Investigating the Effectiveness of DCA Attacks on a White-Box Cryptography Implementation

J. Khelif, G. Le Diréach

Table of Contents

- White-Box model
- Side Channel & Practical Attack
- Gathering Traces & Trace analyse
- Result & Conclusion

White-Box model

White-Box Cryptography

Definition

Cryptographic algorithms designed to resist an attack model known as the white-box.

- AES White-box by chow in 2002.

WB AES-128

General aspect

- Table-based implementations, Look-up tables that map plaintext to ciphertext under a fixed key.
- Unable store all possible output.
- Use smaller, obfuscated tables.

Need to modify the classical AES 128 encryption.

AES-128 modification

- Push AddRoundKey(key0).
- Pop AddRoundKey(key9).
- Swap AddRoundKey() and ShiftRows().

$\texttt{state} \leftarrow \textit{plaintext}$
$\texttt{AddRoundKey}(\texttt{state}, k_0)$
for $r = 1 \ldots 9$
    $\texttt{SubBytes}(\texttt{state})$
    $\texttt{ShiftRows}(\texttt{state})$
    $\texttt{MixColumns}(\texttt{state})$
    $\texttt{AddRoundKey}(\texttt{state}, k_r)$
$\texttt{SubBytes}(\texttt{state})$
$\texttt{ShiftRows}(\texttt{state})$
$\texttt{AddRoundKey}(\texttt{state}, k_{10})$
$\textit{ciphertext} \leftarrow \texttt{state}$

$\texttt{state} \leftarrow \textit{plaintext}$
for $r = 1 \ldots 9$
    $\texttt{AddRoundKey}(\texttt{state}, k_{r-1})$
    $\texttt{ShiftRows}(\texttt{state})$
    $\texttt{SubBytes}(\texttt{state})$
    $\texttt{MixColumns}(\texttt{state})$
$\texttt{AddRoundKey}(\texttt{state}, k_9)$
$\texttt{ShiftRows}(\texttt{state})$
$\texttt{SubBytes}(\texttt{state})$
$\texttt{AddRoundKey}(\texttt{state}, k_{10})$
$\textit{ciphertext} \leftarrow \texttt{state}$

$\texttt{state} \leftarrow \textit{plaintext}$
for $r = 1 \ldots 9$
    $\texttt{ShiftRows}(\texttt{state})$
    $\texttt{AddRoundKey}(\texttt{state}, \widehat{k}_{r-1})$
    $\texttt{SubBytes}(\texttt{state})$
    $\texttt{MixColumns}(\texttt{state})$
$\texttt{ShiftRows}(\texttt{state})$
$\texttt{AddRoundKey}(\texttt{state}, \widehat{k}_9)$
$\texttt{SubBytes}(\texttt{state})$
$\texttt{AddRoundKey}(\texttt{state}, k_{10})$
$\textit{ciphertext} \leftarrow \texttt{state}$

Look-up tables

Goal

Combine certain steps and compute all possible outputs into a table.

- T-box tables → combines ShiftRow and AddRoundKey.
- Tyi tables → map the T-box output to the MixColumns computation .
- XOR tables → used to perform xor computations for the Tyi tables.

$$
\begin{aligned}
&\texttt{state} \leftarrow \mathit{plaintext} \\
&\text{for } r = 1 \ldots 9 \\
&\qquad \texttt{ShiftRows} \\
&\qquad \texttt{TBoxesTyiTables} \\
&\qquad \texttt{XORTables} \\
&\texttt{ShiftRows} \\
&\texttt{TBoxes} \\
&\mathit{ciphertext} \leftarrow \texttt{state}
\end{aligned}
$$

UNIVERSITÉ DE RENNES 1

Internal encoding

Security

We have no protection against key extraction attacks, We need to enforce

- *confusion* between the key and the tables.
- *diffusion* between the input and output.

- Random Bijection $\rightarrow$ use non linear encoding on each tables achieve *confusion*.
- Mixing Bijection $\rightarrow$ use network of linear encoding on tables to achieve *diffusion* .

Side Channel & Practical Attack

Side Channel & Practical Attack

Some definitions :

- What is Side Channel in White-Box Cryptography context ?
- What is Differential Computation Analysis ?

