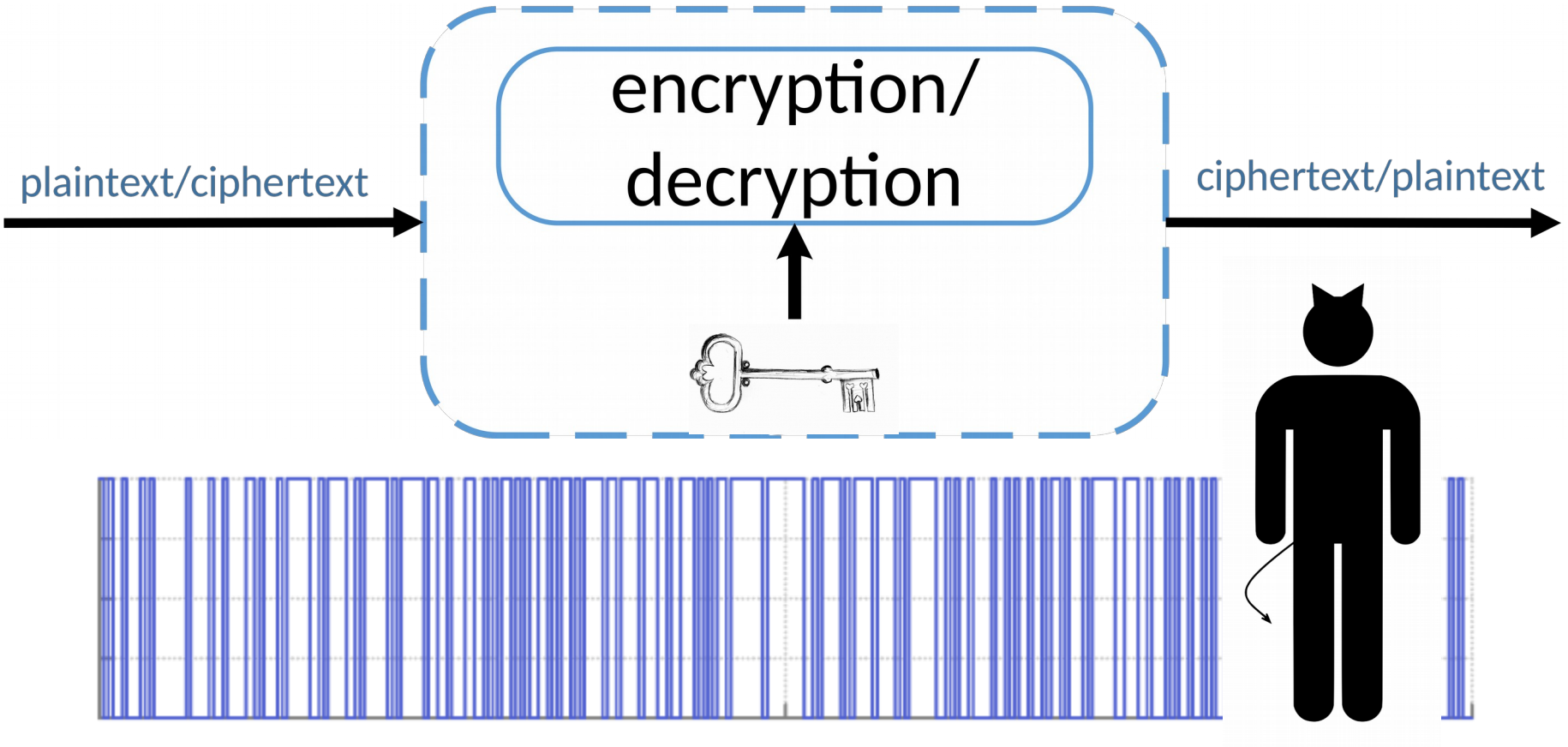
On the Ineffectiveness of Internal Encodings – Revisiting the DCA Attack on White-Box Cryptography

