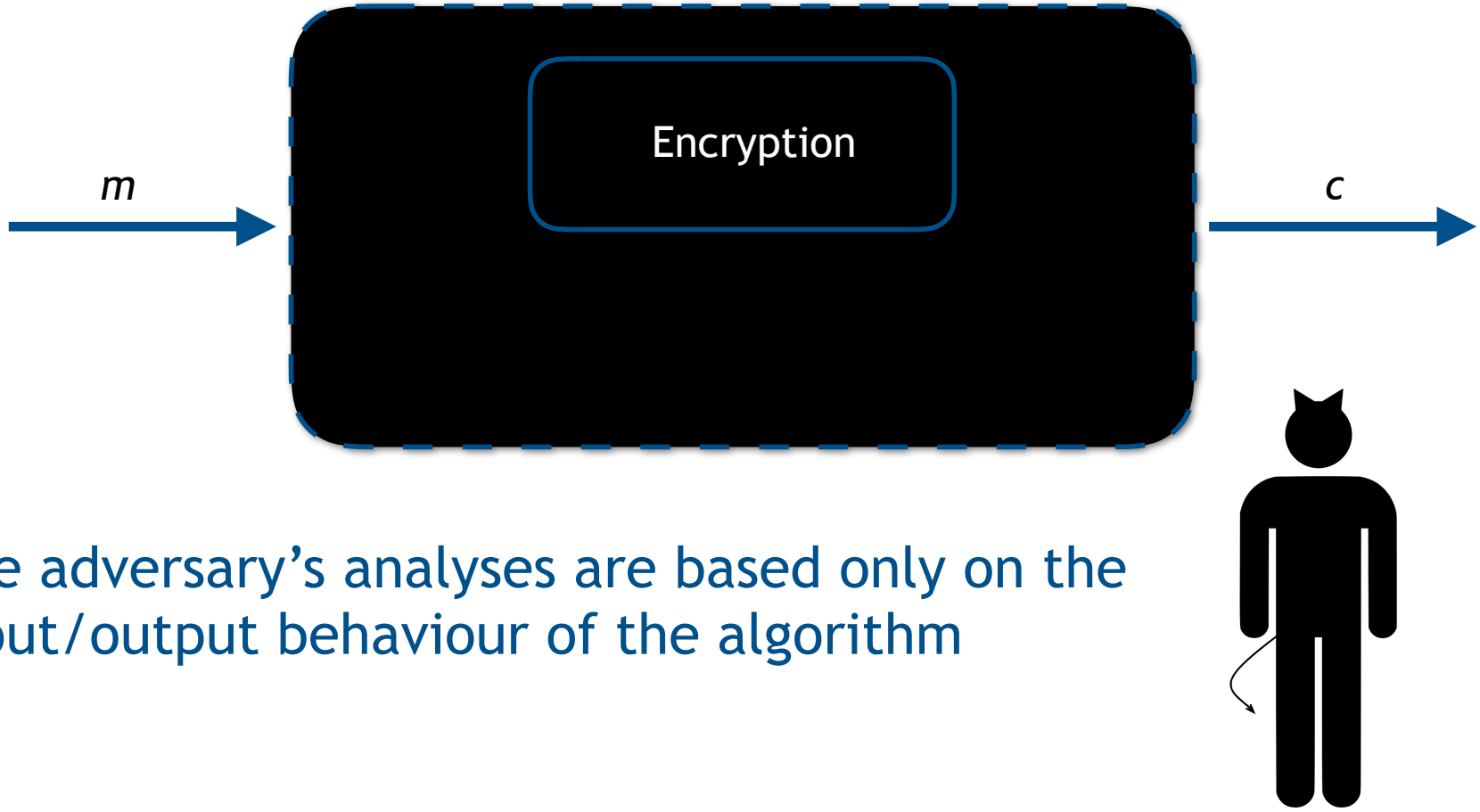
Grey- and White-box Attack Scenarios

Aalto University

Black-box attack scenario



The adversary's analyses are based only on the input/output behaviour of the algorithm

Black box security of cryptographic algorithms

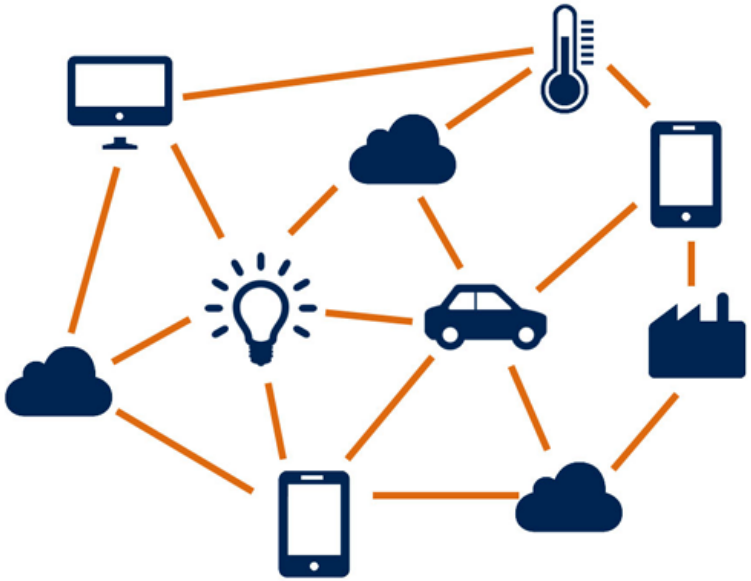
- Public-key cryptography
  - Standardised approaches: Diffie-Hellman key exchange, the RSA cryptosystem and Elliptic Curve Cryptography
  - Security relies on hard problems: *discrete logarithm problem*, *factorisation problem*, *ECDL problem*

- Symmetric key cryptography
  - Standardised algorithms such as AES are secure against known cryptanalysis techniques (diff. and linear cryptanalysis [1])
  - AES has been around for many years now and remains unbroken

[1] C. Blondeau, G. Leander, K. Nyberg: Differential-Linear Cryptanalysis Revisited. J. of Cryptology

Cryptographic implementations

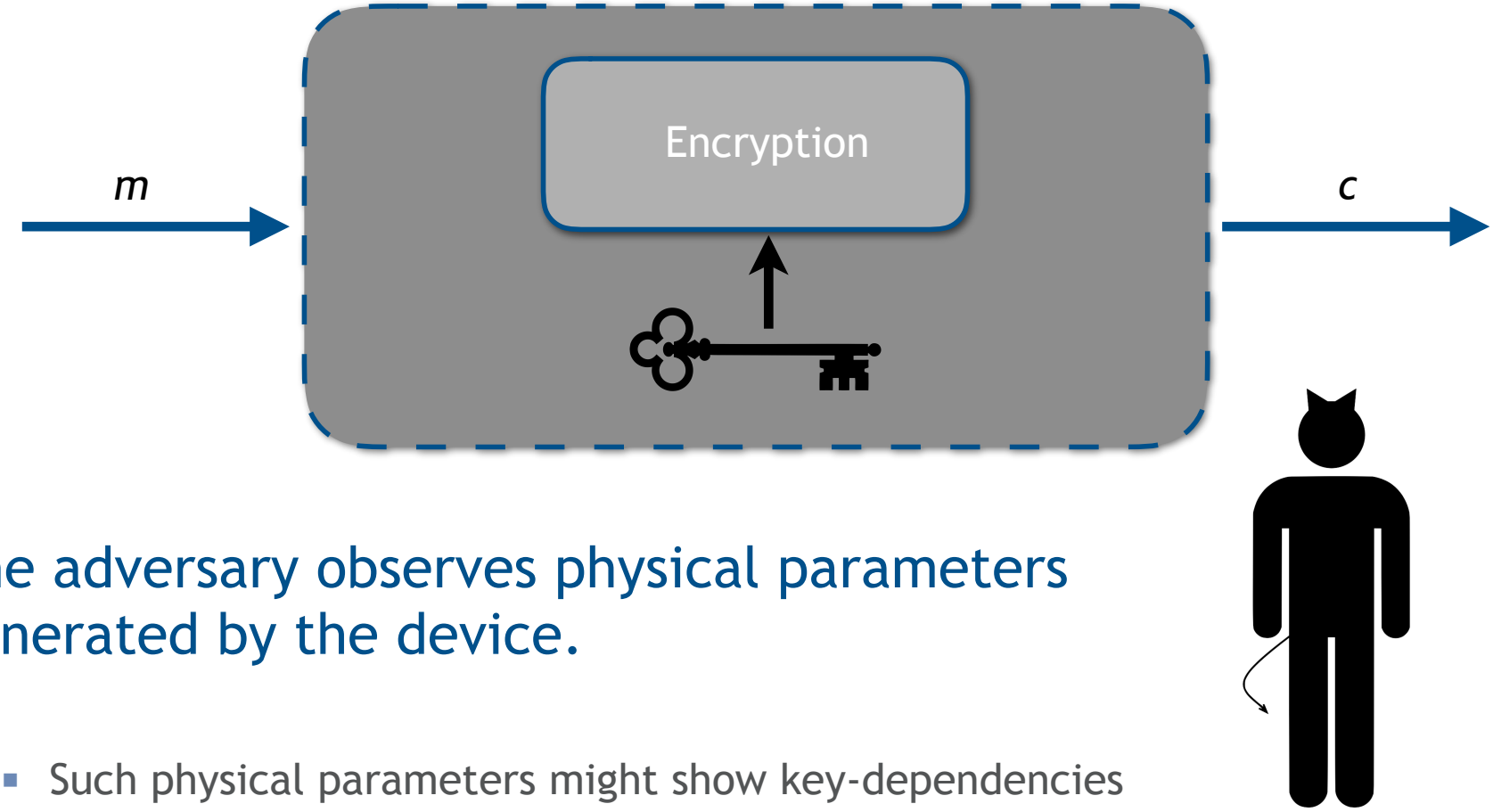
- In practice, cryptographic algorithms are implemented within hardware devices or software applications
- Depending the use case, such devices or applications might be accessible to the adversaries



