Summary of Side-Channel Analysis and Fault Injection: CPA & DFA

PB173 Domain specific development: Side-Channel Analysis

Łukasz Chmielewski

CRoCS,
Masaryk University,
chmiel@fi.muni.cz

A part of the content is reused with permission of Lejla Batina, Radboud University Nijmegen, The Netherlands.

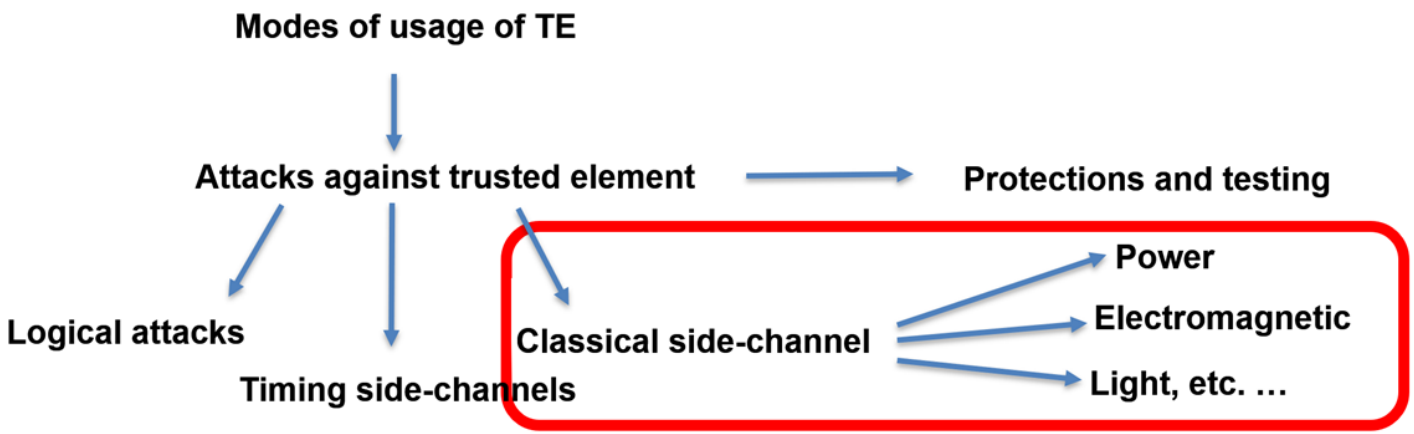
September 21, 2023

CROCS
Centre for Research on Cryptography and Security

Outline

CROCS
Centre for Research on Cryptography and Security

Plan

1. This lecture is originally part of PV204.
2. 

3. If there is time also show Differential Fault Analysis

CROCS
Centre for Research on Cryptography and Security

Timing side-channel: PIN verification

- Software for PIN code verification

```
Input: 4-digit PIN code
Output: PIN verified or rejected
Process CheckPIN (pin[4])
int pin_ok=0;
if (pin[0]==5)
   if (pin[1]==9)
      if (pin[2]==0)
         if (pin[3]==2)
            pin_ok=1;
         end
      end
   end
end
return pin_ok;
EndProcess
```

- What are the execution times of the process for PIN inputs `[0,1,2,3]`, `[5,3,0,2]`, `[5,9,0,0]`
- The execution time increases as we get closer to `[5,9,0,2]`

CRoCS
Centre for Research on Cryptography and Security

Power side channel: CMOS leakage

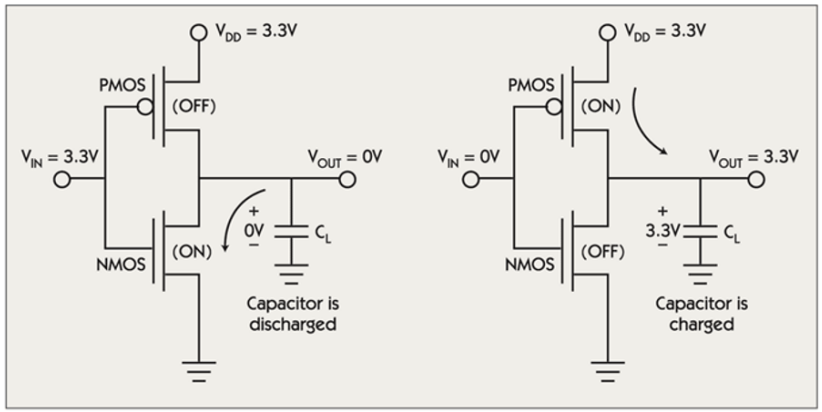


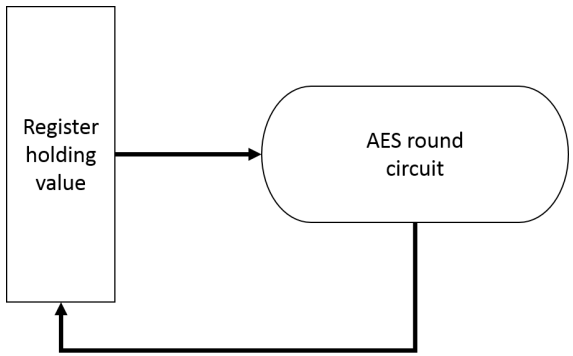
Fig. 1 Dynamic power in a CMOS inverter.

- Complementary Metal-Oxide-Semiconductor (CMOS) is the most popular technology and the CMOS circuits exhibit several types of leakage
- The most relevant for side-channel attacks is the charge and discharge of the CMOS load capacitance a.k.a dynamic power consumption
- Power analysis attack exploits the fact that the dynamic power consumption depends on the data and instructions being processed
- Dynamic power consumption (P_{dyn}) is produced by CMOS transitions from state 0 to 1 and from state 1 to 0
- $P_{dyn} = CV_{DD}^2 P_{0\to1} f$,
  where C the transistor capacitance, V_{DD} the power supply voltage, f the frequency and $P_{0\to1}$ the probability of a $0 \to 1$ transition

CROCS
Centre for Research on Cryptography and Security

Power side channel: Modeling the leakage

- Starting point: we use the number of transitions to model the leakage
- The Hamming distance model counts the number of $0 \rightarrow 1$ and $1 \rightarrow 0$ transitions
- Example 1: A register R is storing the result of an AES round and initial value v_0 gets overwritten with v_1