Estuardo Alpírez Bock, Chris Brzuska, Wil Michiels and Alexander Treff

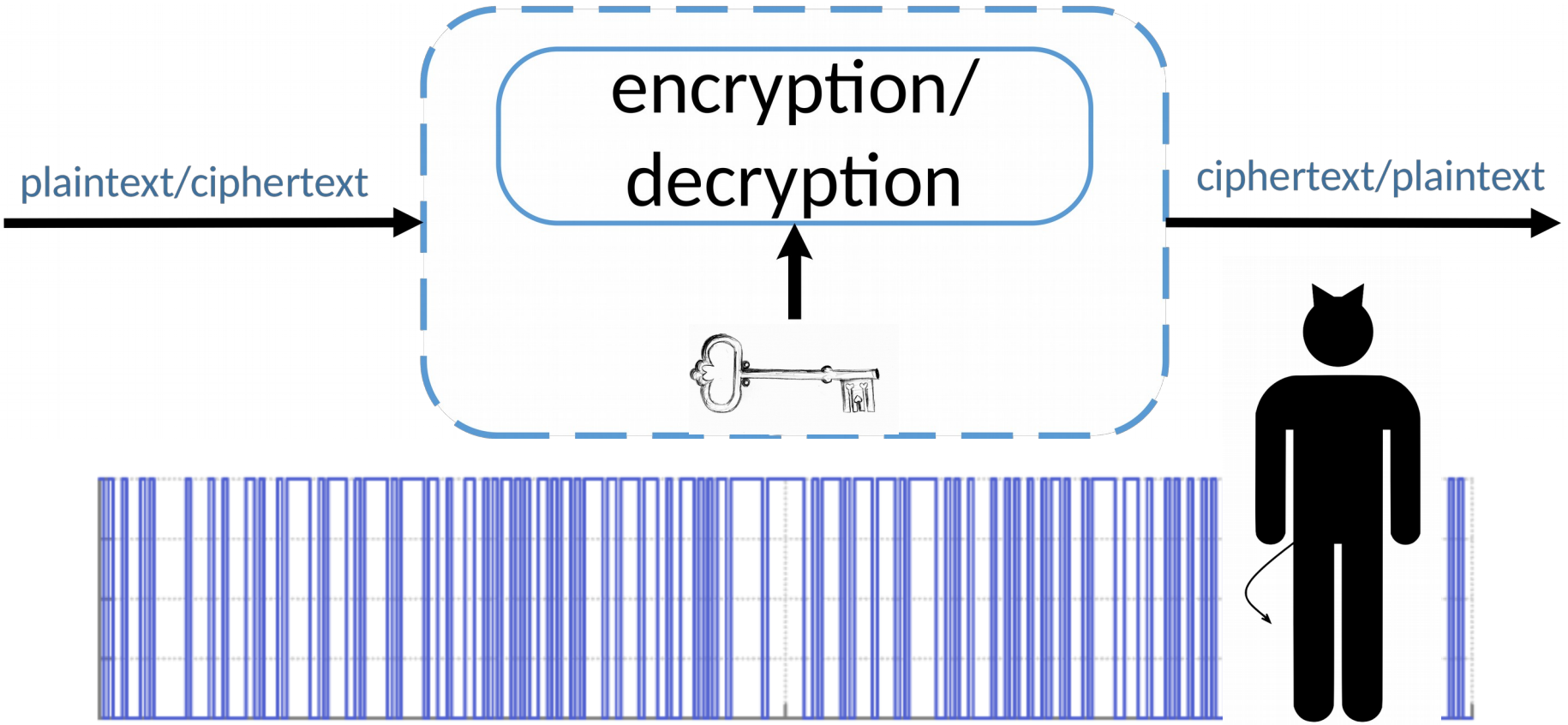
2nd July 2018

16th International Conference on Applied Cryptography and Network Security

Whitebox Attack Scenario



Whitebox Attack Scenario



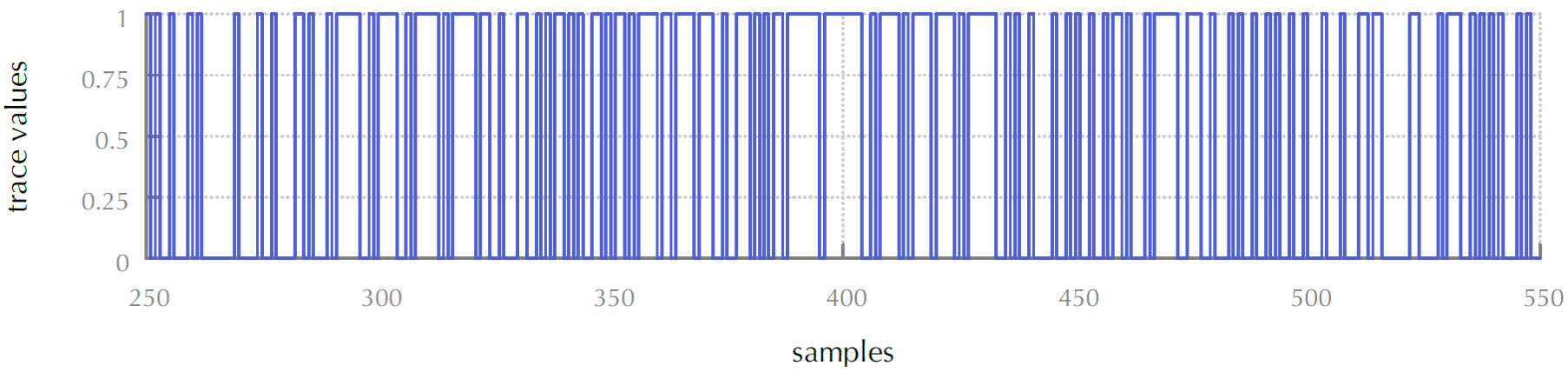
- WB Cryptography aims to maintain a program secure even when subject to such attack scenarios

Attacks on White-Box Implementations

- Attacks based on reverse engineering can successfully perform key extractions from white-box implementations
  - Knowledge of the look up tables is required, plus deobfuscating tools
  - Such attacks demand big efforts from the adversary and require some time
  - Example: BGE attack - Olivier Billet, Henri Gilbert, Charaf Ech-Chatbi: *Cryptanalysis of a White Box AES Implementation*. ASIACRYPT 2003.

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.**

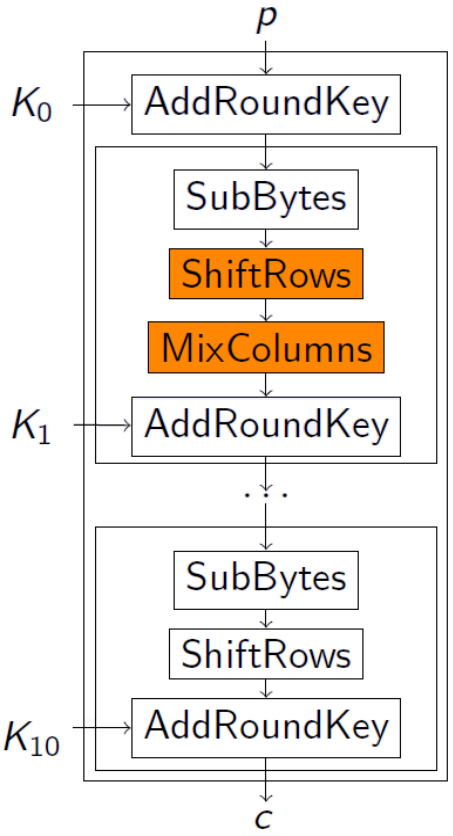
Differential Computation Analysis

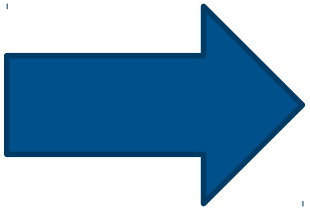
- The DCA can successfully break many openly available white-box implementations

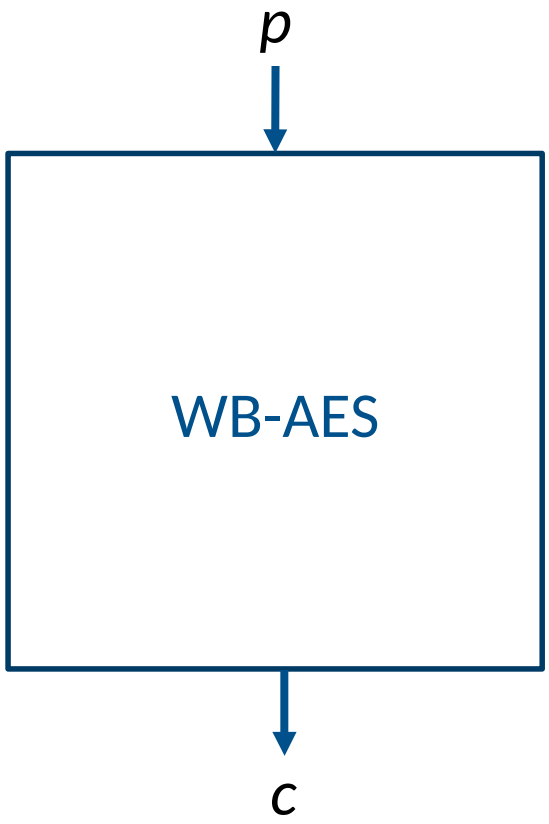
- But why is it successful?
  - In this work we revisit the DCA attack and explain its experimental success
  - We explain how key dependencies are correlated to the information contained in the software traces
  - We conclude how some popular design frameworks for white-box cryptography are not a safe choice in light of the DCA