- We consider thus two further attack scenarios: grey-box and white-box

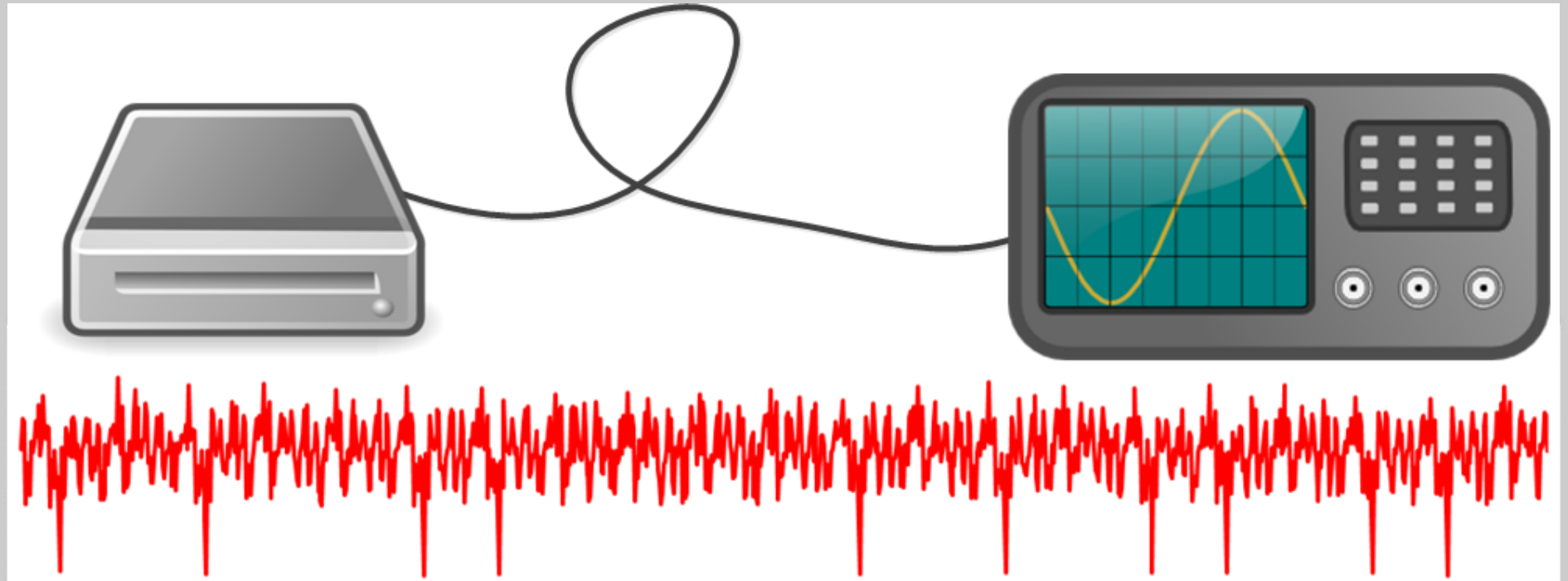
[http://thehackernews.com/2015/08/hacking-internet-of-things-drone.html]

Grey-box attack scenario



The adversary observes physical parameters generated by the device.

- Such physical parameters might show key-dependencies
- The adversary can perform statistical analyses on the parameters
- The adversary can also modify the hardware



Side Channel Analysis Attacks

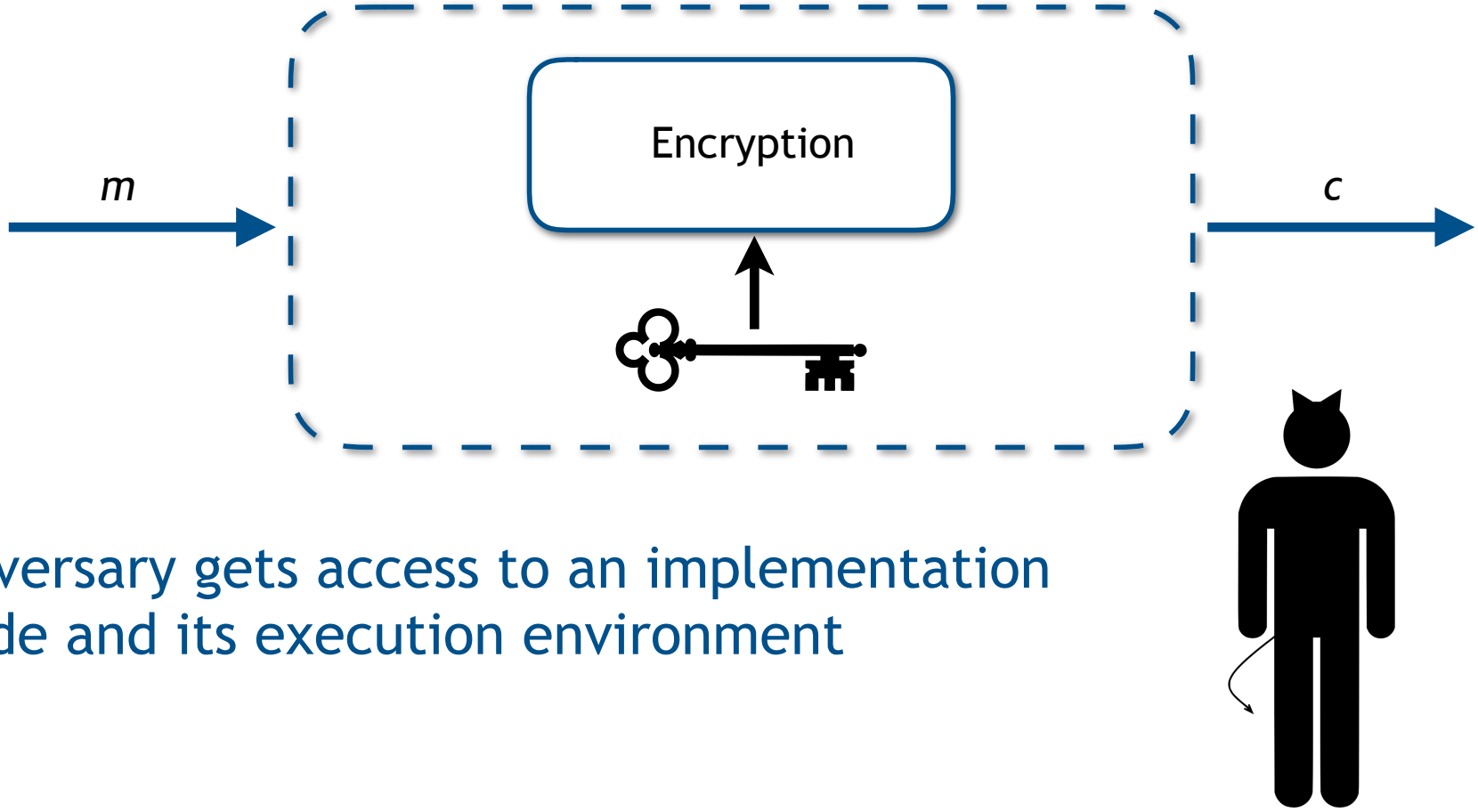
SCA attacks are *passive* physical attacks based on the observation of a hardware device during the execution of cryptographic operations

Parameters observed:

- Power consumption (power analysis)
- Electromagnetic radiation (EM analysis)
- Execution times (timing attacks)
- …

Active attacks influence the behaviour of the device under attack
e.g. via fault injection

White-box attack scenario



Adversary gets access to an implementation code and its execution environment

→ WB Cryptography aims to provide security even under such attack threats

Outline

- Side-channel analysis
  - On example of Elliptic Curve Cryptography (ECC)
  - Simple Power Analysis, Differential Power Analysis
  - Further types of SCA

- White-Box Cryptography
  - From practice to theory
  - Attacks and challenges

Side Channel Analysis

Elliptic Curve Cryptography

ECC is based on elliptic curves (*E*) over Galois Fields (*GF*) (finite fields)

$$E : y^2 = x^3 + ax + b \text{ mod } p, \text{ where } a, b \in \mathbb{Z}_p$$

- All points *P=(x,y)* satisfying this formula are part of the curve, i.e. part of the group.

- Group operations can be performed between the points
  - **point addition** *P* + *Q* and **point doubling** *2P*

Elliptic Curve Cryptography

Additionally, we define a neutral element, also known as point at infinity such that

$$P + \mathcal{O} = P$$

- An operation between two points on the curve always results in another point in the curve

$$P + Q = R$$

$$(x_1, y_1) + (x_2, y_2) = (x_3, y_3)$$

Elliptic Curve Cryptography

The point addition and point doubling are the basis for the elliptic curve point multiplication:

$$kP = \underbrace{P + P + \ldots + P}_{k \text{ times}}$$

The security of ECC is based on the elliptic curve discrete logarithm problem: given P and R such that *R=kP*, it is difficult to determine *k*

—> Workshop on Elliptic Curve Cryptography

Role of the Scalar Multiplication on ECC

Elliptic Curve Diffie-Hellman (ECDH) – used for stablishing a secret over an insecure channel.

- The secret can be used for performing further encryption and decryption operations during the communication
- Implemented in TLS 1.3.

EC point operations can also be used for performing encryption and decryption in ECC

- More than 90% of each operation consist of the scalar multiplication
- In practice, these operations are rather performed using symmetric approaches

Double-and-add algorithm

Input: $k = (k_{l-1}, ..., k_1, k_0)_2, P \in E(GF(2^m))$.

Output: kP.