Side Channel & Practical Attack - DCA

- DPA (Physical) — DCA (Logical)

- Intermediate state: $I(P_i, k)$
  Trace of intermediate state: $L(I(P_i, k) + y)$

- 3 values are available by the attacker:
  $P_i \quad T_i \quad C_i$

- Analyse many $L(I(P_i, k) + y)$ of T_i with Hamming weights method

Side Channel & Practical Attack - Algorithm

Algorithm for first byte

- Gather traces $\rightarrow$ serializing read addresses ,their values.
- Modeling the leak $\rightarrow \mathrm{Sel}(pe, kh, j) := \mathrm{SBox}(pe \oplus kh)[j] = b \in \{0, 1\}$
- Sort traces $\rightarrow$ two set of traces A_1 , A_0 depending of b.
- Compute correlation between b and traces $\rightarrow H(\overline{A}_0$ - $\overline{A}_1)$
- Find best j , repeat two previous step for $1 \leq j \leq 8$ and set score kh to H_j .
- Find best kh , repeat previous step for $1 \leq kh \leq 256$ return kh with highest H_{kh}

UNIVERSITÉ DE RENNES 1

Side Channel & Practical Attack - effect of encoding

Effect of encoding

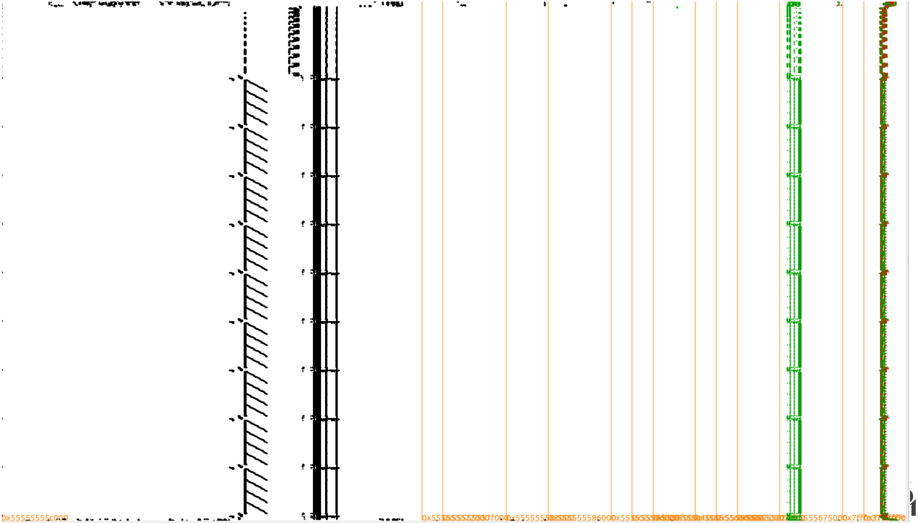
Combination of Linear and Non-Linear Encodings make H_{kh} converging to 0, 0.25, 0.5, 0.75 or 1

- Modification $\rightarrow$ ranked kh according to the results

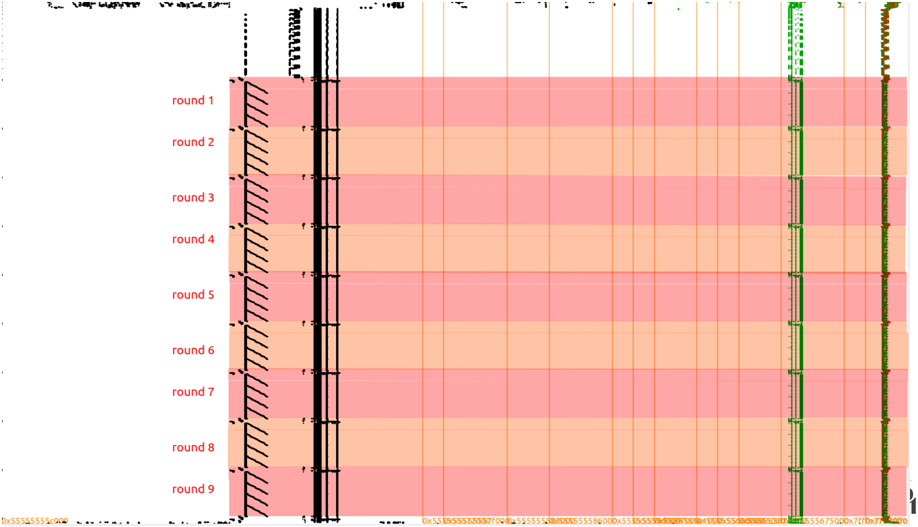
Gathering Traces & Trace analyse

Gathering Traces & Traces Analyses

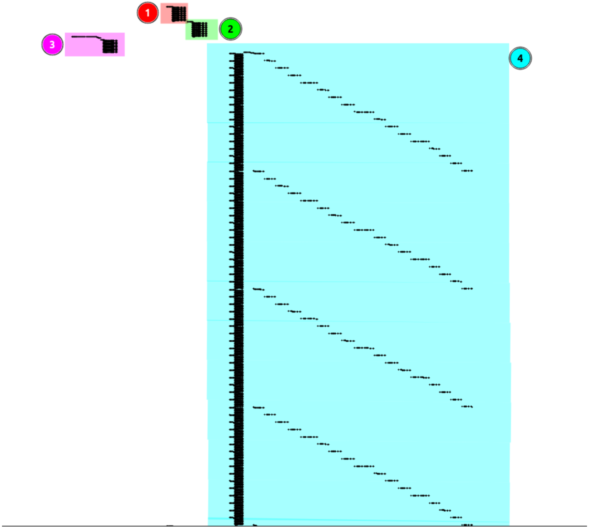
Finding pattern for an classical AES:



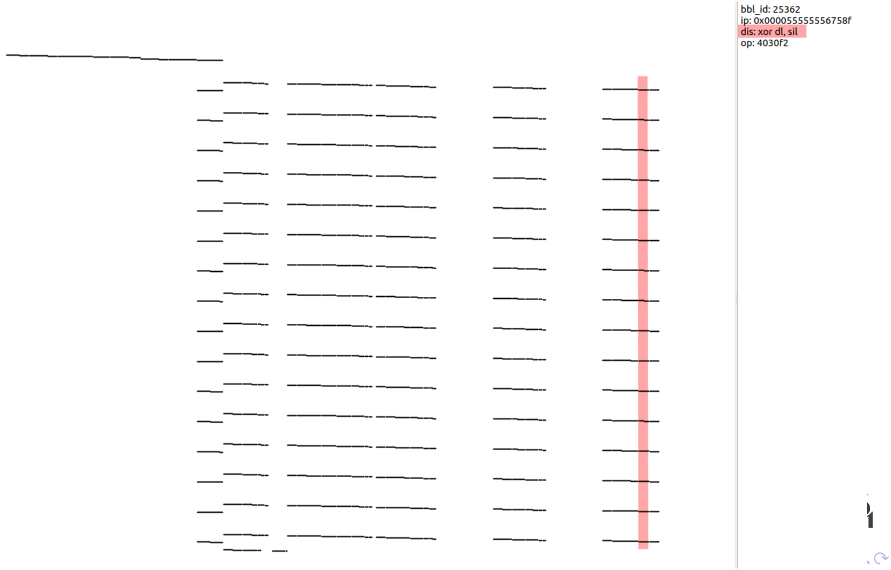
Gathering Traces & Traces Analyses - Finding pattern in classical AES



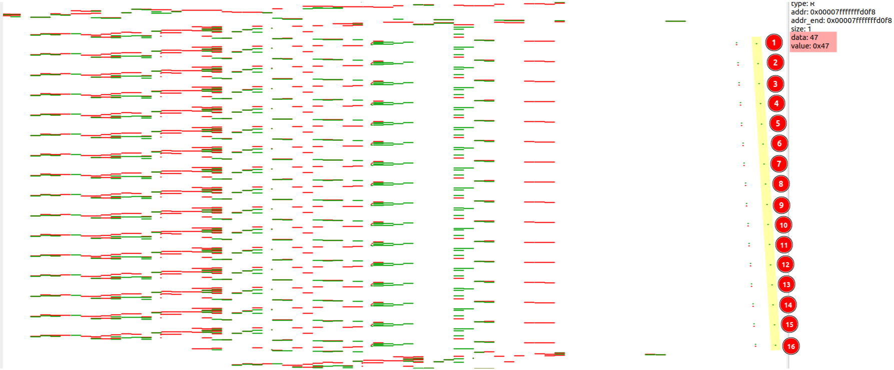
Gathering Traces & Traces Analyses - Finding pattern in classical AES



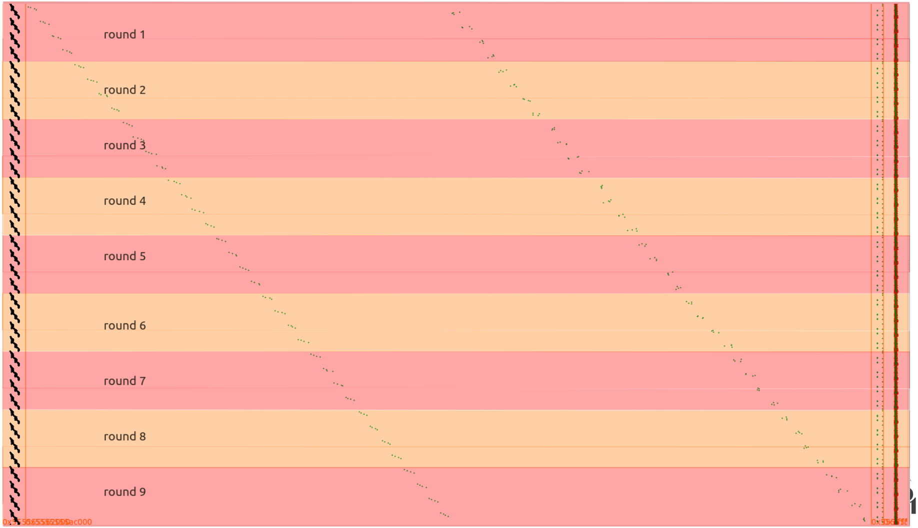
Gathering Traces & Traces Analyses - Finding pattern in classical AES