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

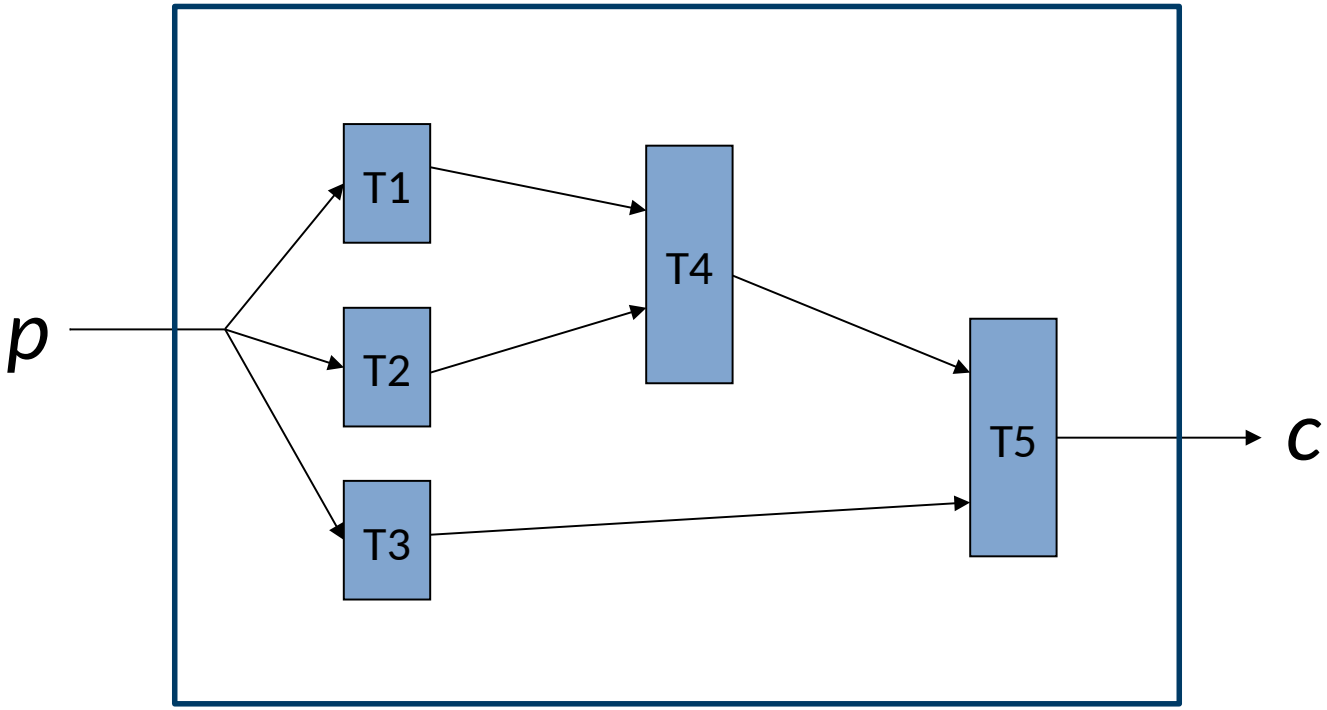






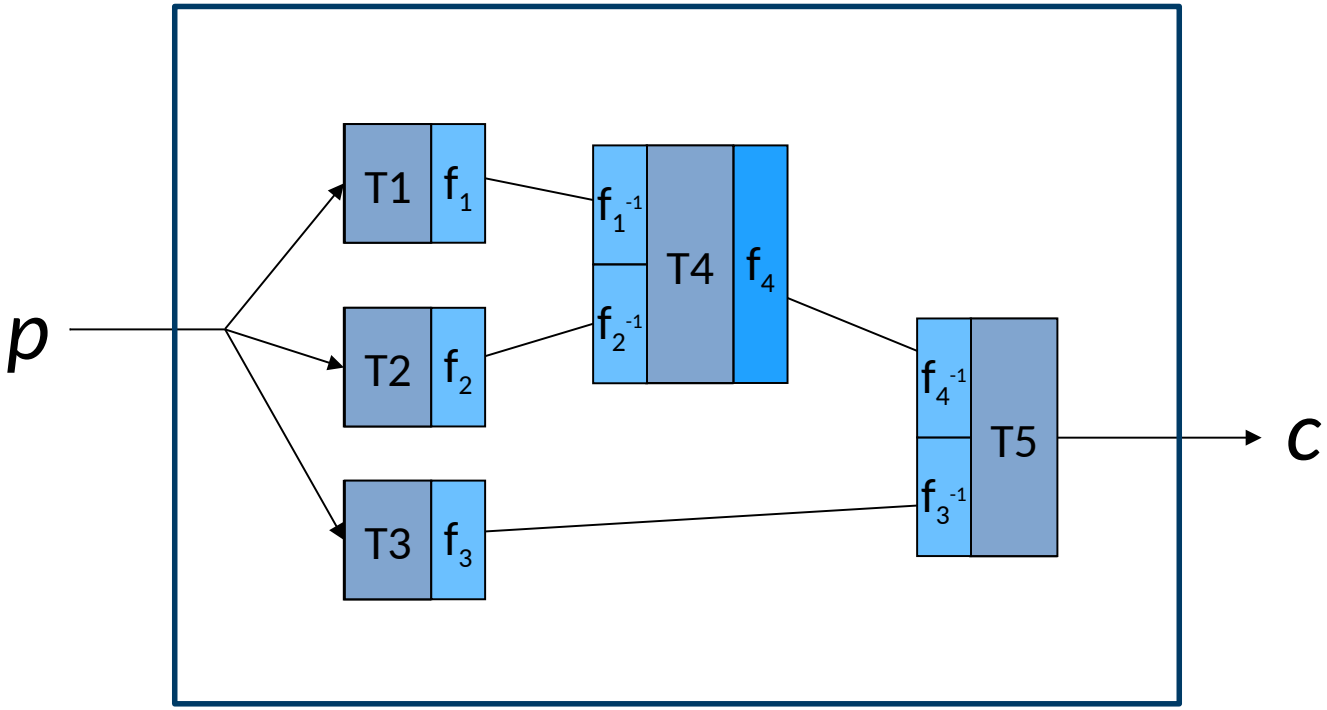
White-box implementations

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



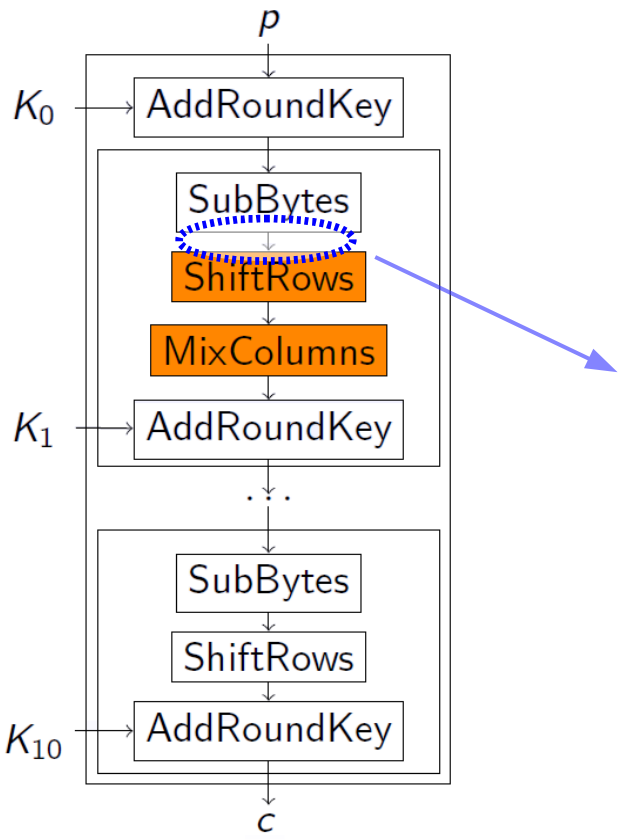
White-box implementations

- General approach: implement a cipher as a network of key-dependent look-up tables
- Each look-up table corresponds to a step of the algorithm
- The contents of each look-up table can be obscured via input and output encodings.