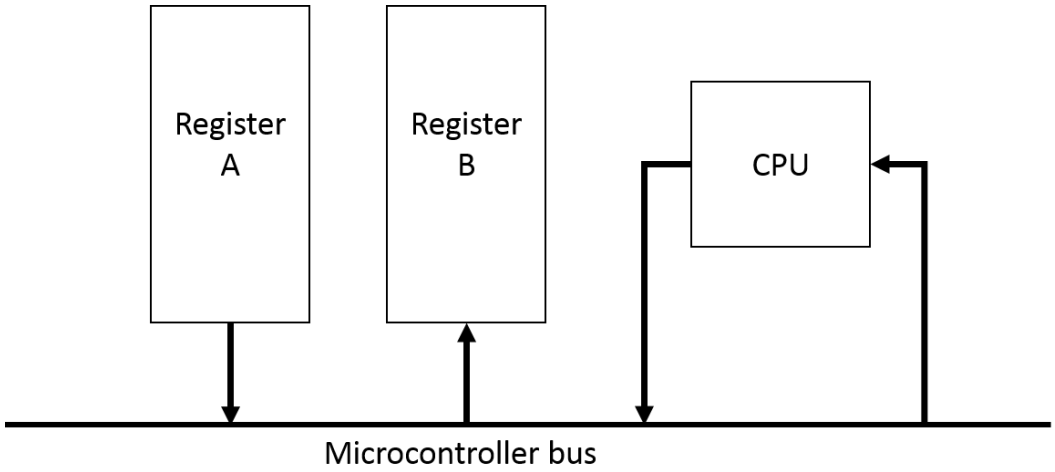
- The power consumption because of the register transition $v_0 \rightarrow v_1$ is related to the number of bit flips that occurred
- Thus it can be modeled as HammingDistance(v_0, v_1) = HammingWeight$(v_0 \oplus v_1)$
- Common leakage model for hardware implementations (FPGA, ASIC)

CROCS
Centre for Research on Cryptography and Security

Power side channel: Modeling the leakage

- Example 2: In a microcontroller, a register A contains value v_0 and an assembly instruction moves the content of register A to B

```
mov rB, rA
```

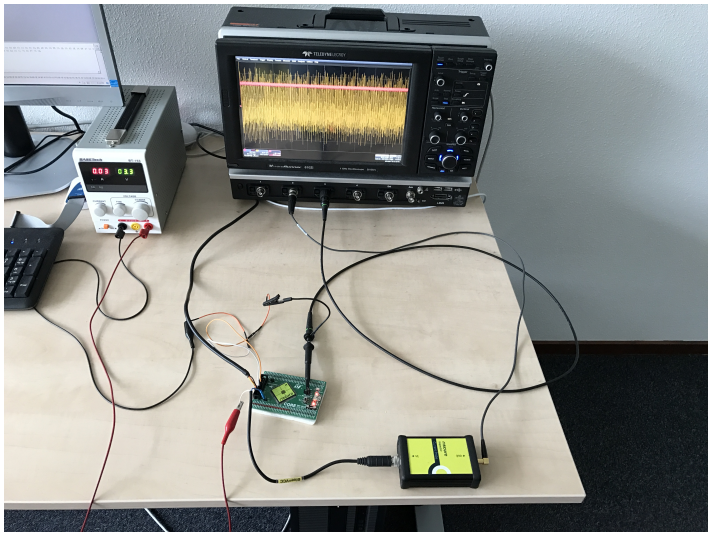


- This instruction transfers v_0 from A to B via the CPU, using the bus
- Typically the bus is precharged at all bits being zeros or one (busInitialValue)
- The power consumption of the instruction can be modeled as HammingDistance(busInitialValue,v_0) = HammingWeight($v_0 \oplus 0$) = HW(v_0)
- Common leakage model for software implementations (AVR/ARM)

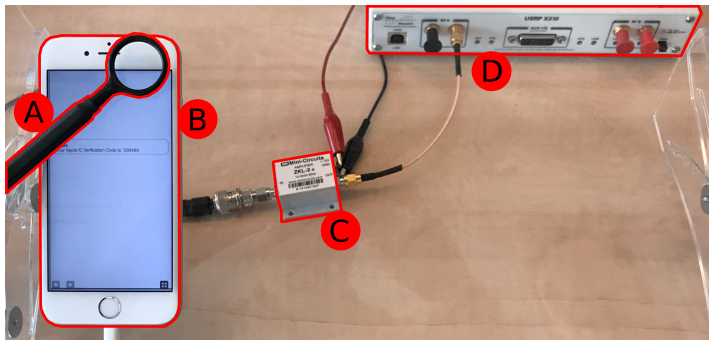
CROCS
Centre for Research on Cryptography and Security

Actual setups

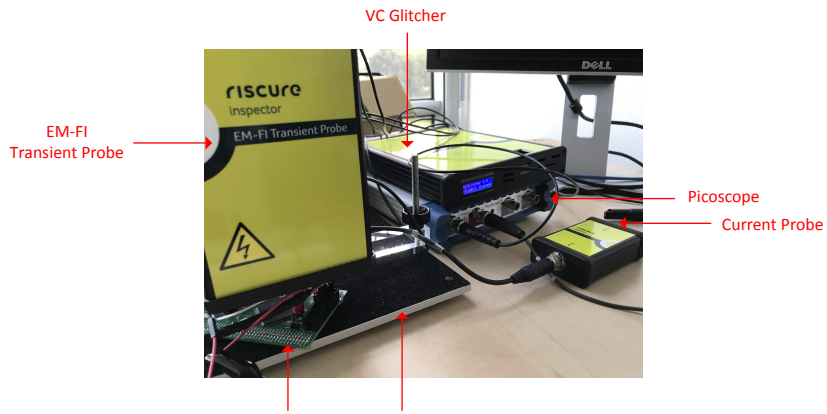
DPA setup with ARM CortexM4



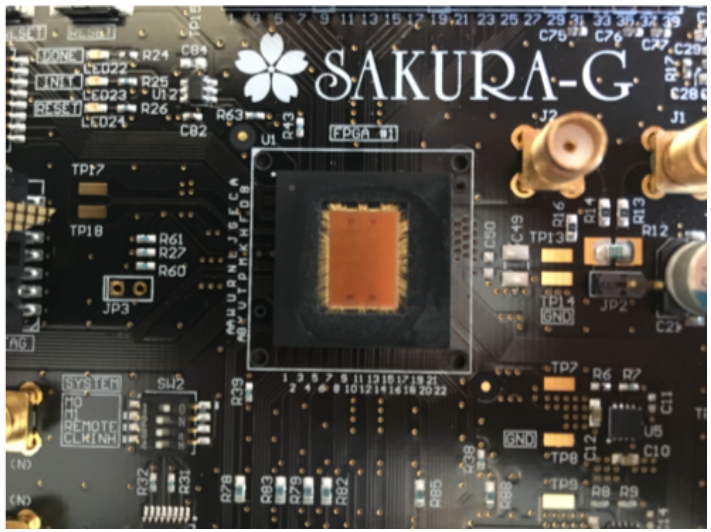
Tempest



FA setup



FPGA board for SCA



CROCS
Centre for Research on Cryptography and Security

Analysis capabilities

- Simple Power Analysis (SPA): one or a few measurements - visual inspection or some simple signal processing
- Differential attacks (DPA): multiple measurements - use of statistics, signal processing, etc.
- Higher order attacks: Univariate vs multivariate
- Profiled attacks: Template and Deep Learning attacks
- Combining two or more side-channels
- Combining side-channel attack with theoretical cryptanalysis