$Q \leftarrow O$.

for i from $l-1$ downto 0 **do**

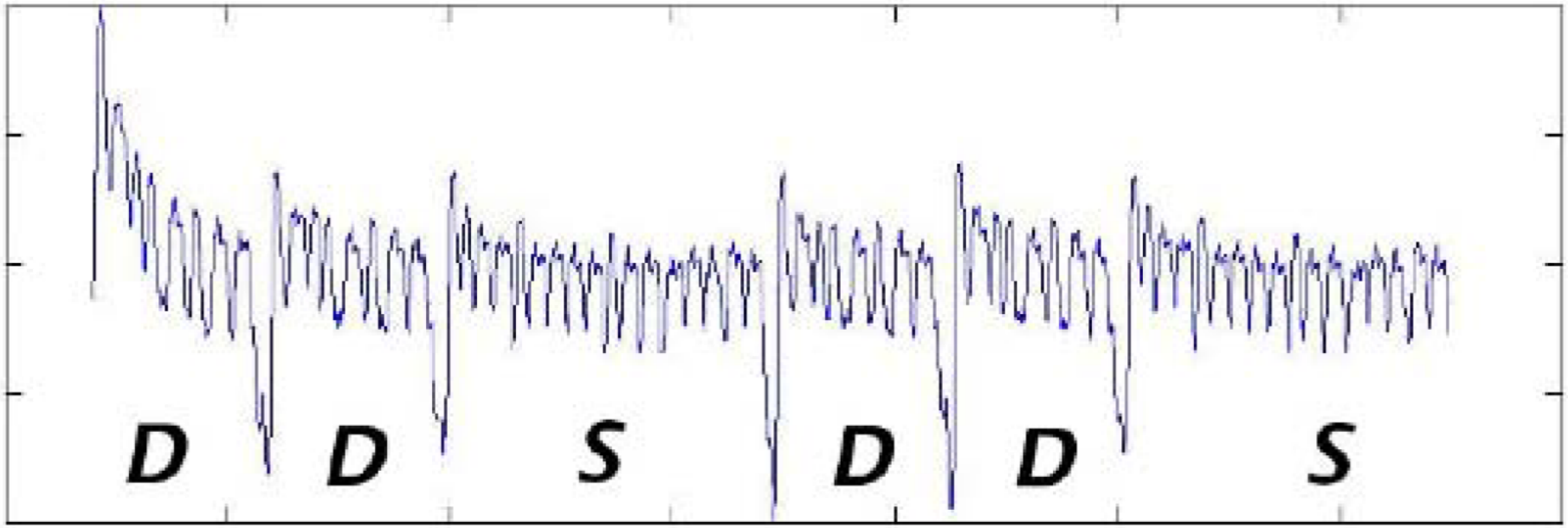
$Q \leftarrow 2Q$.

If $k_i = 1$ then $Q \leftarrow Q + P$.

end for

return Q.

Vulnerable to Side Channel Analysis Attacks



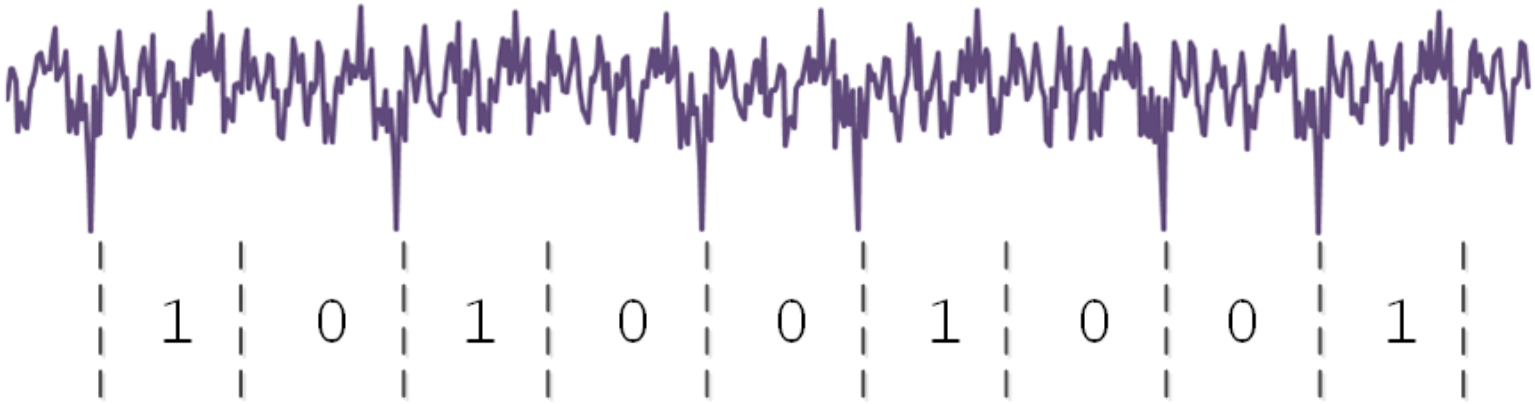
0 1 0 1 ...

[D. Hankerson –Guide to Elliptic Curve Cryptography]

Simple Power Analysis

Implementations of the double-and-add algorithm can be easily attacked using Simple Power Analysis (SPA)

- SPA is performed with simple eye observations
- Only one measurement is needed to extract the value of the key



So how can we fix the double-and-add algorithm?

How can we make it SPA resistant?

Input: $k = (k_{l-1}, ..., k_1, k_0)_2, P \in E(GF(2^m))$.

Output: kP.

$Q \leftarrow O$.

for i from $l-1$ downto 0 **do**

$Q \leftarrow 2Q$.

If $k_i = 1$ then $Q \leftarrow Q + P$.

end for

return Q.

Countermeasures against SPA

Countermeasures against SPA are based on *balance:* always performing the same operations, independently of the value of the key bit being processed.

- Double-and-add-always algorithm

Input: $k = (k_{l-1}, ..., k_1, k_0)_2, P \in E(GF(2^m))$.

Output: kP.

$Q[0] \leftarrow O$.

for i from $l-1$ downto 0 **do**

$Q[0] \leftarrow 2Q$, $Q[1] \leftarrow Q[0] + P$.

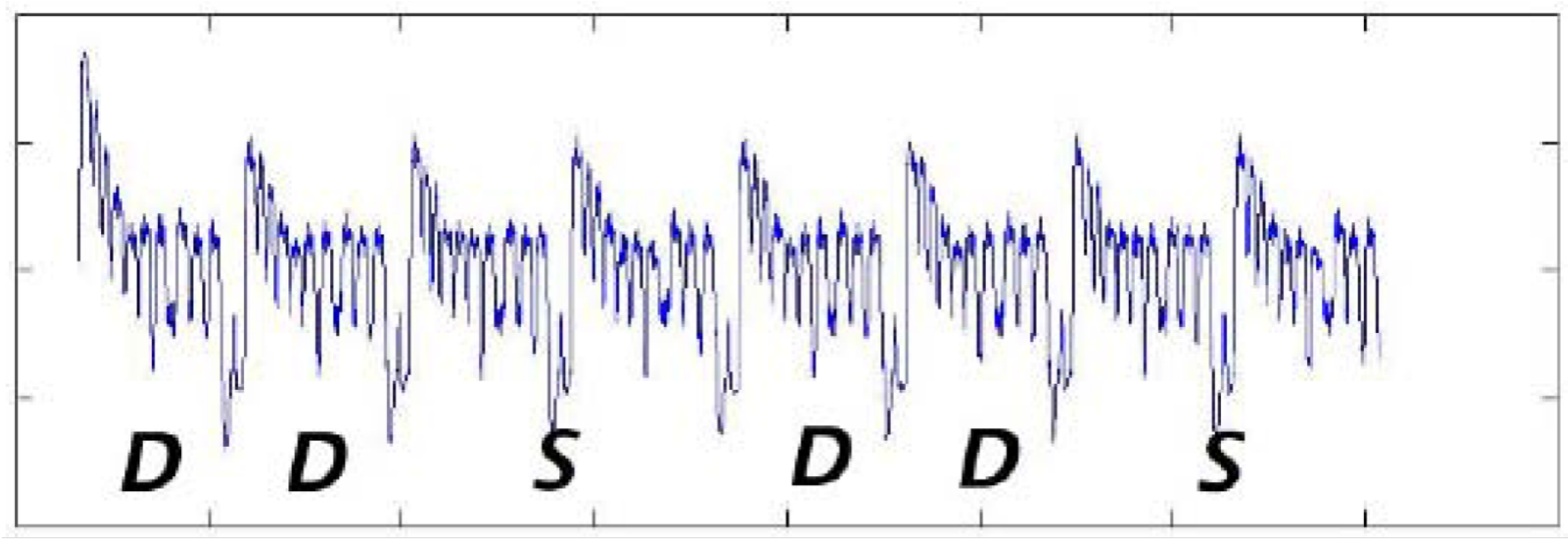
$Q[0] \leftarrow Q[k_i]$.

end for

return $Q[0]$.

Countermeasures against SPA (2)

- Indistinguishable EC point operations: all operations performed by the algorithm consist of the same arithmetic operaitons.
- Alternative regarding the double-and-add-algorithm: implement each point addition as two doubling operations.



[D. Hankerson –Guide to Elliptic Curve Cryptography]

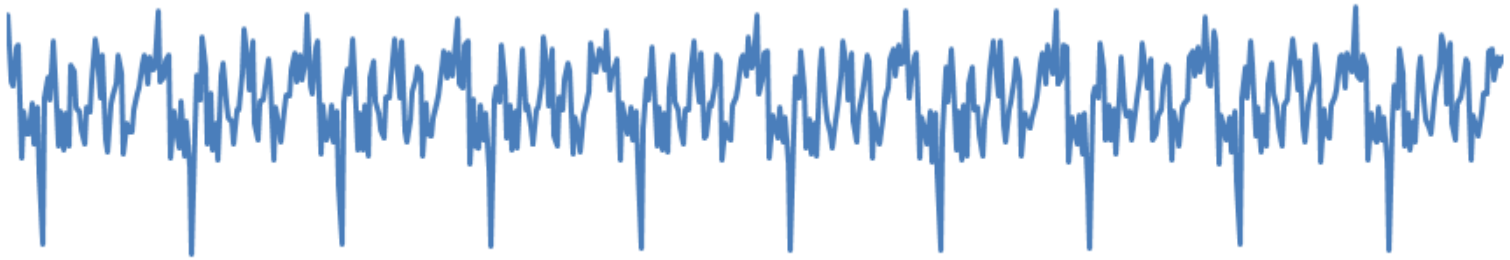
Countermeasures against SPA: Montgomery Ladder

- Balanced and more efficient algorithms can be implemented as a countermeasure against SPA, for example the Montgomery Ladder

Input: $k = (k_{l-1}, ..., k_1, k_0)_2$ with $k_{l-1} = 1, P = (x, y) \in E(GF(2^m))$.
Output: kP.

$Q[0] \leftarrow P$, $Q[1] \leftarrow 2P$.
for i from $l-2$ downto 0 **do**
  if $k_i = 1$ **then**
    $Q[0] \leftarrow Q[0] + Q[1]$, $Q[1] \leftarrow 2Q[1]$.
  else
    $Q[1] \leftarrow Q[0] + Q[1]$, $Q[0] \leftarrow 2Q[0]$.
  end if
end for
return $Q[0]$.