Differential Computation Analysis

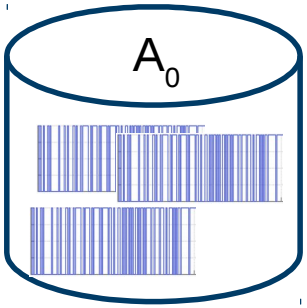
1. Encrypt *n* plaintexts and record one software trace by each encryption
2. Define a *selection function* $sel = z[b] \in \{0,1\}$ where z is an intermediate value calculated based on the known plaintext p_i and a key guess k^h

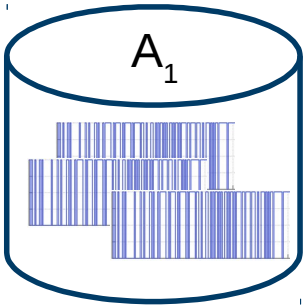


Sbox($p \oplus k$) = z

Differential Computation Analysis

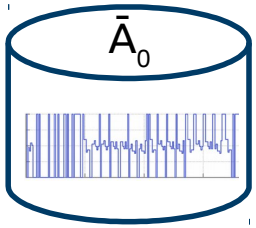
1. Encrypt *n* plaintexts and record one software trace by each encryption
2. Define a *selection function* $sel = z[b] \in \{0,1\}$ where z is an intermediate value calculated based on the known plaintext p_i and a key guess k^h
3. For each plaintext p_i, calculate $sel(p_i,k^h)=b$ and sort each software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

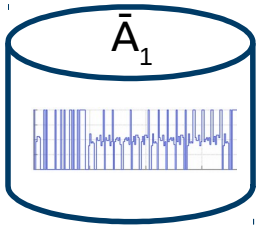




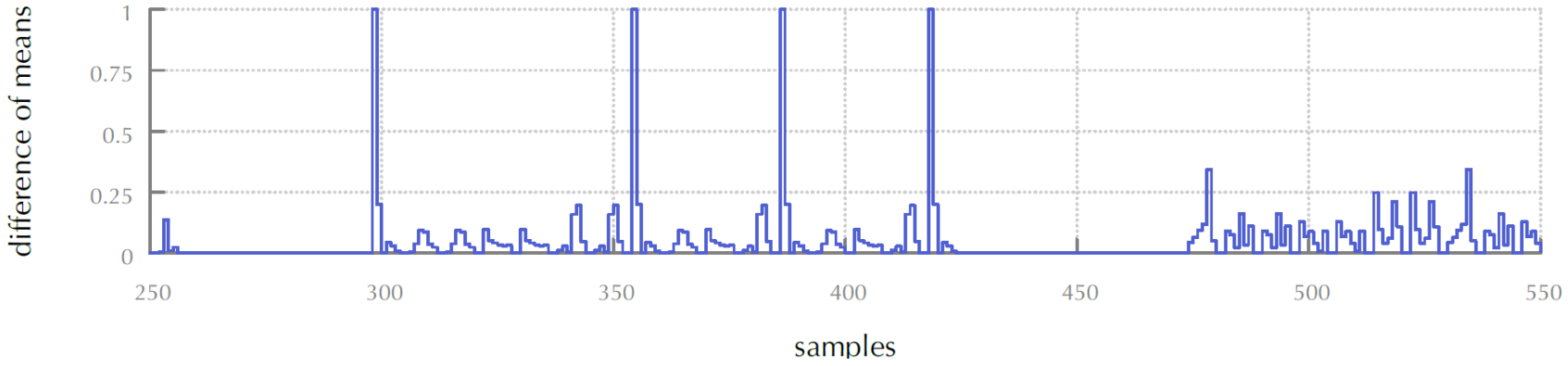
Differential Computation Analysis

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

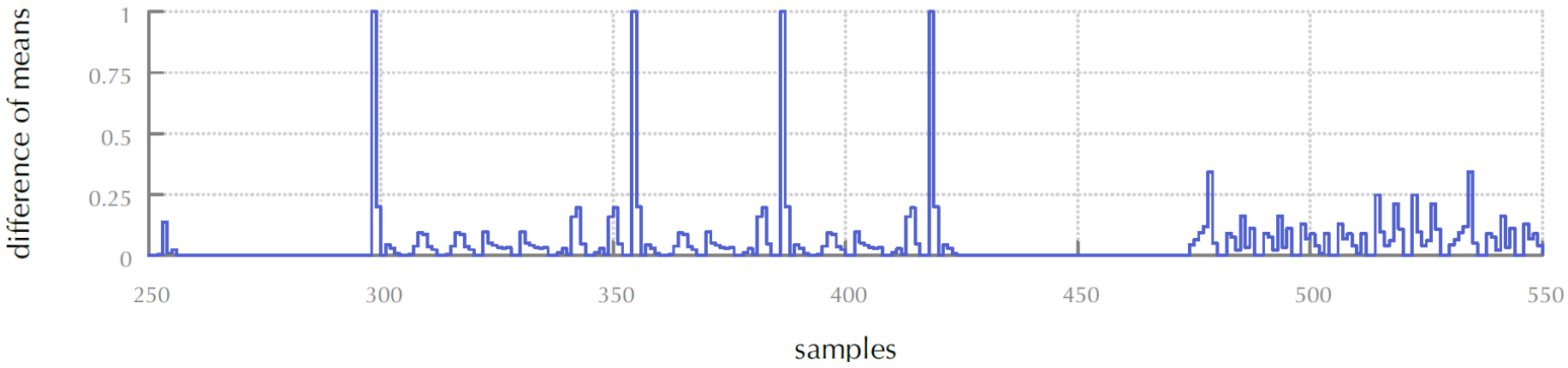




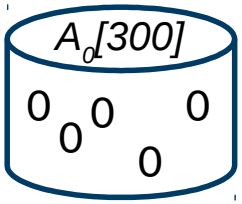
5. Calculate the difference between the average of each set $\Delta = |\bar{A}_0 - \bar{A}_1|$

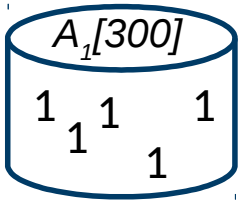


Analysing the results

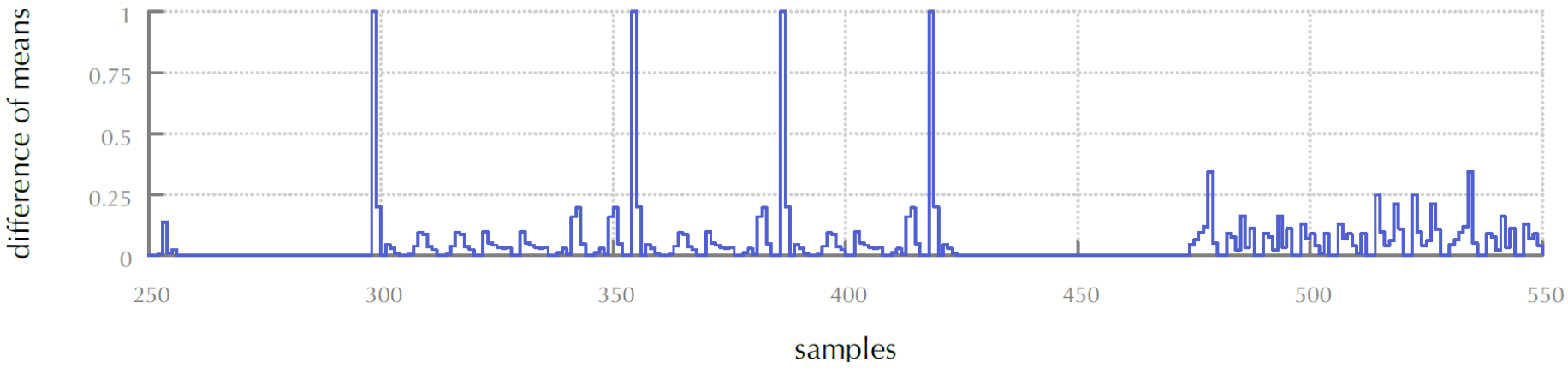


The peaks help us recognize that our key guess was correct: we calculated all values z[b] correctly and the traces have been sorted correctly in the sets.