CROCS
Centre for Research on Cryptography and Security

Simple Power Analysis (SPA)

- Based on one or a few measurements
- Mostly discovery of data-(in)dependent but instruction-dependent properties e.g.
  - Symmetric:
    - Number of rounds (resp. key length)
    - Memory accesses (usually higher power consumption)
  - Asymmetric:
    - The key (if badly implemented, e.g. RSA / ECC)
    - Key length
    - Implementation details: for example RSA w/wo CRT

CROCS
Centre for Research on Cryptography and Security

Reminder: DPA, main concepts

The DPA assumption: the attack is possible assuming the existence of a sensitive variable (for which exhaustive key search is possible) that depends on something we know (msg) and something we want to learn (key).

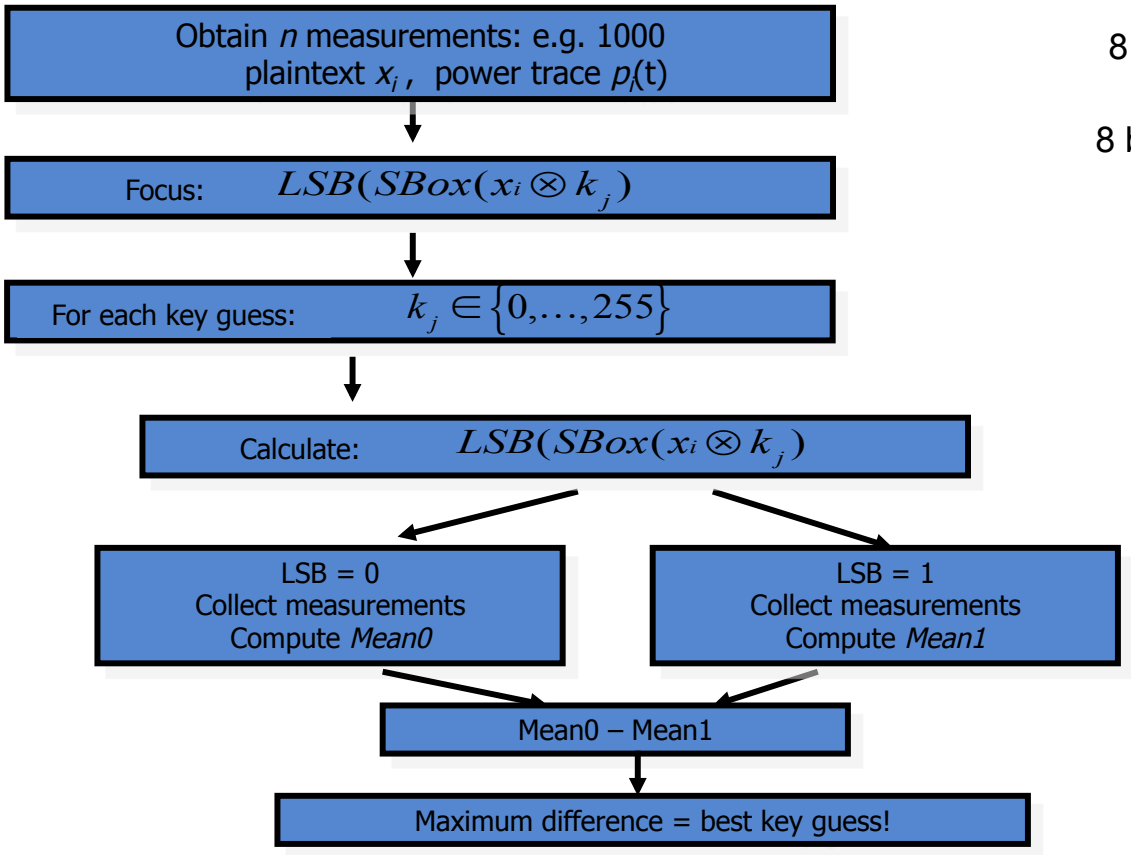
Main steps

1. Choose your sensitive variable
2. Collect measurements, known plaintext/ciphertext, sub-key guesses
3. Predict (hypothetical) intermediate values
4. Decide on the leakage model
5. Recover the key by statistical means, using partition or comparison method as the side-channel distinguisher

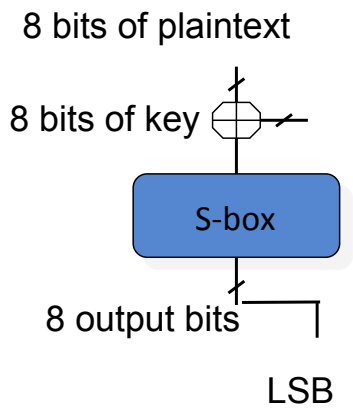
CROCS
Centre for Research on Cryptography and Security

DPA with Distance of Means (DoM) as distinguisher

Classical 1-bit DPA on AES using DoM



AES impl.



1000 measurements *
time window *t* *
256 key guesses

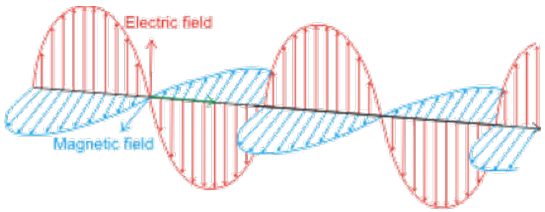
[Kocher et al.]

Power analysis notes and literature

- Very powerful attacks that require contact with the target
- Countermeasures on different layers required i.e. algorithm, implementation, transistor
- P. Kocher, J. Jaffe, B. Jun. “Differential Power Analysis”, CRYPTO 1999.
- T. Eisenbarth et al. “On the Power of Power Analysis in the Real World: A Complete Break of the KeeLoqCode Hopping Scheme”, CRYPTO 2008.
- Mangard et al. Power Analysis Attacks, Springer, 2006.
- T. Kasper et al. “All You Can Eat or Breaking a Real-World Contactless Payment System”, Financial Cryptography 2010.
- J. Balasch et al. “Power Analysis of Atmel CryptoMemory - Recovering Keys from Secure EEPROMs”, CT-RSA 2012.
- N. Samwel et al. “Breaking Ed25519 in WolfSSL.”, CT-RSA 2018.