Power trace of an implementation of the Montgomery Ladder

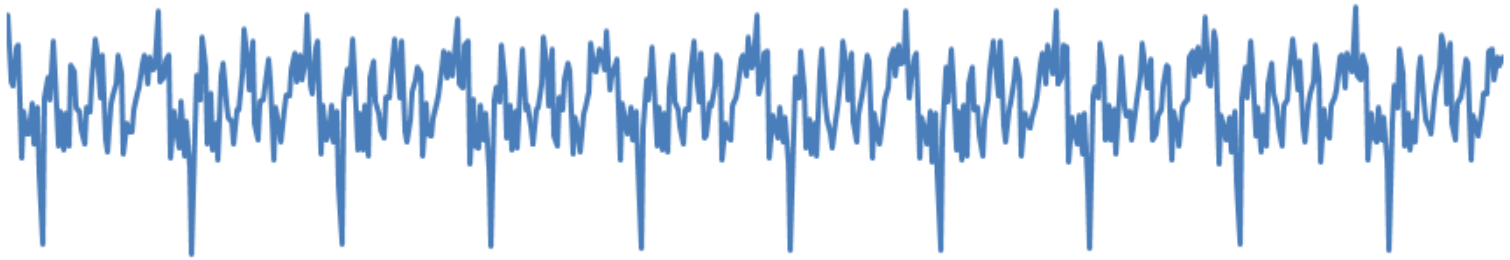


- No differences can be observed between the slots
- This implementation cannot be attacked using SPA
- Statistical methods can still be applied to find differences between the slots in the power trace and perform a key extraction

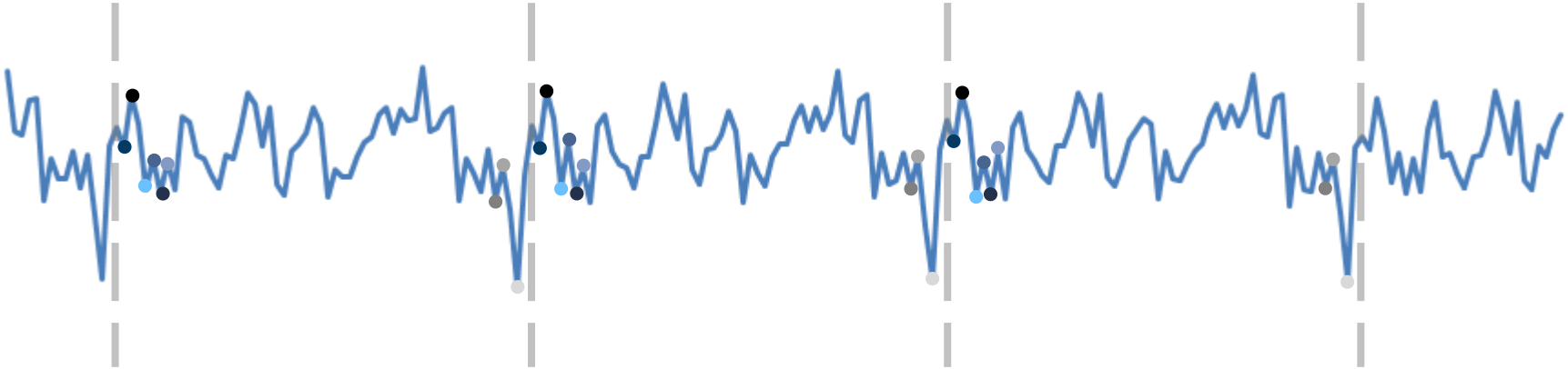
Differential Power Analysis

Differential Power Analysis (DPA) applies statistical means in order to perform a key extraction

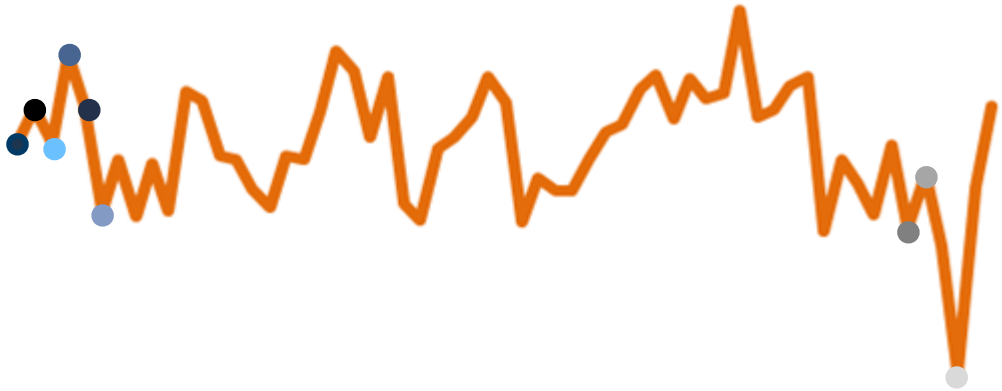
- Applies statistical methods on a big number of measurements. Many power traces are used (or many slots of one power trace).
- Exploits „small“ key dependencies
- The statistical methods also help reducing noise, which uncovers more details in the power trace.



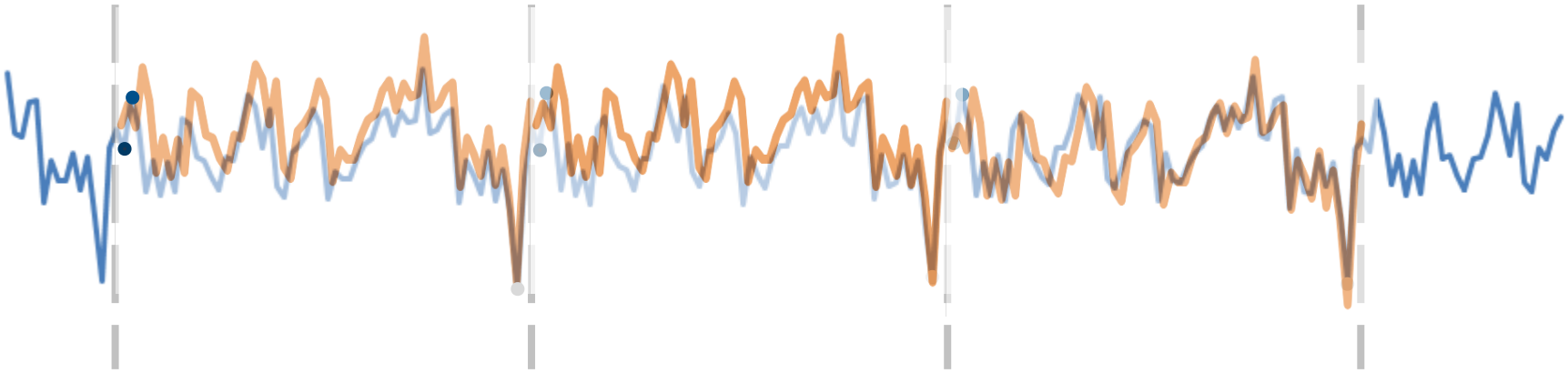
Example of a horizontal DPA attack



Pointwise calculation of the *mean* curve



Example of a horizontal DPA attack: difference-of-means test (2)

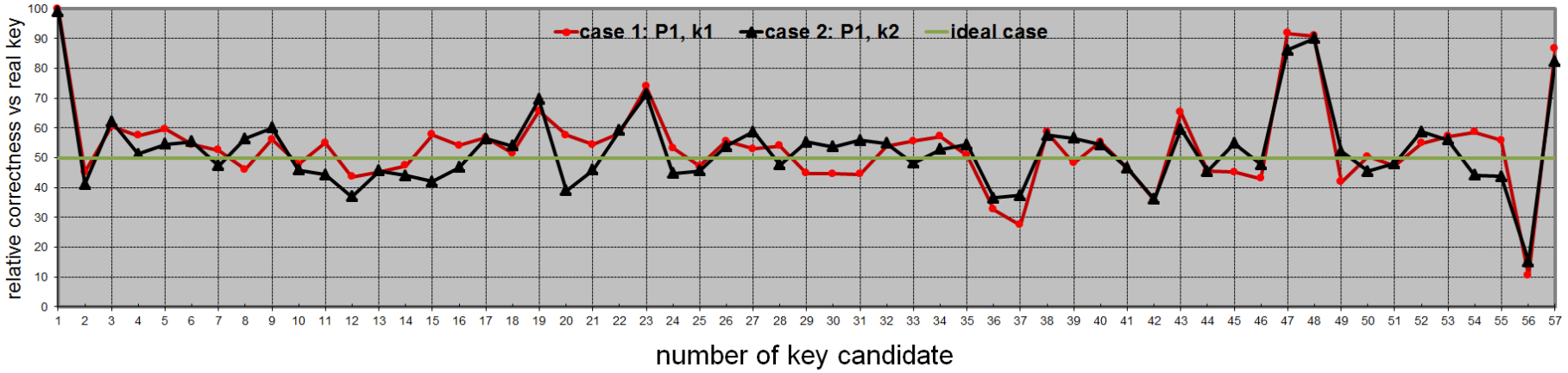


	Point 1	2	…	57	*k*
Slot 1	1	1	…	1	1
Slot 2	1	0	…	0	0
Slot 3	0	0	…	0	0
…	…	…	…	…	…
Slot 231	0	1	…	1	1
Correct extractions	33%	46%	…	100%	

→ Compare

Example of a horizontal DPA attack: difference-of-means test (3)

Graphical representation of the results



Further statistical methods used for performing DPA: are for example the T-test or the correlation power analysis.

