!
First key byte
for AES = 0x12

…

How many traces do we need?

- If we would know the time, only ca. 30 would suffice (but we do not)

$r_{k,26927}$



Number of measured traces

Defences – Hiding

- Dependence of consumption on data must be low
- Consumption must appear random
- Hiding in amplitude
  - Parallel data processing (ASIC)
  - Turn on unrelated circuits (ADC ...)
  - Turn on special noise generagtors (switching large capacitance)
  - Special logic gates – dual rail, precharge logic, ...
- Hiding in time
  - Empty (*dummy*) cycle insertion
  - Swapping the order of operations – *shuffling*
  - Random clock frequency changes

Defences – masking

- Logic (boolean)
  - Disrupt dependency between intermediate value and consumption = *xor-in* a random mask, which we later *xor away*
- Arithmetic
  - Similar, but using arithmetic operations
  - Multiplication homomorphism of RSA: $(a \cdot b)^d \equiv a^d \cdot b^d \mod n$
  - Choose mask *m*, compute $m^e \mod n$ (e is public)
  - RSA sign normally: $s = x^d \mod n$
  - RSA sign a message *x* with mask *m*:

$$s_m = (x \cdot m^e)^d \mod n = x^d \cdot m \mod n$$

Logic masking - example

- We have a substitution table S, used as $y = S(x)$
- Choose random **input mask** *m* and **output mask** *m'* (attacker must not know)
- Transform table S to S_m so that

$$S_m(x \oplus m) = S(x) \oplus m'$$

- Attacker cannot successfully guess intermediate values, because he does not know the masks
- Beware of sequential processing of m and $x \oplus m$
  - HD (m, $x \oplus m$) = HW (x) !
- Attacks on masking => Higher order DPA

Conclusion

- We have presented some of the side channel principles
- We have shown an attack on an AES key with differential power analysis (DPA)
- Our example was simplified by using a defenceless processor Atmel AVR (ATmega)
- „Real“ crypto modules (smart cards) usually employ several kinds of defence against DPA – attack is complicated (but not impossible)

Literature

- P. Kocher, J. Jaffe, B. Jun, *Differential Power Analysis*, Advances in Cryptology - Crypto 99 Proceedings, Lecture Notes In Computer Science Vol. 1666, M. Wiener, ed., Springer-Verlag, 1999
- Mangard, S., Oswald, E., Popp., T.: *Power Analysis Attacks – Revealing the secrets of smart cards*, Springer, 2007, ISBN 0-387-30857-1
  - www.dpabook.org