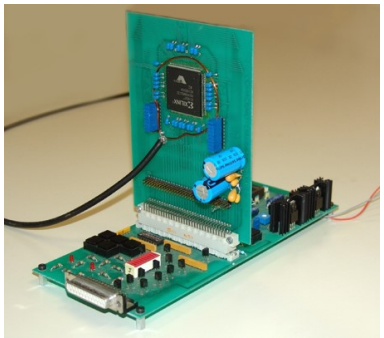
CRoCS
Centre for Research on Cryptography and Security

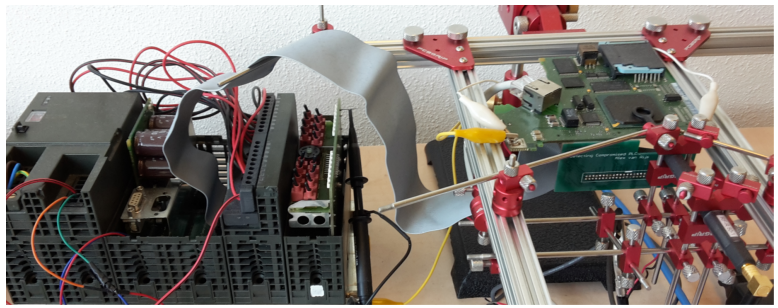
Electromagnetic side channel



EM side channel: Probing

- Observing a power signal in more complex systems can be messy
- Complicated SoCs with multiple peripherals
- Countermeasures trying to flatten the power consumption signal
- Use an electromagnetic probe instead



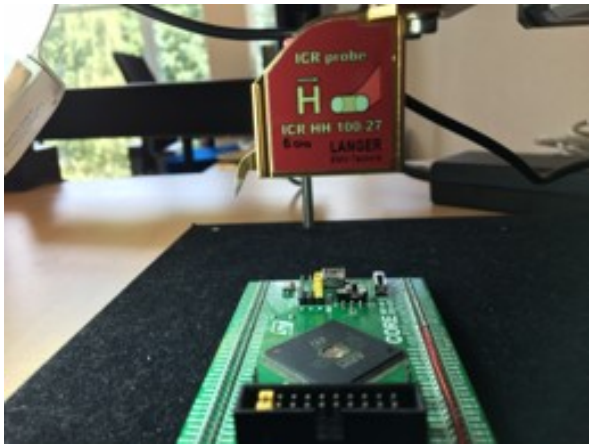


- A probe is used to access the power consumption with less board modifications
- Smaller probes can focus on interesting locations and ignore interference from unrelated el. components

CROCS
Centre for Research on Cryptography and Security

EM side channel: Decapsulation and Microprobing

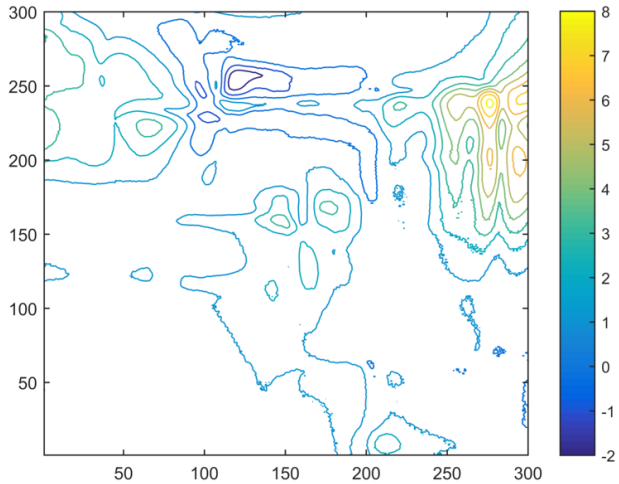
- To improve spatial resolution of analysis use a micrometer-sized antenna
- To exploit more leakage decapsulate the chip using chemicals



CROCS
Centre for Research on Cryptography and Security

EM side channel: Decapsulation and Microprobing

- Left: close inspection of decapsulated ARM processor using a microscope
- Right: EM emission heat-map of the same chip



CROCS
Centre for Research on Cryptography and Security

EM side channel: notes and literature

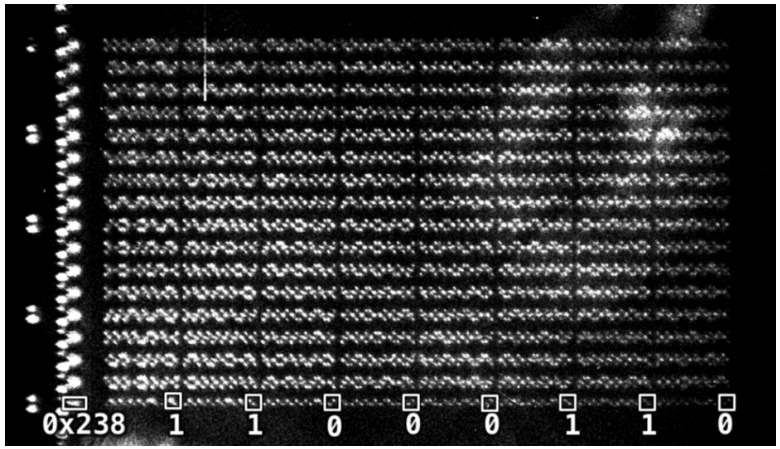
- EM enables side-channel attacks both in high proximity scenarios and distance scenarios
- Main side channel for SoCs, FPGAs, contactless cards due to their complexity and communication methods
- TEMPEST-like attacks are also targeting private data and authentication methods such as code etc.
- Gandolfi et al.: Electromagnetic Analysis: Concrete Results, CHES 1999.
- Andrikos et al.: Location, location, location: Revisiting modeling and exploitation for location-based side channel leakages, ASIACRYPT2019

CROCS
Centre for Research on Cryptography and Security

Exotic side channels

Exotic side channels

Optical Emission



- Accessing the chip SRAM cells emits photons that can be detected by a high-resolution camera
- Visual inspection can reveal the memory location accessed
- The memory location maps to a specific value (e.g. in the AES LUT), i.e. it maps directly to $Sbox(in \oplus key)$
- Since the input *in* is known, knowledge of the memory location reveals the key
- Schlösser et al.: Simple Photonic Emission Analysis of AES, CHES 2012.

