



# Side-channel attacks

## Cryptoanalysis



UNIVERSITÀ DI PISA

- **Cryptanalysis** is the art and science of analyzing information systems in order to study the hidden aspects of the systems
- **Mathematical analysis** of cryptographic algorithms
- **Side Channel Attacks**

## What is a side channel?



UNIVERSITÀ DI PISA

- A side channel is based on information gained from the **physical implementation** of a cryptosystem
- No theoretical weaknesses in the algorithm
- No brute force

09/05/2018

Side-channel attacks

3

## Side channel attacks



UNIVERSITÀ DI PISA

- The attacker must **have physical access** to the device under attack
- The attacker **knows the algorithm** under attack
  - The only secret is the key
- Two stages
  - 1st stage → Measurements
  - 2nd stage → Analysis of the measurements
    - Statistical analysis
    - Application of cryptanalysis

09/05/2018

Side-channel attacks

4

## Types of side channel attacks



UNIVERSITÀ DI PISA

- Fault injection
- Power analysis
- Timing analysis

09/05/2018

Side-channel attacks

5

## Why side channels?



UNIVERSITÀ DI PISA

- More effective against modern cryptosystem
- Embedded systems change the threat model
  - The adversary may physically attack the system
    - E.g.: smart meter, electronic passports, identity cards, driver licenses, point of sales, digital rights management, access control, pay tv, etc etc
  - The adversary physically interfere with the system
  - The adversary has a scale advantage
    - Ex. Extracting one key from a single Pay TV smartcard allows to program several new smartcards with the same key  
→ **clones**

09/05/2018

Side-channel attacks

6

Side channel attacks

# PHYSICAL ATTACKS

09/05/2018

Side-channel attacks

7

## CRT and RSA optimization



UNIVERSITÀ DI PISA

- **Chinese Remainder Theorem** allows us to compute RSA (decryption, signing) more efficiently
- **Problem:** Efficiently compute  $y = x^d \pmod{n}$ 
  1. Compute  $x_p = x \pmod{p}$  and  $x_q = x \pmod{q}$
  2. Compute  $y_p = x_p^{d \pmod{p-1}} \pmod{p}$  and  $y_q = x_q^{d \pmod{q-1}} \pmod{q}$
  3. Compute  $y = a_p y_p q + a_q y_q p$  where  $a_p$  and  $a_q$  are properly (pre-)computed coefficients

09/05/2018

Side-channel attacks

9

## CRT and RSA optimization



UNIVERSITÀ DI PISA

- **Performance advantage**

- Computation of  $y_p$  and  $y_q$  is the most demanding
- It requires  $\#MUL+\#SQ = 1.5t$ , on average
  - 2 exponentiations on  $t/2$  bits  $\rightarrow 2 \times (1.5 t/2) = 1.5t$
- Each squaring/multiplication involves  $t/2$ -bit operands
  - So, multiplication/squaring takes  $O(t^2/4)$
- Thus **the total speedup obtained through CRT is a factor of 4.**

09/05/2018

Side-channel attacks

10

## A fault-injection attack against CRT-based RSA



UNIVERSITÀ DI PISA

- **Attack intuition:** by injecting a fault the adversary is able to factorize  $n$
- **The attack**
  - Cause an **hw fault** while computing  $y_p$  which produces  $y'_p$  and thus  $y' = a_p y'_p q + a_q y_q p$
  - It follows that  $y - y' = a_p(m'_p - m_p)q$
  - Thus,  $\gcd(y - y', n) = q$  which can be efficiently computed with the Euclidean algorithm
- **Practical considerations**
  - **causing hw fault:** tamper with computing circuitry
  - **countermeasures:** checking results (10% slow down)