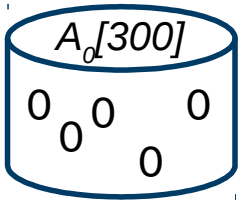


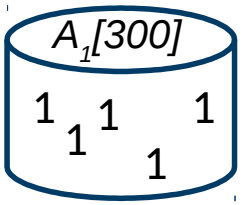


Analysing the results



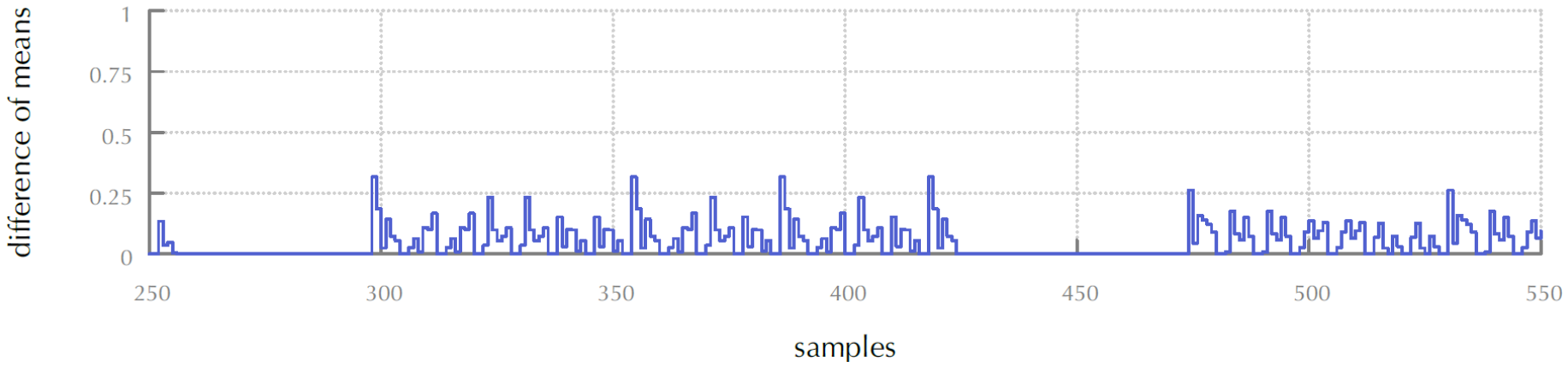
We also learn that the intermediate values z were not encoded by the white-box implementation





Analysing the results

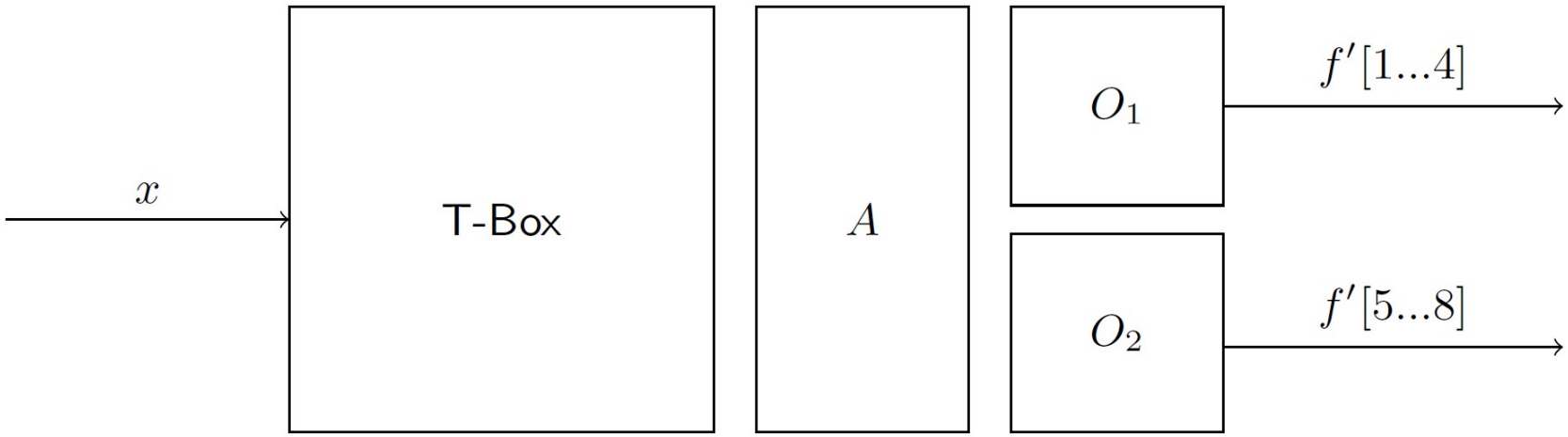
If we sort according to an incorrect key guess, the output of Sel(p_i does not always match with the computations of the white-box



But what happens if the intermediate state values are encoded?

Encodings for a WB-Implementation

- Combination of linear and non-linear encodings to protect key-dependent look-up tables in a white-box design.



  - Suggested by Chow et al. [3] as a countermeasure against reverse engineering attacks

[3] S. Chow, P.A. Eisen, H. Johnson and P.C. van Oorschot: *White box cryptography and an AES implementation*. **SAC 2002**.

Our Analysis

- Construct an 8-bit to 8-bit key-dependent look-up table (T-box) and analyse it via DCA using exactly 2^8=256 different inputs
- Apply different types of encodings to the T-boxes and attack them via DCA
- Construct AES-WB-Implementations which make use of the different types of encodings and attack them via DCA
    - How are the results of the DCA affected by each use of encodings?

DCA results on different WB-implementations

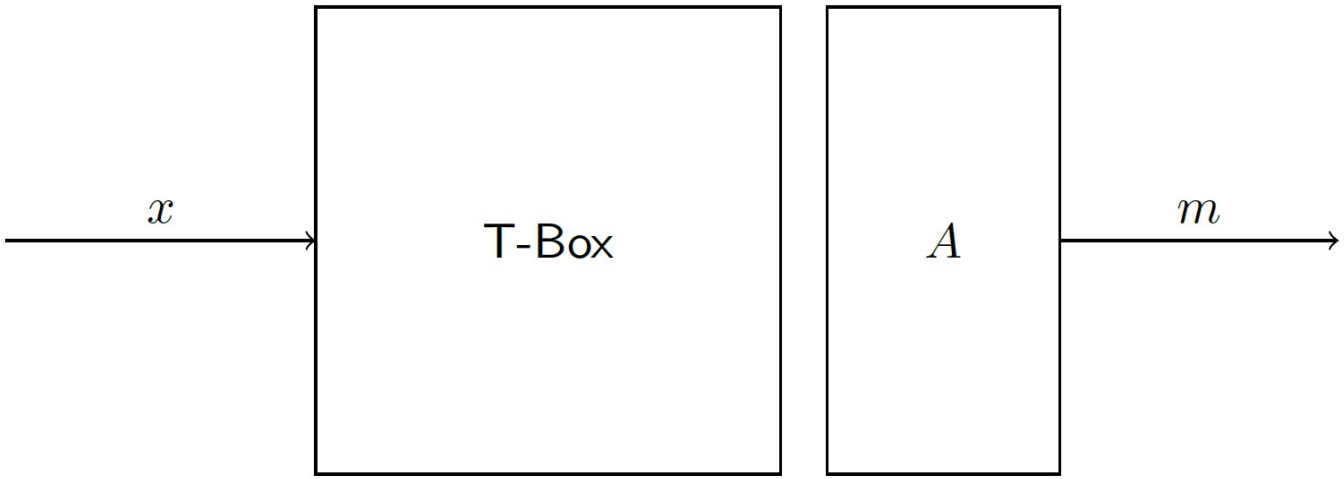
Encoding type	Single T-box	WB-AES
None	DCA successful w/ perfect correlations	DCA successful w/ perfect correlations
Linear	DCA (partly) successful w/ perfect correlations	DCA (partly) successful w/ perfect correlations
Non-linear	DCA successful - Peak values of 0.25, 0.5, 0.75 or 1	DCA successful - Peak values of ca. 0.25, 0.5, 0.75 or 1
Linear and non-linear	DCA partly successful - Peak values of 0.25, 0.5, 0.75 or 1	DCA partly successful - Peak values of ca. 0.25, 0.5, 0.75 or 1