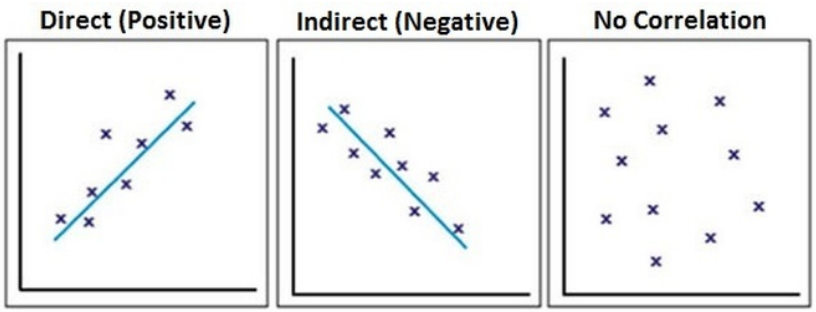
CROCS
Centre for Research on Cryptography and Security

Out-of-order and speculative execution

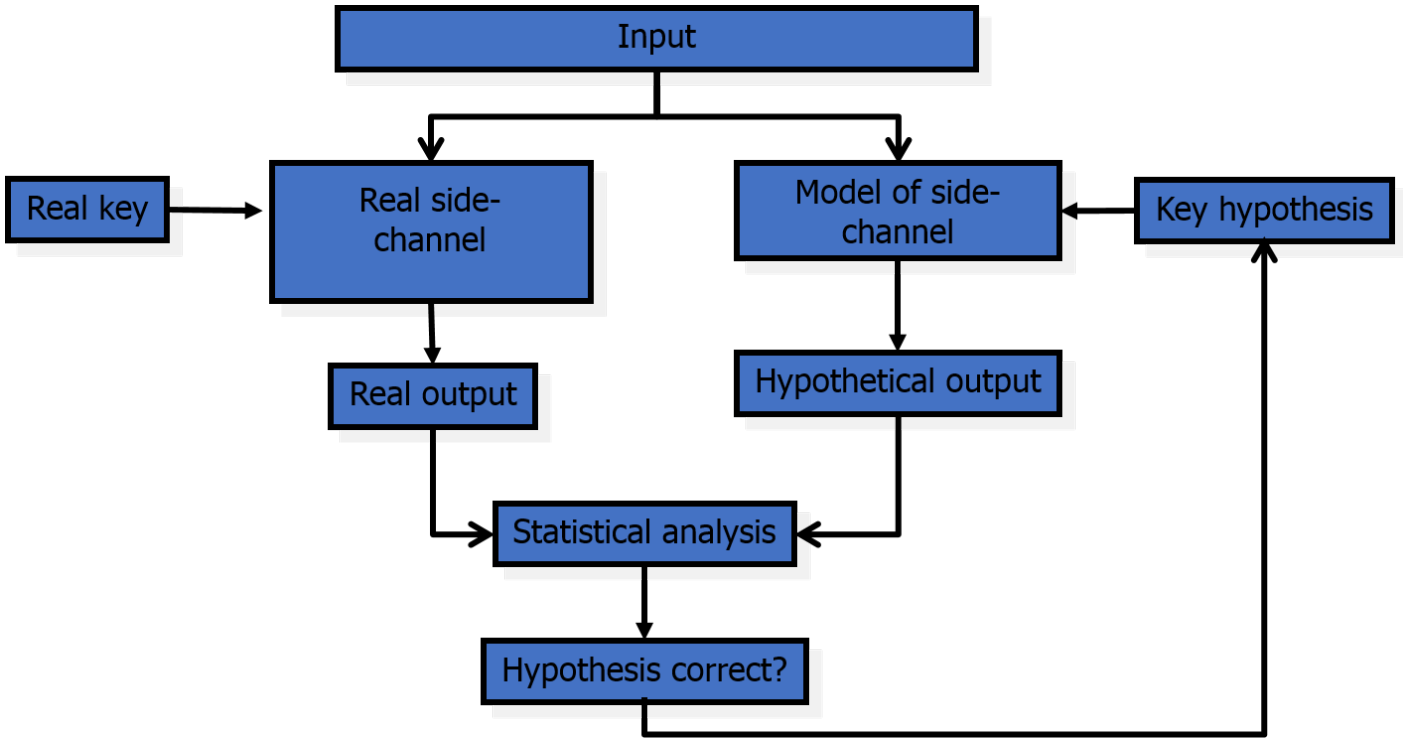
- Meltdown and Spectre Attacks
- Kernel addresses were access unintentionally due to out-of-order execution
- Taking all possible branches may also cause issues
- Foreshadow as a variant of Meltdown on SGX
- Recent ones: RIDL, ZombieLoad, ...

CROCS
Centre for Research on Cryptography and Security

Correlation Power Analysis (CPA)



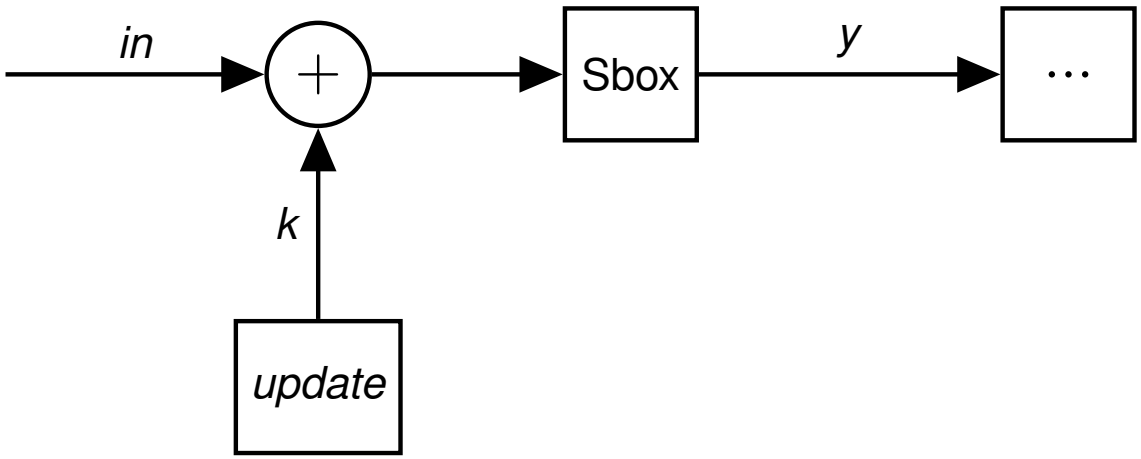
CPA: the main principle



- Brier et al.: Correlation Power Analysis with a Leakage Model, CHES 2004
- Chapter 6 in the blue book