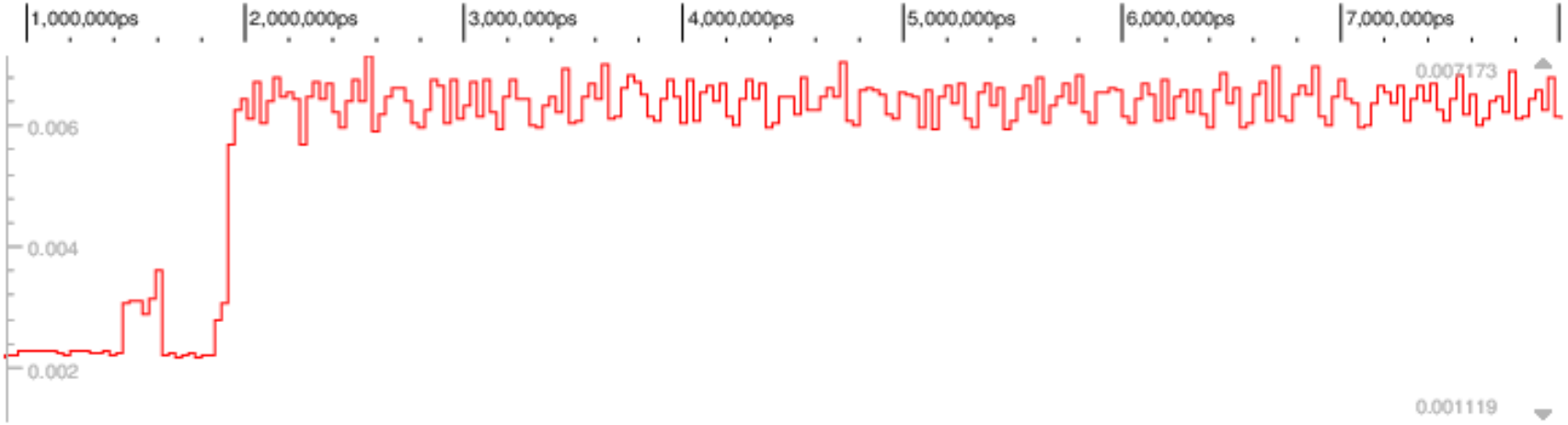
Performing a DPA can also help a designer to evaluate its own work and eventually find security gaps or design errors.

Countermeasures against DPA

Efficient countermeasures against DPA attacks are based on randomization.

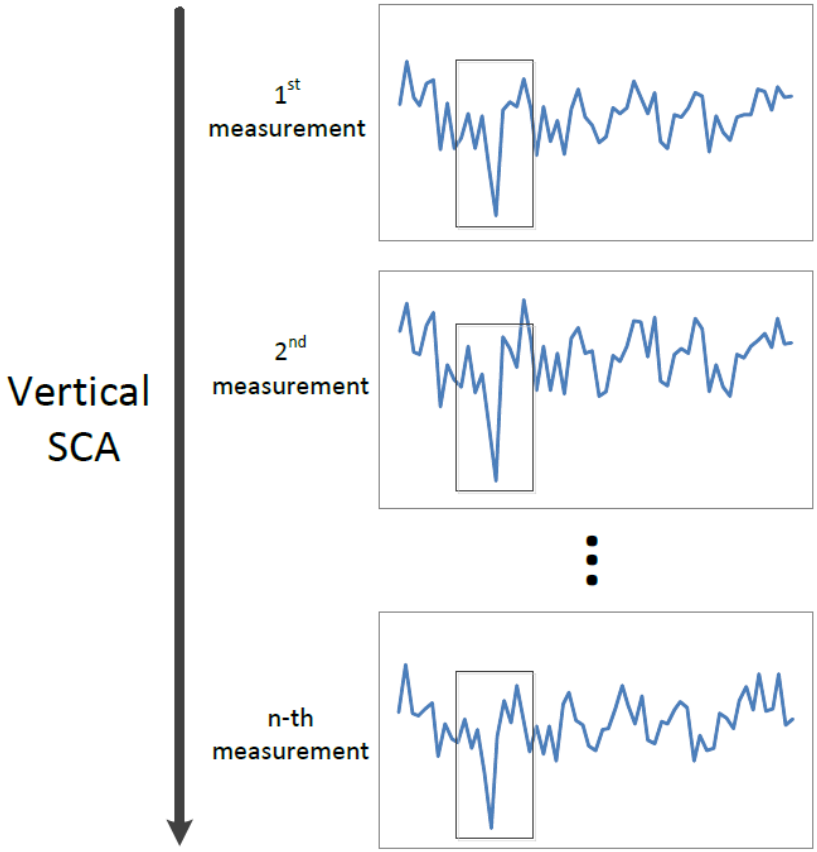
- Key randomization, input data randomization, operation execution sequence randomization, etc.

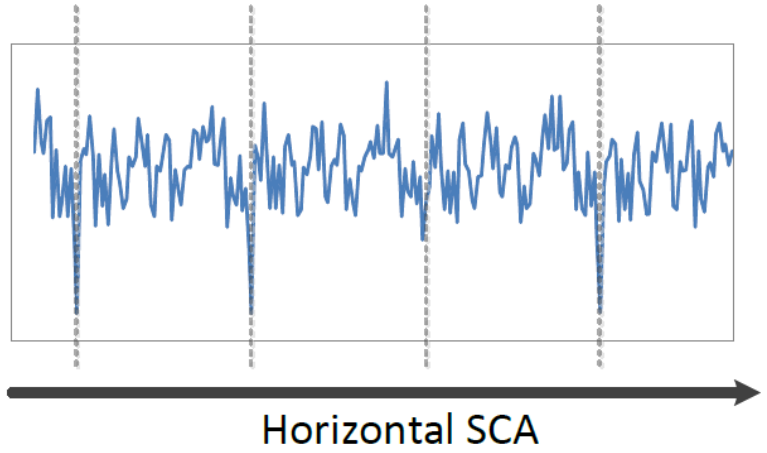
- The implementation of efficient cryptographic algorithms in a *balanced* form can provide protection against selected SCA attacks without implying any additional costs.



[E. Alpirez Bock – SCA Resistant Implementation of the Montgomery *kP*-Algorithm]

Vertical vs horizontal DPA





Electromagnetic analysis

Works similar to power analysis…

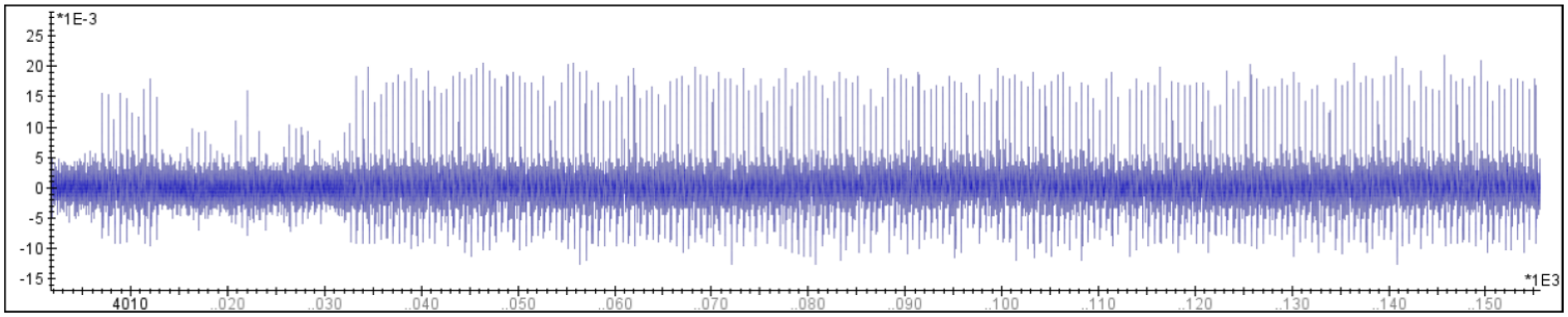
- Measuring the electromagnetic emission of the device
- Place electromagnetic coil next to the device
- Attacks could be more powerful since the coil could measure the emissions of single blocks in the chip architecture and their behaviour could show key dependencies.
  Example with an implementation of the Montgomery kP-algorithm:

$$
\begin{aligned}
&\textbf{if } k_i = 1 \textbf{ then} \\
&\quad Q[0] \leftarrow Q[0] + Q[1],\ Q[1] \leftarrow 2Q[1]. \\
&\textbf{else} \\
&\quad Q[1] \leftarrow Q[0] + Q[1],\ Q[0] \leftarrow 2Q[0]. \\
&\textbf{end if}
\end{aligned}
$$