DCA results on different WB-implementations

Encoding type	Single T-box	WB-AES
None	DCA successful w/ perfect correlations	DCA successful w/ perfect correlations
Linear	DCA (partly) successful w/ perfect correlations	DCA (partly) successful w/ perfect correlations
Non-linear	DCA successful - Peak values of 0.25, 0.5, 0.75 or 1	DCA successful - Peak values of ca. 0.25, 0.5, 0.75 or 1
Linear and non-linear	DCA partly successful - Peak values of 0.25, 0.5, 0.75 or 1	DCA partly successful - Peak values of ca. 0.25, 0.5, 0.75 or 1

Linear encodings

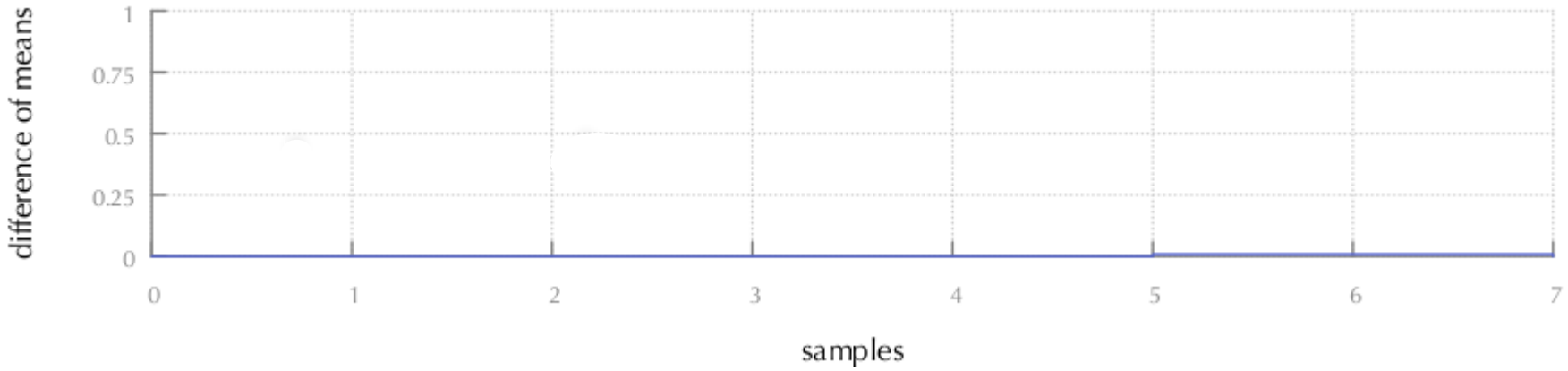
- We multiply each output of a T-box with a randomly generated invertible matrix and obtain a linearly encoded T-box



- Maps each input *x* to a linearly encoded output *m*

Attack on a linearly encoded T-box

Using the correct key-guess, we obtain the following difference of means curve:



- No significant peaks can be observed → the DCA is not successful

Attack on a linearly encoded T-box

What if we choose a different matrix to encode the outputs of the T-box?

- Recall that, to calculate each encoded output bit *m[i]* we do the following:

$$m[i] = \sum_{j} a_{ij} \cdot z[j] = \sum_{j : a_{ij} = 1} z[j]$$

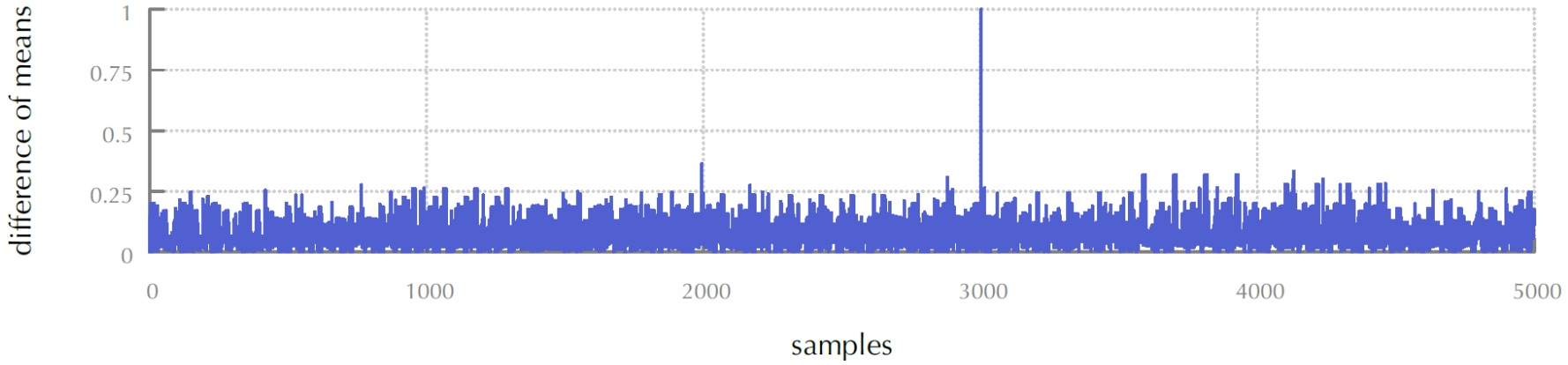
where z*[j]* denotes the jth the T-box output and a_{ij} are the entries of the matrix *A*.

- If row *i* has a Hamming weight (HW)=1, then it follows that:

$$m[i] = z[b]$$

DCA on a complete white-box

- WB constructed using randomly generated linear encodings



Is an invertible matrix, which does not have any identity row a good countermeasure against the DCA?

Attack on a linearly encoded T-box

Is an invertible matrix, which does not have any identity row a good countermeasure against the DCA?

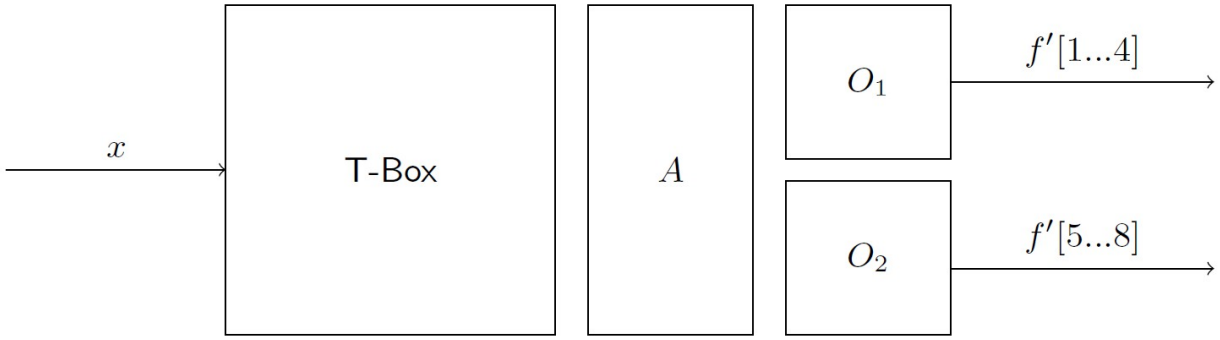
→ Not really, our selection function could be easily modified such that, after calculating an 8-bit state z, it calculates all possible linear combinations of z