CROCS
Centre for Research on Cryptography and Security

CPA Step 1: Choose intermediate value and decide on the leakage model

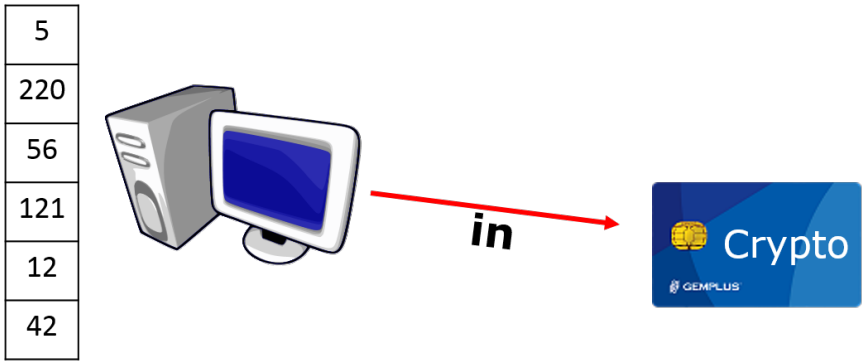


- Assume a software AES implementation e.g. for AVR microcontrollers (8-bit arch.)
- Step 1: Choose an intermediate value v of the AES cipher to attack
- The value v must be a function of the input and the key, i.e. $v = f(in, k)$
- A common choice for v is the Sbox output, i.e. $v = y = Sbox(in \oplus k)$
- Throughout the attack the key k must remain constant
- Throughout the attack the input in is random

CROCS
Centre for Research on Cryptography and Security

CPA Step 2: Measure the power consumption

- In Step 2 we record power consumption traces for multiple random inputs
- Generate randomly n 8-bit inputs. Typically n is large (thousands to millions!)
- Store the inputs in vector $\mathbf{in} = [in_1\, in_2\, in_3 \ldots in_n]^\intercal$

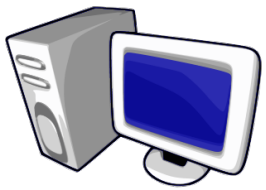


CROCS
Centre for Research on Cryptography and Security

CPA Step 2: Measure the power consumption

- For every generated 8-bit input we measure the power consumption of the AES implementation over time
- For every input $in_j, j = 1, \ldots, n$ we capture a digitized trace over time
- We denote the trace related to input in_j as $\mathbf{t_j} = [t_j^1\, t_j^2 \ldots t_j^m]^\intercal$. It contains m points in time (a.k.a. samples)

15mV	30.7mV	...	82.1mV

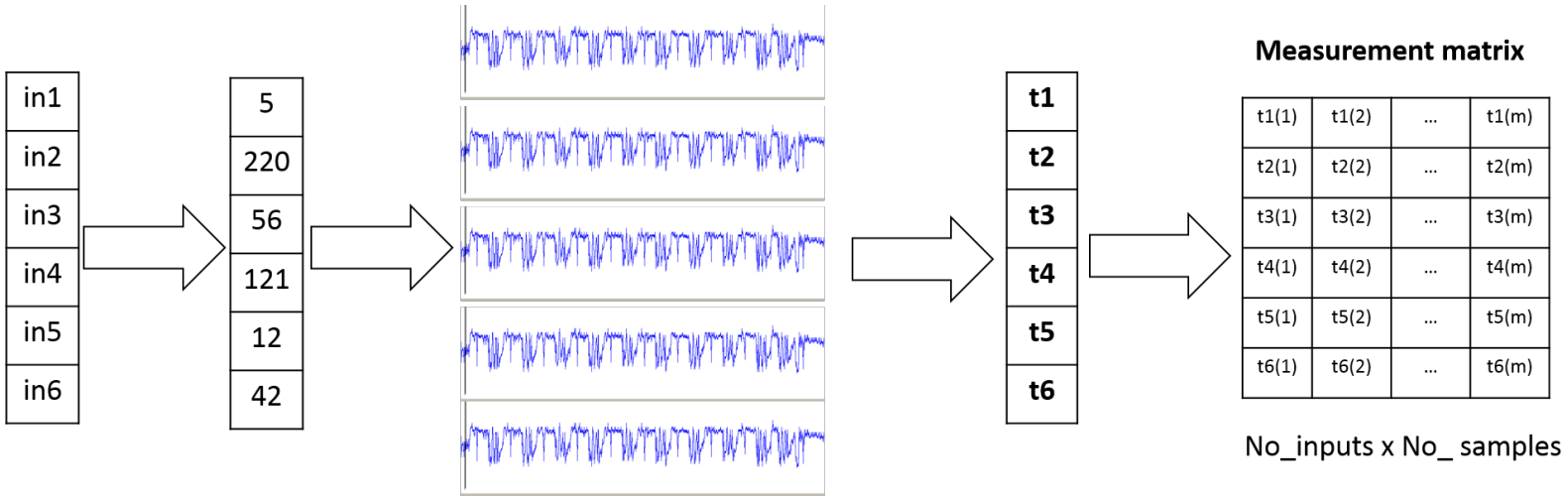


t[j]

CROCS
Centre for Research on Cryptography and Security

CPA Step 2: Measure the power consumption

- Capturing 6 power traces with *m* time points (samples) results in the following measurement matrix

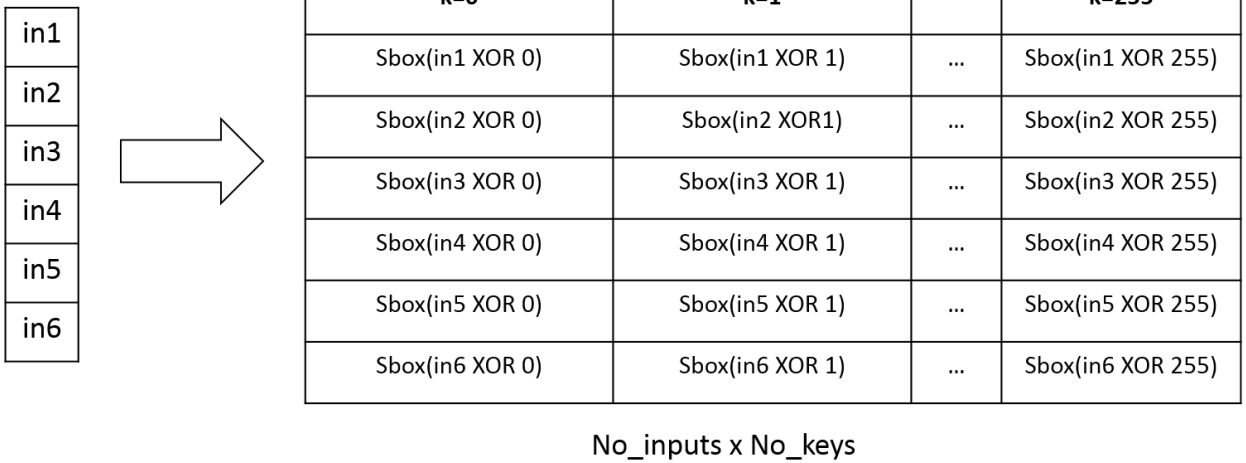


- Note that the power traces originate from the device, i.e. they are related to the secret key stored inside the device
- We will refer to the unknown key that is stored in the device as k_{dev}