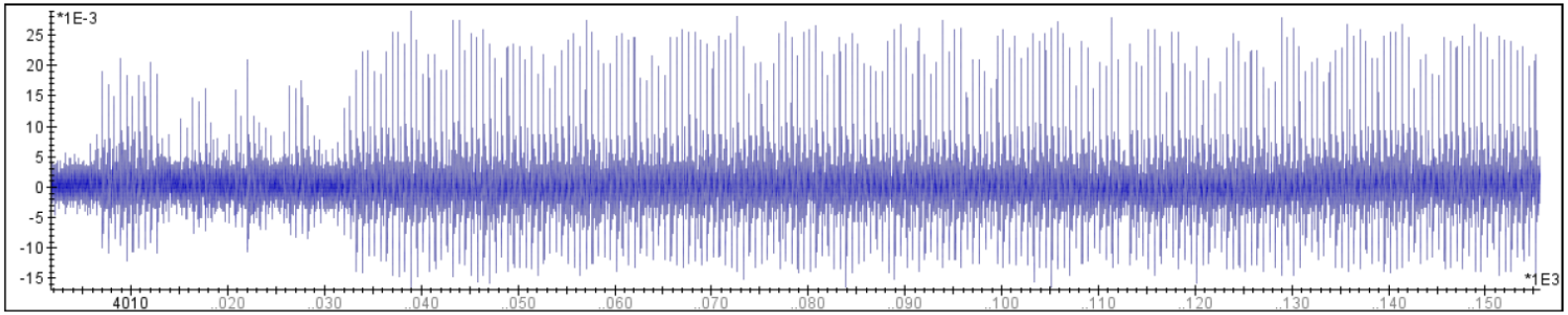
Electromagnetic analysis

Chip decapsulating can lead to better measurement results of the EM radiation

- EM trace measured on top of a non-decapsulated chip



- EM trace measured on top of a decapsulated chip



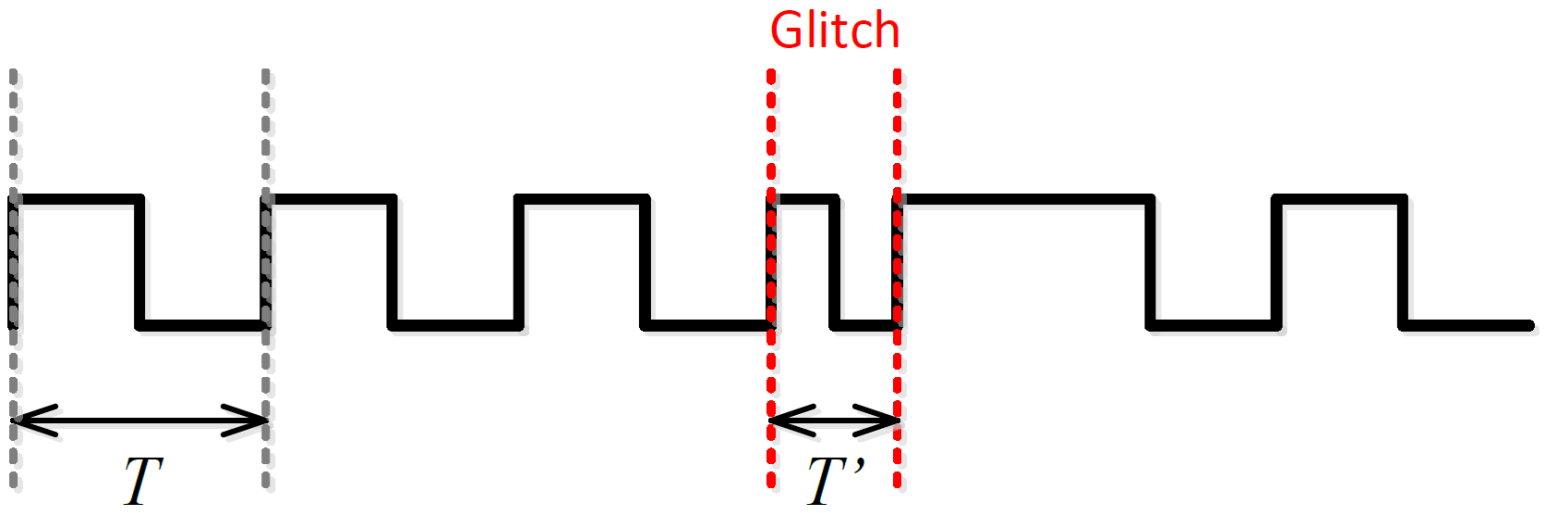
[Source: C. Wittke - Preparation of SCA Attacks: Successfully Decapsulating BGA Packages]

Active Attacks

Active attacks aim at manipulating the behaviour of the device under attack, for example through fault injection. These attacks are **not** grouped under SCA attacks.

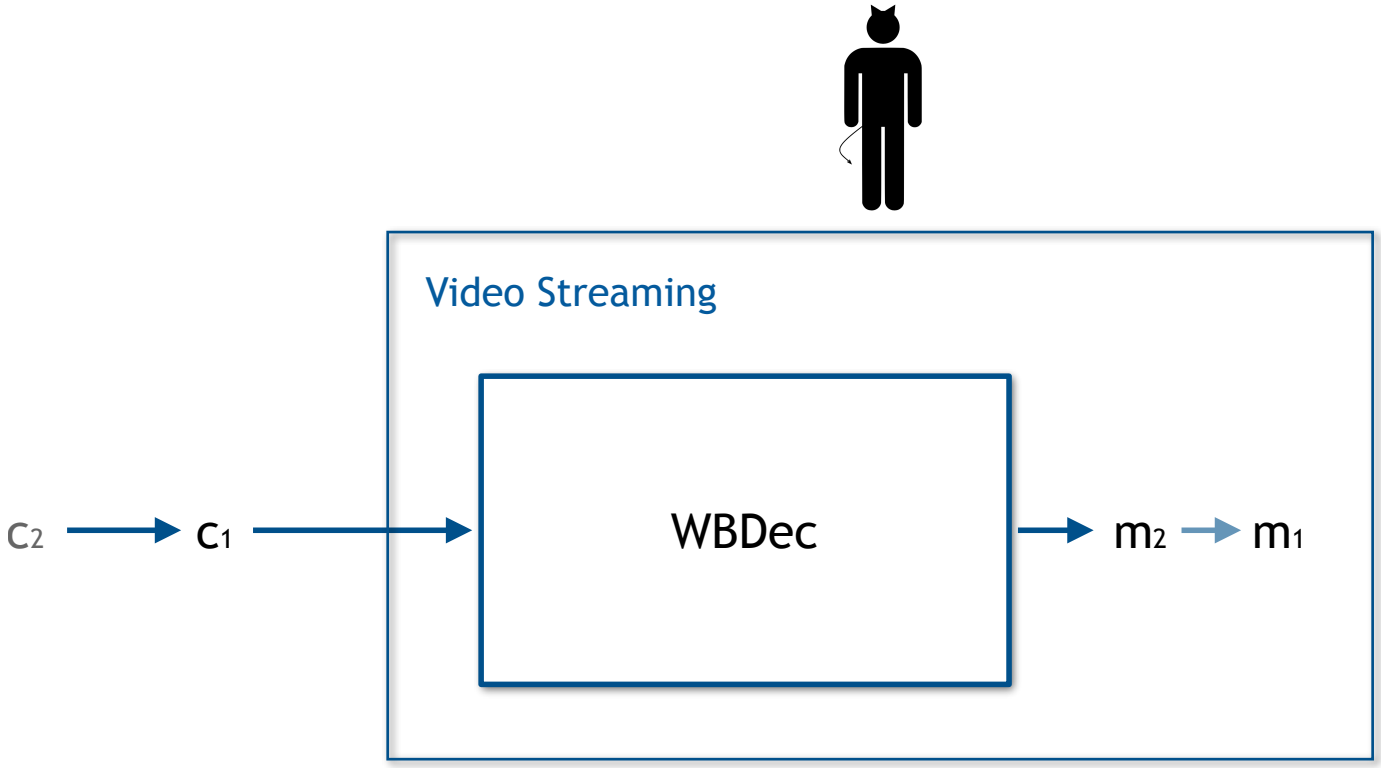
Fault injection techniques:

- Optical fault injection – generated with a strong light source
- Voltage perturbation
- Clock frequency perturbation



White-box cryptography

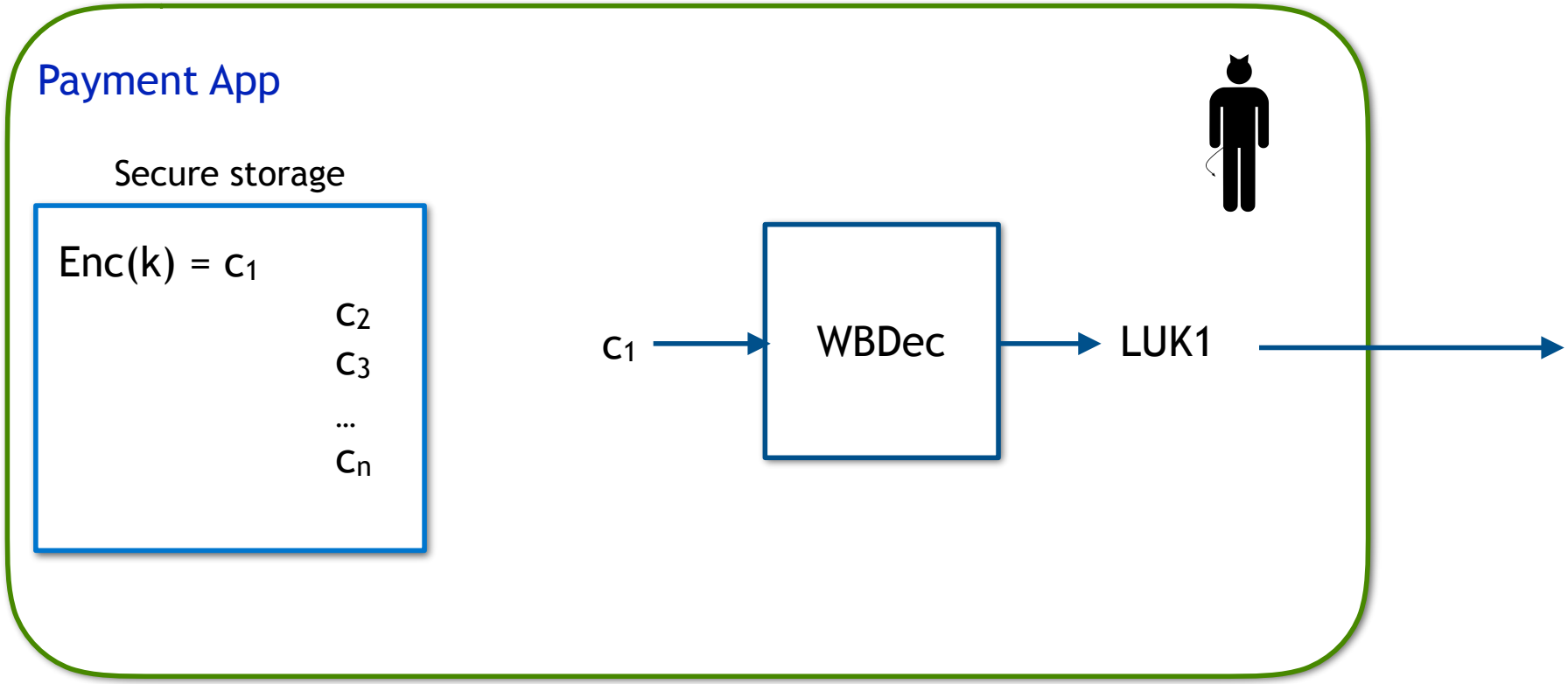
Use case: Digital Rights Management



- White-box crypto should help mitigate piracy
- Owner of the application is considered an adversary