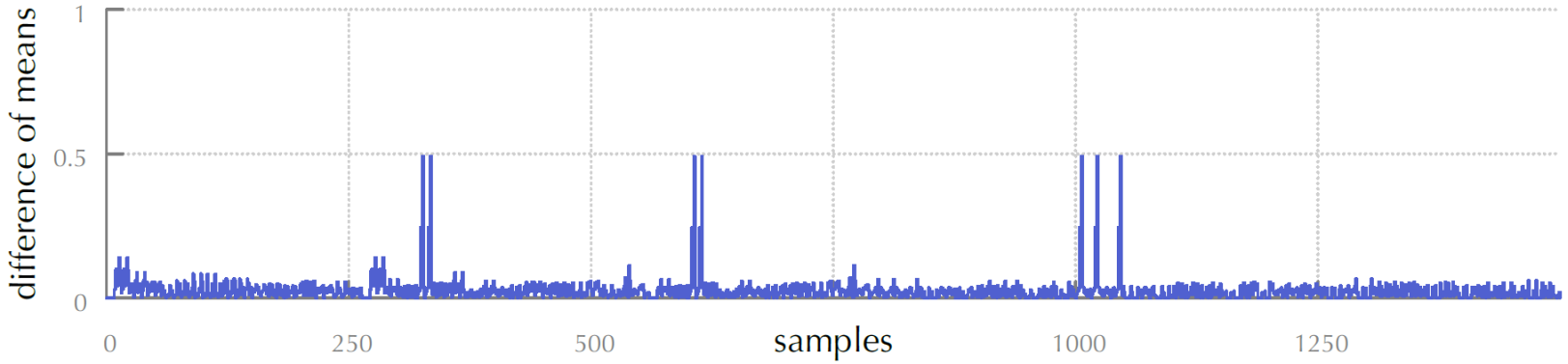
$$sel = LC(z) \in \{0,1\}$$

There will be an LC which will be equal to the LC defined by some matrix row i. For that case we will obtain perfect correlations in our DCA

Attacking a complete white-box

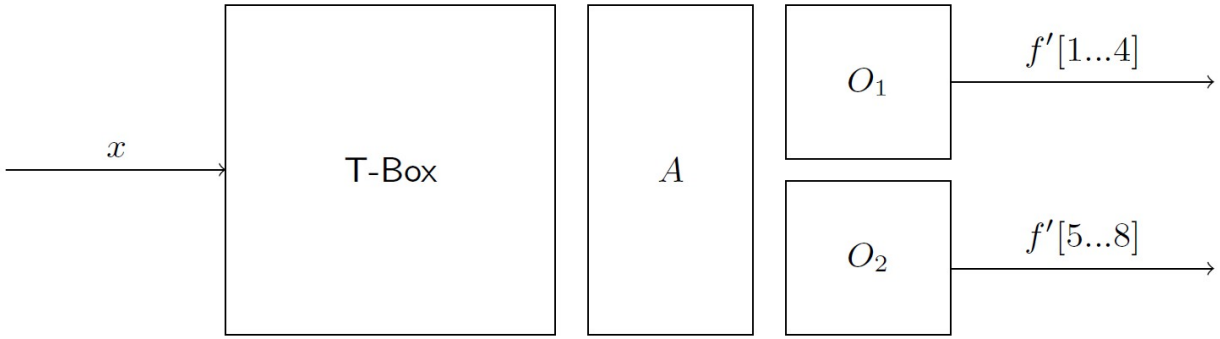
White-box constructed using randomly generated linear and non-linear encodings

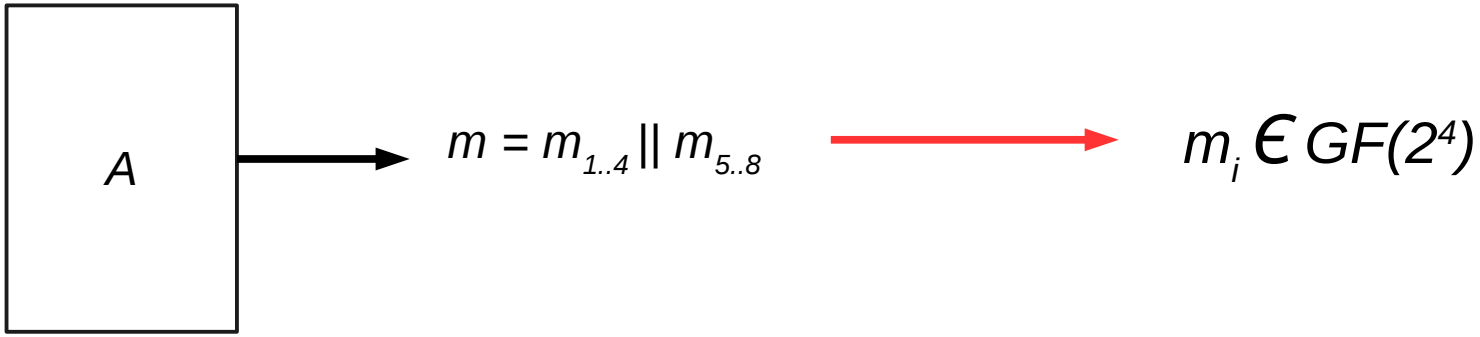




Attacking a complete white-box

White-box constructed using randomly generated linear and non-linear encodings

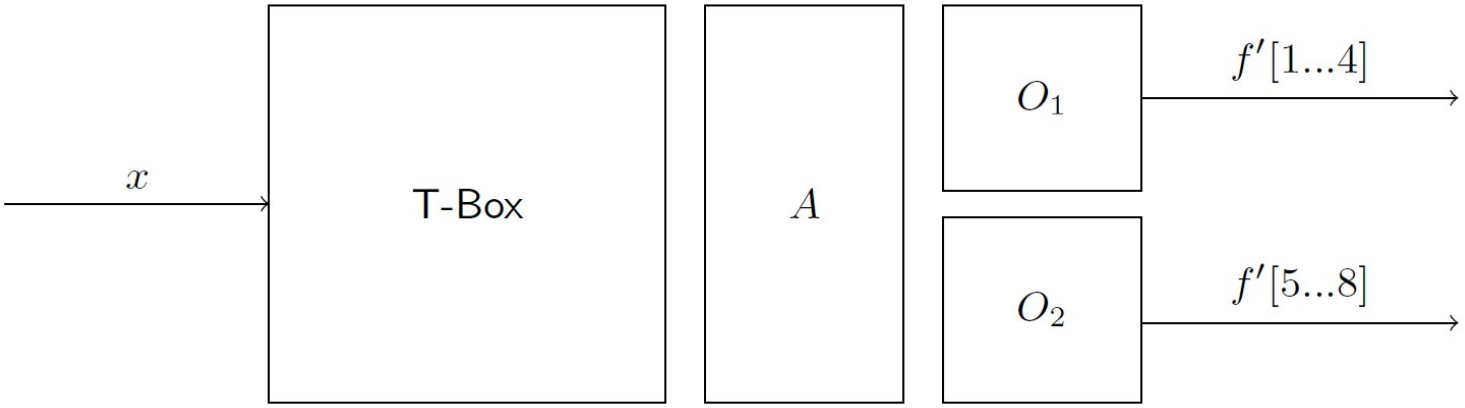




Thank you for your attention!
Dank je wel!

Encoding type	Single T-box	WB-AES
None	DCA successful w/ perfect correlations	DCA successful w/ perfect correlations
Linear	DCA (partly) successful w/ perfect correlations	DCA (partly) successful w/ perfect correlations
Non-linear	DCA successful - Peak values of 0.25, 0.5, 0.75 or 1	DCA successful - Peak values of ca. 0.25, 0.5, 0.75 or 1
Linear and non-linear	DCA partly successful - Peak values of 0.25, 0.5, 0.75 or 1	DCA partly successful - Peak values of ca. 0.25, 0.5, 0.75 or 1

Attacking a complete white-box



- The output nibbles of *A* need to take any possible 4 bit long value in order to provide robustness against the DCA.