CRoCS
Centre for Research on Cryptography and Security

CPA Step 3: Predict (hypothetical) intermediate values

- In our device $y = Sbox(in \oplus k_{dev})$, but k_{dev} is unknown!
- For a given input in we can compute the value y for all possible keys $k \in \{0, 1, \ldots, 255\}$
- ForAll $in \in$ **in**
  ForAll $k \in \{0, 1, \ldots, 255\}$
  Compute $y(in, k) = Sbox(in \oplus k)$



- One of the columns of this value-prediction matrix is the correct one!
- Divide and Conquer strategy

CROCS
Centre for Research on Cryptography and Security

CPA Step 4: Leakage model

- We map the hypothetical intermediate values to hypothetical power consumption values, producing the power-prediction matrix

Power-prediction matrix

k=0	k=1		k=255
HW(Sbox(in1 XOR 0))	HW(Sbox(in1 XOR 1))	...	HW(Sbox(in1 XOR 255))
HW(Sbox(in2 XOR 0))	HW(Sbox(in2 XOR1))	...	HW(Sbox(in2 XOR 255))
HW(Sbox(in3 XOR 0))	HW(Sbox(in3 XOR 1))	...	HW(Sbox(in3 XOR 255)
HW(Sbox(in4 XOR 0))	HW(Sbox(in4 XOR 1))	...	HW(Sbox(in4 XOR 255))
HW(Sbox(in5 XOR 0))	HW(Sbox(in5 XOR 1))	...	HW(Sbox(in5 XOR 255))
HW(Sbox(in6 XOR 0))	HW(Sbox(in6 XOR 1))	...	HW(Sbox(in6 XOR 255))

No_inputs x No_keys

- A common choice is Hamming weight but keep in mind that other models may be applicable

CRoCS
Centre for Research on Cryptography and Security

CPA Step 5: Comparison

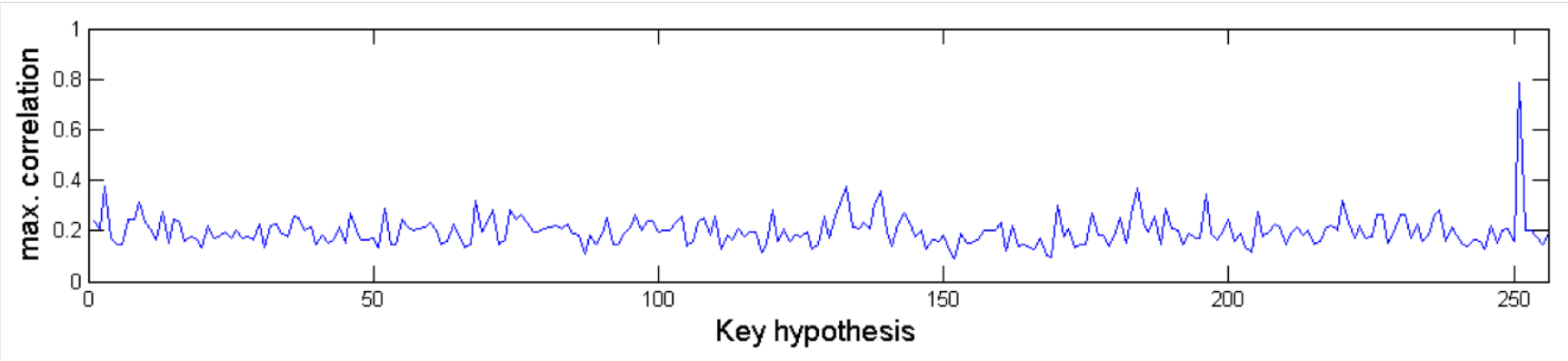
- Compare the hypothetical power consumption values with the real measurements using Pearson correlation
- ForAll columns of measurement matrix
  ForAll columns of power prediction matrix
  Compute the correlation between columns
- The highest correlation value reveals the key

Pearson correlation coefficient $\rho_k(L, HW(V_g))$ for leakage L, Hamming weight of the intermediate value computed $HW(V)$ and key guess K_g :

$$\rho_k(L, HW(V_g)) = \frac{cov[L, HW(V_g)]}{\sqrt{Var[L] \cdot Var[HW(V_g)]}} \tag{1}$$

CROCS
Centre for Research on Cryptography and Security

CPA Step 5: Comparison



Countermeasures

Masking and hiding

The idea

Purpose: break the link between the actual data and power consumption

- Masking: power consumption remains dependent on the data on which computation is performed but not the actual data
  - A random mask concealing every intermediate value
  - Can be on different levels (arithmetic $\rightarrow$ gate level)
- Hiding: power consumption is independent of the intermediate values and of the operations
  - Special logic styles like WDDL, MDPL etc.
  - Randomizing in time domain
  - Lowering SNR ratio

CROCS
Centre for Research on Cryptography and Security

Software countermeasures

- Time randomization:
  - Operations are randomly shifted in time
  - Use of NOP operations
  - Add random delays
  - Use of dummy variables and instructions (sequence scrambling)
- Register renaming and nondeterministic processor
  - Idea is to exploit ILP within an instruction stream
  - Processor selects an instruction and a memory access randomly
- Permuted execution
  - rearranged instructions e.g. S-boxes
- Masking techniques

CROCS
Centre for Research on Cryptography and Security

Hardware countermeasures

- Noise generation:
  - Hardware noise generator from e.g. RNG
  - Total power is increased
- Desynchronization:
  - Introducing some fake clock cycles during the computation or using a weak jitter
- Power signal filtering:
  - ex.: RLC filter (R-resistor, C-capacitor, L-inductor) smoothing the pow. cons. signal by removing high frequency components
  - Using active comp. (transistors) in order to keep pow. cons. relatively constant
- Novel circuit designs e.g. special logic styles

CROCS
Centre for Research on Cryptography and Security

DFA

- Bellcore attack in 1995
  - Differential faults on RSA-CRT signatures
  - Requires 1 correct and 1 wrong signature
- Attack on DES in 1997 by Biham and Shamir
- Special attacks on AES, ECC etc.