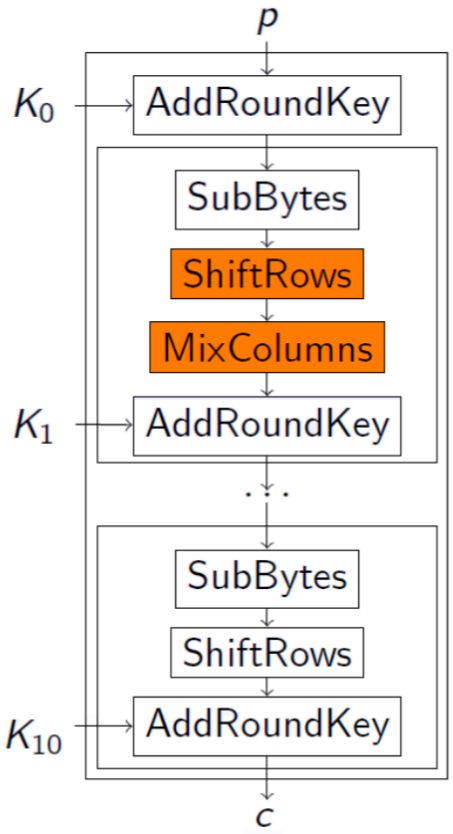
Use case: mobile payment applications

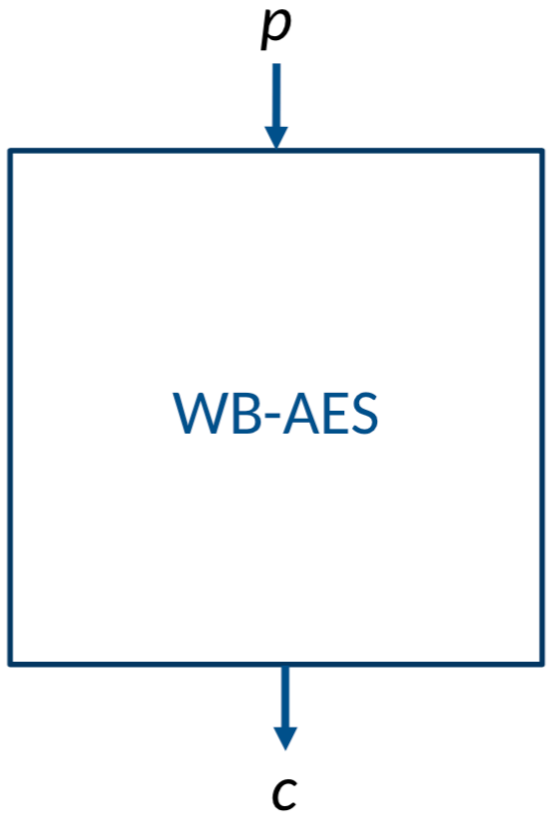
- Payment applications make use of tokens for performing transactions
  - Cryptographic keys need to be stored securely



White-box implementations

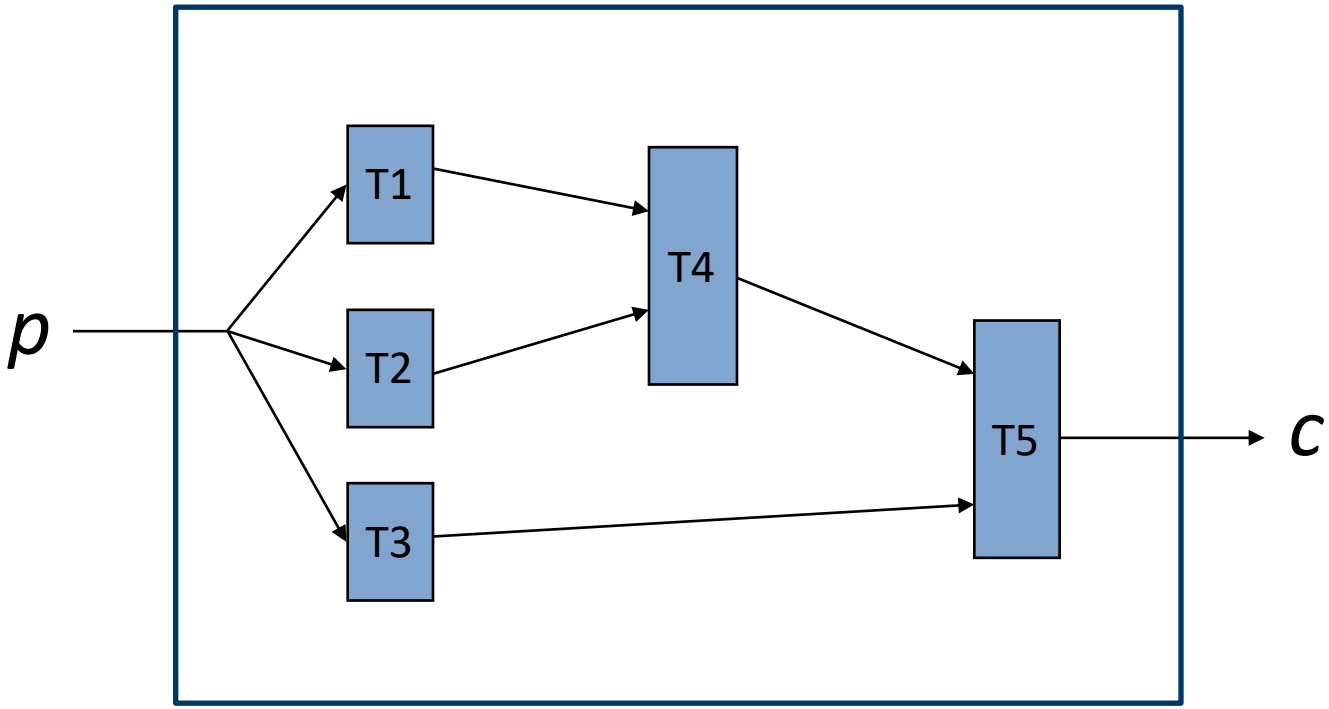
- A table based implementation has been the most popular approach in the literature for white-box designs
- The “perfect” white-box would consist of a single look-up table which directly maps an plaintext to a ciphertext





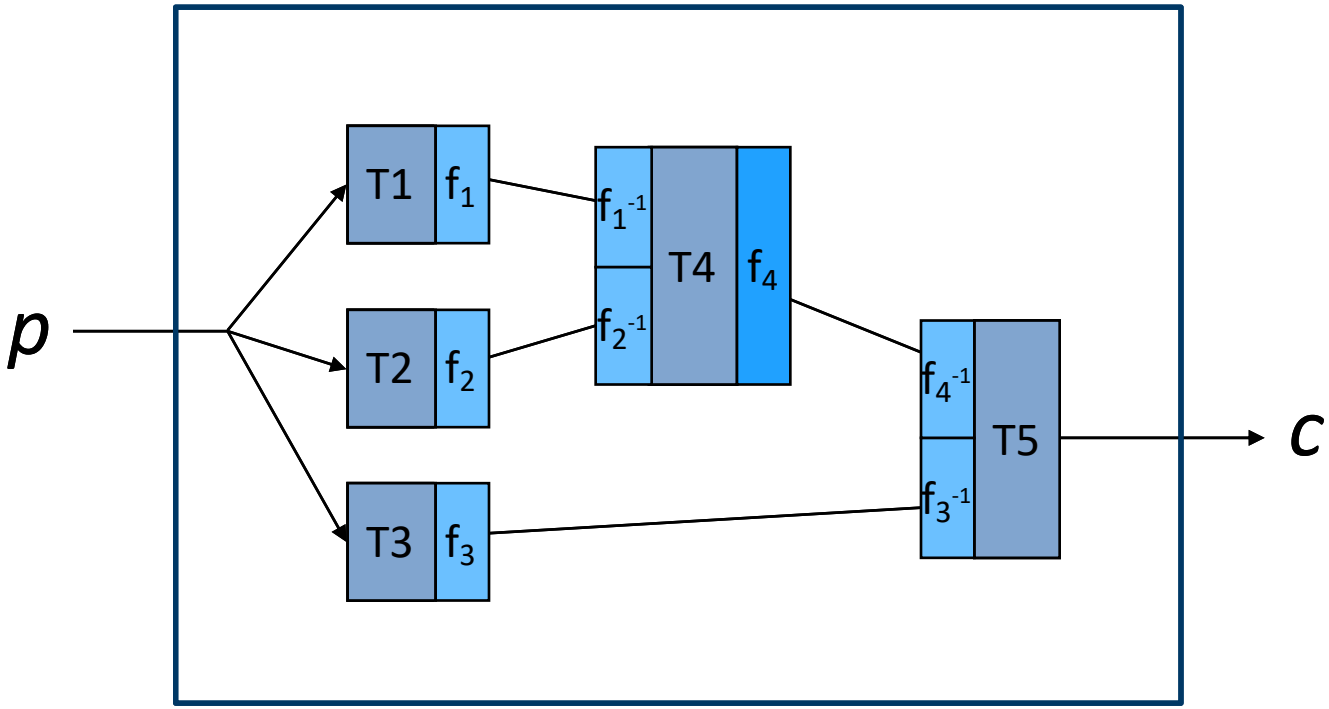
White-box implementations

- General approach: implement the cipher as a network of key-dependent look-up table
- Each look-up table corresponds to a step in the algorithm



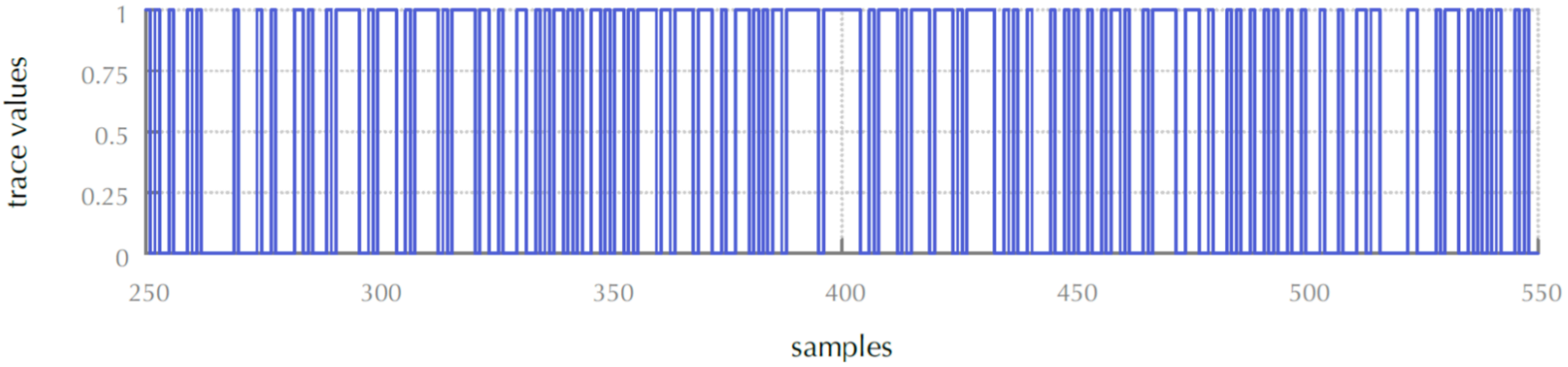
White-box implementations

- The contents of the look-up table can be *obscured* via randomised encodings



Differential Computation Analysis

- Automated and efficient attack on white-box implementations presented by Bos et al. [1] and Sanfelix et al. [2]
- Records the memory addresses accessed during the encryption process and obtains *software execution traces*