09/05/2018

Side-channel attacks

11

Side channel attacks

# POWER ANALYSIS

09/05/2018

Side-channel attacks

12

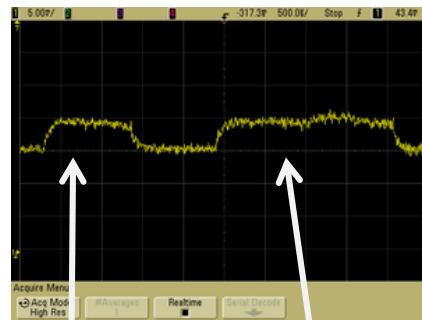
## Power Analysis



UNIVERSITÀ DI PISA

- Power analysis is a **side channel** attack in which the attacker studies the power consumption of a cryptographic hardware device
  - smart card, tamper-resistant "black box", or integrated circuit
- The attack is non-invasive
- **Simple power analysis (SPA)** involves *visual examination* of graphs of the current used by a device over time.
  - Variations in power consumption occur as the device performs different operations.

### Power Analysis of RSA



**Key bit = 0**  
**No multiplication**


**Key bit = 1**  
**multiplication**

09/05/2018

Side-channel attacks

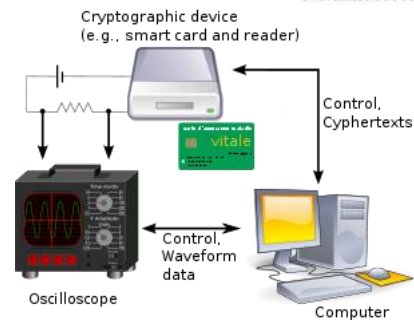
13

# Power Analysis



UNIVERSITÀ DI PISA

- **Differential power analysis (DPA)** involves statistically analyzing power consumption measurements from a cryptosystem.
  - DPA attacks have signal processing and error correction properties which can extract secrets from measurements which contain too much noise to be analyzed using simple power analysis.



Oscilloscope      Computer

09/05/2018
Side-channel attacks
14

Side channel attacks

# TIMING ATTACKS

09/05/2018
Side-channel attacks
15

## Timing attack



UNIVERSITÀ DI PISA

- A **timing attack** is a **side channel** attack in which the attacker attempts to compromise a cryptosystem by analyzing the time taken to execute cryptographic algorithms
  - Execution time depends on inputs (e.g., key!)
  - Require precise measurement of time
  - Attack is application dependent
  - E.g., square-and-multiply for  $\text{exp mod } n$ 
    - time depends on number of “1” in the key
    - Statistical analysis of timings with same key and different inputs

09/05/2018

Side-channel attacks

16

## Timing Attack against HMAC



UNIVERSITÀ DI PISA

- Example<sup>(\*)</sup>: George Keyczar crypto library (Python, Java) [simplified]

```
def Verify(key, msg, tag):
    return HMAC(key, msg) == tag
```

- **The problem:** ‘==’ implemented as a byte-by-byte comparison
- Comparator returns false when first inequality found
- This provides a timing side-channel

<sup>(\*)</sup> N.Lawson, Side-Channel Attacks on Cryptographic Software, IEEE Security & Privacy, 2009


09/05/2018

Side-channel attacks


17



## Timing attack


  
 UNIVERSITÀ DI PISA

target  
msg **m**



$\xrightarrow{\text{m, tag}}$

$\xleftarrow{\text{accept or reject}}$



**k**

**Timing attack:** to compute tag **tag** for target message **m** do:


- Step 1: Query server with random tag
- Step 2: Loop over all possible first bytes and query server.  
stop when verification takes a little longer than in step 1
- Step 3: repeat for all tag bytes until valid tag found

trial tag

3	47	*	*	*	*
---	----	---	---	---	---

09/05/2018
Side-channel attacks
18

## Defense #1

  
 UNIVERSITÀ DI PISA

**Make string comparator always take same time**  
(Python) :

```

return false if tag has wrong length
result = 0
for x, y in zip( HMAC(key,msg) , tag):
    result |= ord(x) ^ ord(y)
return result == 0
  
```

***Can be difficult to ensure due to optimizing compiler***

09/05/2018
Side-channel attacks
19

## Defense #2



UNIVERSITÀ DI PISA

**Make string comparator always take same time**  
(Python) :

```
def Verify(key, msg, tag):  
    mac = HMAC(key, msg)  
    return HMAC(key, mac) == HMAC(key, tag)
```

***Attacker doesn't know values being compared!***