CROCS
Centre for Research on Cryptography and Security

DFA on symmetric-key ciphers

- Basic DFA scenario:
  - adversary obtains a pair of ciphertexts that are derived by encrypting the same plaintext (one is correct value and the other is faulty)
  - two encryptions are identical up to the point where the fault occurred
  - $\rightarrow$ two ciphertexts can be regarded as the outputs of a reduced-round iterated block cipher where the inputs are unknown but show a small (and possibly known) differential
- DFA on DES
  - the original attack of Biham and Shamir exploits computational errors occurring in the final rounds of the cipher
  - assumes that one bit of the right half of the DES internal state is flipped at a random position

CROCS
Centre for Research on Cryptography and Security

Countermeasures

Generic approaches

- Correctness check: encrypt twice
- Random delays: limits the precision
- Masking: secret sharing complicates probing wires of the device

Hardware countermeasures:

- light detectors, supply voltage, frequency detectors
- active shields
- redundancy: duplication of hardware blocks
- dual rail implementations

CROCS
Centre for Research on Cryptography and Security

Symmetric-key countermeasures

- Introducing redundancy is harder than for PKC
- Multiple execution is expensive
- Using the inverse
- Loop invariant:
  - 2nd variable counting in the opposite way prevents tampering the counter of a loop
  - add a signature that is updated in every run of the loop (checksum)
- To ensure the integrity of the stored data, Cyclic Redundancy Check (CRC) can be added

CROCS
Centre for Research on Cryptography and Security

RSA cryptosystem

- Key generation:
  - e and the length of n are given
  - Generate "large" prime numbers p, q such that $\gcd(p-1, e) = 1$, $\gcd(q-1, e) = 1$;
  - Compute $n = p \cdot q$ and $\varphi(n) = (p-1)(q-1)$;
  - Compute d satisfying $ed \equiv 1 \pmod{\varphi(n)}$
  - Public key: $p_k = (n, e)$; private key: $s_k = (n, d)$.
- Public-key op. $c \leftarrow \mathrm{Enc}(pk, m)$: $c := m^e \pmod{n}$.
- Secret-key op. $m \leftarrow \mathrm{Dec}(sk, c)$: $m := c^d \pmod{n}$.

- There are further limitations to the choice of p, q.
- e is typically fixed in advance (e.g., $65537 = 2^{16} + 1$).

- CRT
  - Find $d_p = d \pmod{p-1}$, computed as $d_p = e^{-1} \pmod{p-1}$
  - Find $d_q = d \pmod{q-1}$
  - Compute $U = p^{-1} \pmod{q}$
  - $c_p = c \pmod{p}$, $c_q = c \pmod{q}$

CROCS
Centre for Research on Cryptography and Security

RSA with CRT

Optimisation of computing a signature giving about 4-fold speedup:
$n = p \cdot q$ Signature: $s = m^d \mod n$

Precomputed values $d_p := d \mod (p-1)$ $d_q := d \mod (q-1)$
$i_q := q^{-1} \mod p$

$s_p := m^{d_p} \mod p$ $s_q := m^{d_q} \mod q$

Garner's method (1965) to recombine s_p and s_q:
$s = s_q + q \cdot (i_q(s_p - s_q) \mod p)$

CROCS
Centre for Research on Cryptography and Security

Injecting fault in branch s_p

- Assume that an adversary can inject a *fault* in the computation of s_p, resulting in $\hat{s}_p$.
- Moreover, an invalid signature $\hat{s}$.
- The adversary can make a correct and an incorrect signatures:
  $s = s_q + q \cdot (i_q(s_p - s_q) \mod p)$
  $\hat{s} = s_q + q \cdot (i_q(\hat{s}_p - s_q) \mod p)$
  $s - \hat{s} = q \cdot \big((i_q(s_p - s_q) \mod p) - (i_q(\hat{s}_p - s_q) \mod p)\big)$.
- Then the adversary can recover q as follows: $\boxed{q = \gcd(n, s - \hat{s})}$

CROCS
Centre for Research on Cryptography and Security

Injecting fault in branch s_q

- Assume that an adversary can inject a *fault* in the computation of s_q, resulting in $\hat{s}_q$.
- The adversary can make a correct and an incorrect signatures:
  $s = s_q + q \cdot (i_q(s_p - s_q) \mod p)$
  $\hat{s} = \hat{s}_q + q \cdot (i_q(s_p - \hat{s}_q) \mod p)$
- Subtracting the 2 signatures:

$$\begin{aligned} s - \hat{s} &\equiv (s_q - \hat{s}_q) + q \cdot (i_q(s_p - s_q) \mod p) \\ &\quad - q \cdot (i_q(s_p - \hat{s}_q) \mod p) \\ &\equiv (s_q - \hat{s}_q) + (q \cdot i_q \mod p)(s_p - s_p \mod p) \\ &\quad - \underbrace{(q \cdot i_q \mod p)}_{\equiv 1 \pmod p}(s_q - \hat{s}_q \mod p) \\ &\equiv 0 \pmod p. \quad \Longrightarrow \quad \boxed{p = \gcd(n, s - \hat{s})} \end{aligned}$$

CROCS
Centre for Research on Cryptography and Security

Countermeasure for RSA

- Compute a signature twice and compare the two results
- Verify the signature with the public exponent e, but in some applications (Java card), one does not have access to the public exponent e during signature generation
- Shamir: random r is first chosen and then modular exp. is computed based on $r \cdot n$

- Find $s^* = m^d \pmod{r \cdot n}$
- Find $Z = m^d \pmod{r}$
- If $s^* = Z \pmod{r}$
- Output $s = s^* \pmod{n}$

CROCS
Centre for Research on Cryptography and Security

Conclusions

- The goal of this lecture is to go more into detail of Side-Channel and Fault Injection Analyses by presenting in detail the two most powerful techniques against cryptographic implementations.
  - What are the assumptions of CPA?
  - What are the assumptions of DFA on RSA?
- Provide more background on Side-Channel Attacks
- Thank you for the attendance :-)

CROCS
Centre for Research on Cryptography and Security

Questions

?

CROCS
Centre for Research on Cryptography and Security