  $\longrightarrow$ we otherwise return to the same issues presented due to the use of nibble encodings

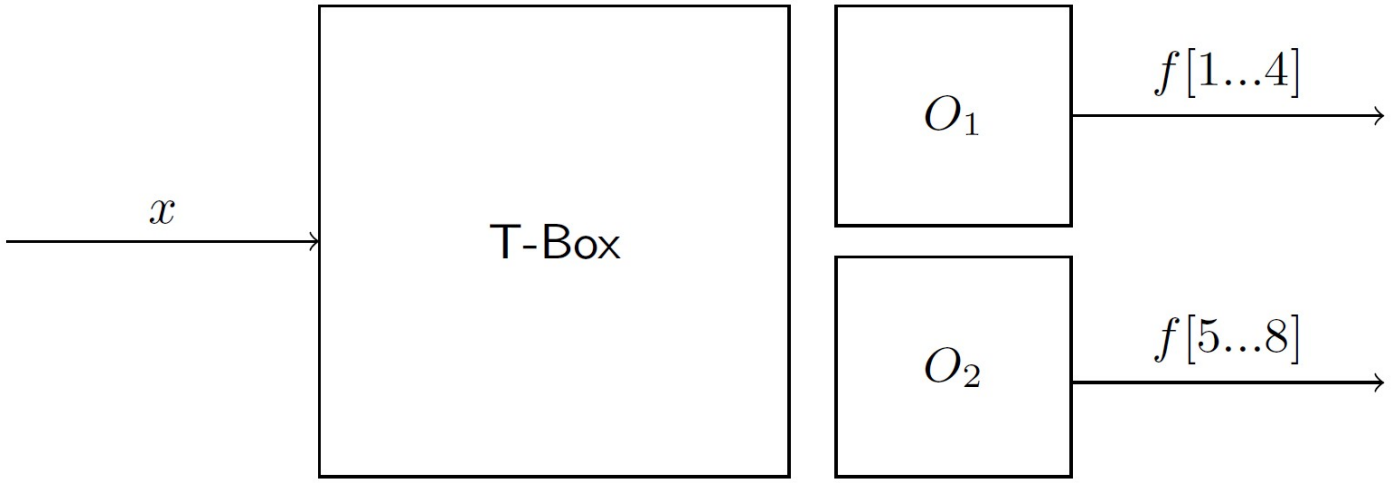
Conclusions

- Revisited the analysis steps of the DCA using software execution traces
- Explained why the DCA attack was successful on white-box implementations attacked in [1] and why so many implementations are vulnerable to this attack [3] .
- Nibble encodings make a white-box very vulnerable against the DCA.
- These results motivate the study for new encodings alternatives for WB-designs.

[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. S. Gonzalez, C. Mune, J. de Haas: *Unboxing the White-Box: Practical Attacks Against Obfuscated Ciphers*. **Black Hat Europe 2015.**
[3] CHES WhibOx Contest https://whibox.cr.yp.to/

Non-linear encodings

- Each output of the LUT is split in two halfs. Each half is encoded by a different non-linear function



- Maps each input *x* to a non-linearly encoded output *f = f[1..4] || f[5..8]*

Attack on non-linearly encoded T-boxes

- We construct 10 000 different non-linearly encoded T-boxes and attack all of them via the DCA.
  - The attack succeeds for 9 997 non-linearly encoded T-boxes
  - For the successful cases, the following peak values are returned for the correct key guesses:

Peak value for correct key	Nr. of non-lin. encoded T-boxes
1	55
0.75	2804
0.5	7107
0.25	31

  - The wrong key guesses return lower peak values.

Attack on non-linearly encoded T-boxes

- We construct 10 000 different non-linearly encoded T-boxes and attack all of them via the DCA.
  - The attack succeeds for 9 997 non-linearly encoded T-boxes
  - For the successful cases, the following peak values are returned for the correct key guesses:

Peak value for correct key	Nr. of non-lin. encoded T-boxes
1	55
0.75	2804
0.5	7107
0.25	31

  - The wrong key guesses return lower peak values.

Attack on non-linearly encoded T-boxes

- We construct 10 000 different non-linearly encoded T-boxes and attack all of them via the DCA.
  - The attack succeeds for 9 997 non-linearly encoded T-boxes
  - For the successful cases, the following peak values are returned for the correct key guesses:

Peak value for correct key	Nr. of non-lin. encoded T-boxes
1	55
0.75	2804
0.5	7107
0.25	31

  - The wrong key guesses return lower peak values.

Attack on non-linearly encoded T-boxes

- We construct 10 000 different non-linearly encoded T-boxes and attack all of them via the DCA.
  - The attack succeeds for 9 997 non-linearly encoded T-boxes
  - For the successful cases, the following peak values are returned for the correct key guesses:

Peak value for correct key	Nr. of non-lin. encoded T-boxes
1	55
0.75	2804
0.5	7107
0.25	31

  - The wrong key guesses return lower peak values.

Attack on non-linearly encoded T-boxes

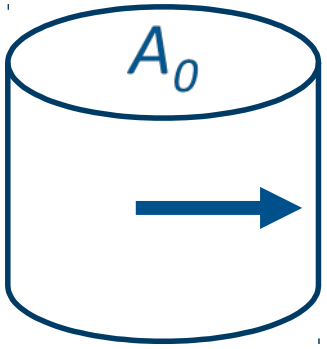
Say we calculate z[1] with sel

If z[1] = 0

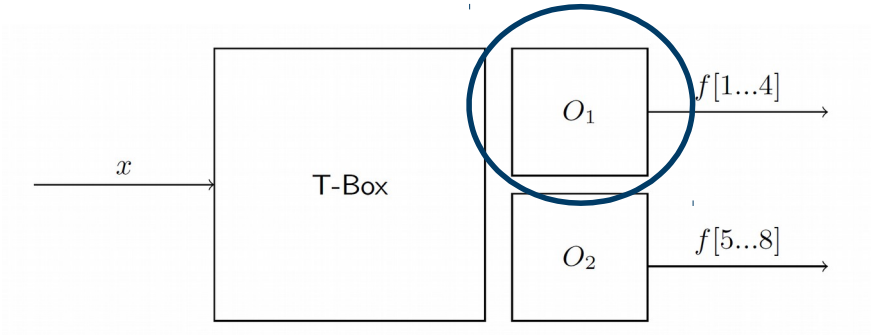
then $A_0 := A_0 \cup f[1..4]$

→There exist only 8 possible nibble values for which z[1]=0

→Each one of this values is always encoded by the same bijective function



contains only 8 different values each one repeated exactly 16 times



Attack on non-linearly encoded T-boxes

- We construct 10 000 different non-linearly encoded T-boxes and attack all of them via the DCA.
  - The attack succeeds for 9 997 non-linearly encoded T-boxes
  - For the successful cases, the following peak values are returned for the correct key guesses:

Peak value for correct key	Nr. of non-lin. encoded T-boxes
1	55
0.75	2804
0.5	7107
0.25	31

  - The wrong key guesses return lower peak values.

Attack on non-linearly encoded T-boxes

- We construct 10 000 different non-linearly encoded T-boxes and attack all of them via the DCA.
  - The attack succeeds for 9 997 non-linearly encoded T-boxes
  - For the successful cases, the following peak values are returned for the correct key guesses:

Peak value for correct key	Nr. of non-lin. encoded T-boxes
1	55
0.75	2804
0.5	7107
0.25	31

  - The wrong key guesses return lower peak values.