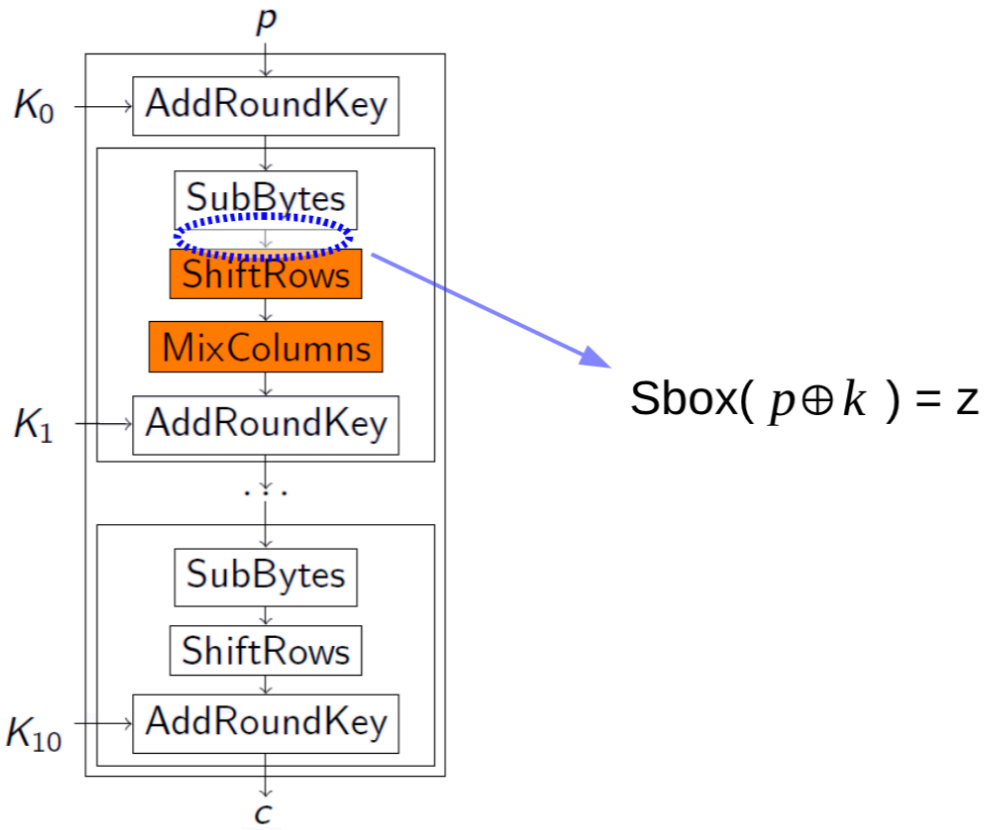


- Software traces can be analysed with traditional DPA tools

[1] J. W. Bos, C. Hubain, W. Michiels, and P. Teuwen: *Differential Computation Analysis: Hiding your White-Box Designs is Not Enough*. **CHES 2016.**
[2] E. Sanfelix, C. Mune, J. de Haas: *Unboxing the White-Box: Practical Attacks Against Obfuscated Ciphers*. **Black Hat Europe 2015.**

Differential Computation Analysis

1. Encrypt *n* plaintexts and record one software trace by each encryption
2. Define a *selection function sel = z[b]* $\in \{0,1\}$ where *z* is an intermediate value calculated based on the known plaintext p_i and a key guess k^h

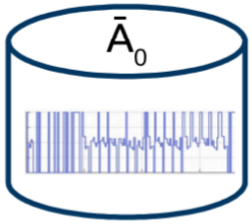


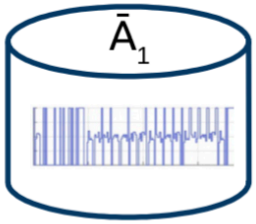
Differential Computation Analysis

1. Encrypt *n* plaintexts and record one software trace by each encryption
2. Define a *selection function* $sel = z[b] \in \{0,1\}$ where z is an intermediate value calculated based on the known plaintext p_i and a key guess k^h
3. For each plaintext p_i, calculate $sel(p_i,k^h)=b$ and sort each software trace s_i in the set $\boldsymbol{A}_b$, with $b \in \{0,1\}$

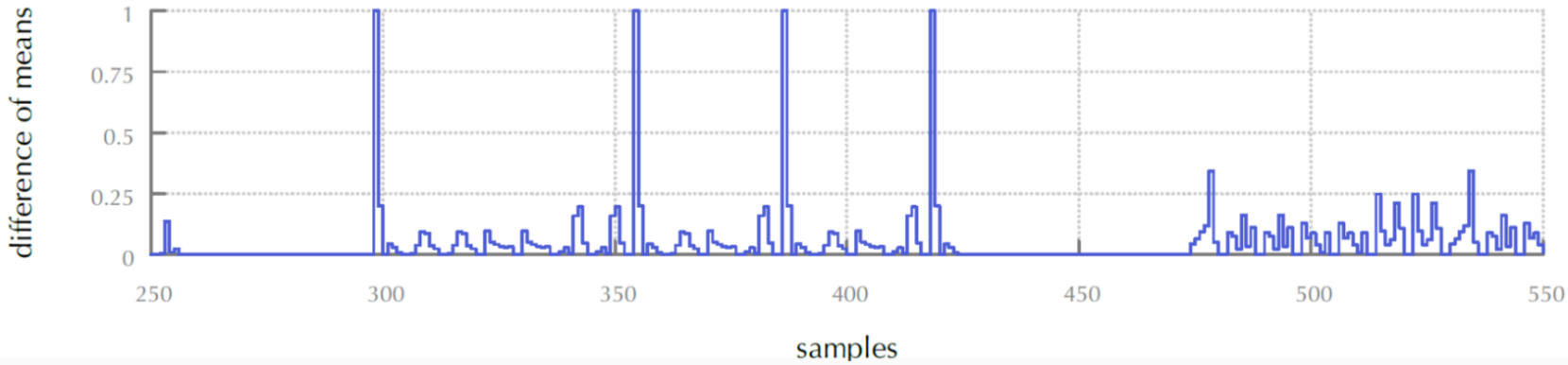
Differential Computation Analysis

4. Calculate the mean value $\bar{A}_b$ of each set.

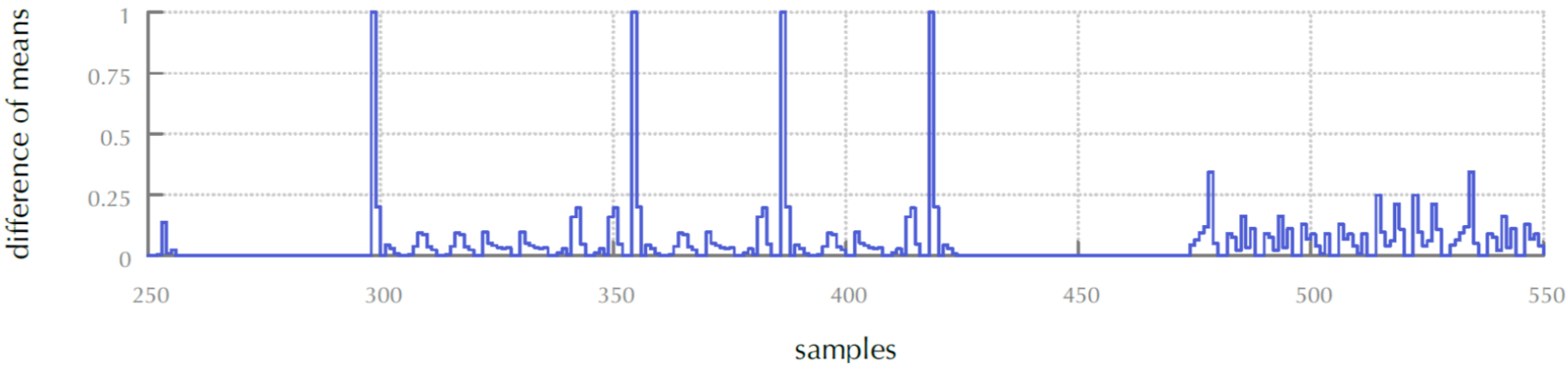




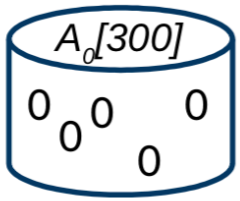
5. Calculate the difference between the average of each set $\Delta = |\bar{A}_0 - \bar{A}_1|$

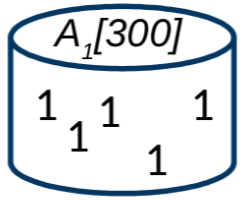


Analysing the results



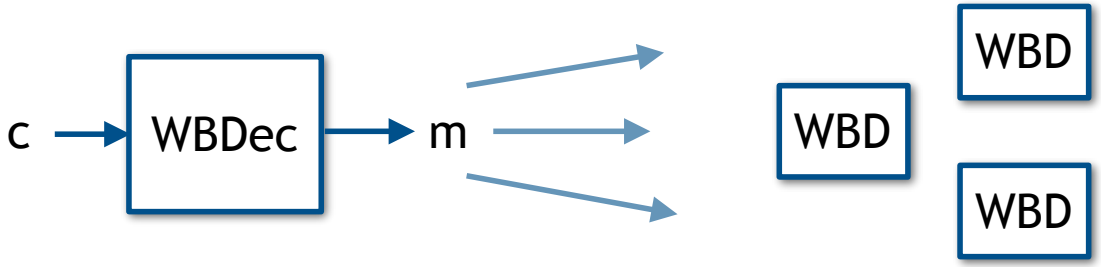
The peaks help us recognize that our key guess was correct: we calculated all values z[b] correctly and the traces have been sorted correctly in the sets.





Further issues in the white-box attack scenario

- Besides protecting against key extraction attacks, a software program can also suffer from code-lifting attacks



Link the execution of the program to a specific hardware device, or make it Dependent of an authentication process linked to the owner of the program

Links to program obfuscation

- One of the main tools for implementing white-box programs is known as program or code obfuscation
  - An obfuscator takes as input a program P and outputs a functionally equivalent program, which is intelligible
- Obfuscation is widely used in real life applications, but its foundations have also been studied within the scientific community

Thank you for your attention!

estuardo.alpirezbock@aalto.fi
Aalto University,
Department of Mathematics and Systems Analysis
Otakaari 1, Espoo
Room Y250c
Finland

Aalto University