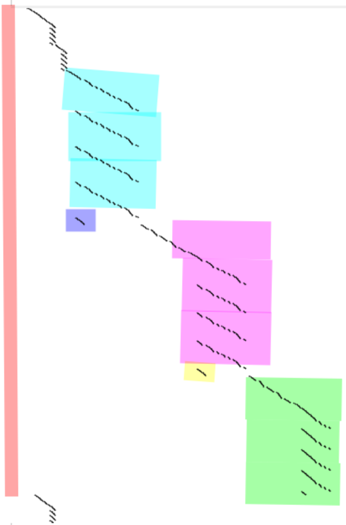
Gathering Traces & Traces Analyses - Finding pattern in classical AES



Gathering Traces & Traces Analyses - Finding pattern in White-Box AES

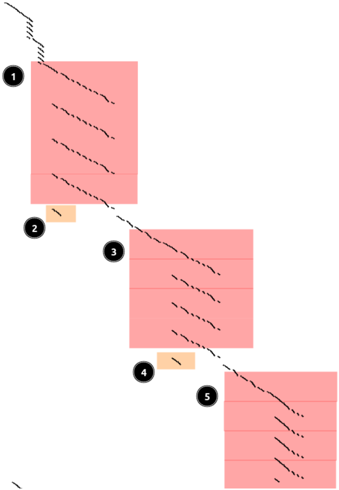


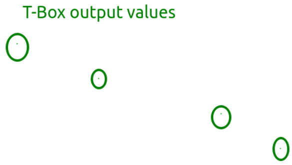
Gathering Traces & Traces Analyses - Finding pattern in White-Box AES



```rust
pub fn encryption_block(bytes: &[u8; 16]) -> [u8; 16] {
    let mut state: [u8; 16] = *bytes; //init of tab with i=[0,1,2,3,4,5,6,7,8,9,10,

    for i in 0..9 {// number round (for an aes 128 is 10)
        shift_rows(&mut state);//for round 2, take 0eme, 5eme, 10eme and 15eme valu

        for g in 0..4 {
            let mut ty_i: [[u8; 4]; 4] = [[0; 4]; 4];//init
            let mut tmp: [[u8; 4]; 4] = [[0; 4]; 4];//init

            for x in 0..4 {//is (L invert) -> Tx -> Ty and MB step
                ty_i[x] = tylm_boxes[i][g][x][state[g * 4 + x] as usize];
            }
//here

            //three XOR before encapsulation with invert of MB an L3
            let xor_ty_i_0 = xor_32b(i, g * 3, &xor_table, &ty_i[0], &ty_i[1]);
            let xor_ty_i_1 = xor_32b(i, g * 3 + 1, &xor_table, &ty_i[2], &ty_i[3]);
            let xor_result_ty_i = xor_32b(i, g * 3 + 2, &xor_table, &xor_ty_i_0, &x

            for r in 0..4 {//invert of MB and L2+1
                tmp[r] = ml_box[i][g][xor_result_ty_i[r] as usize][r];
            }

            //three XOR after encapsulation with invert of MB an L3
            let xor_tmp_0 = xor_32b(i, g * 3, &xor_ml_tables, &tmp[0], &tmp[1]);
            let xor_tmp_1 = xor_32b(i, g * 3 + 1, &xor_ml_tables, &tmp[2], &tmp[3])
            let xor_result_tmp = xor_32b(i, g * 3 + 2, &xor_ml_tables, &xor_tmp_1,

            for res_xor in 0..4 {
                state[g * 4 + res_xor] = xor_result_tmp[res_xor];
            }
        }
    }
```

Gathering Traces & Traces Analyses - Finding pattern in White-Box AES





Result & Conclusion

- Problem : when H_{kh} near 0.2 . Two possible convergence : 0 or 0.25
- Solution : key ranking now testing with different convergence when H_{kh} near 0.2

Annexes

Annexes-Random Bijection

Definition

From T we get a new table T' using two random bijection f,g.

$$T' = g \circ T \circ f^{-1}$$

Restriction

- Split in nibbles and cancel for the Xor operation.
- Cancel each other if there output is feed as an input.
- Very first and the very last tables are not encoded $\rightarrow$ External encoding

Annexes-Mixing Bijection

Definition

From T we get a new table T' using two random linear bijection A, L.
$T' = A \circ T \circ L^{-1}$

Restriction

- Commute with the XOR-operation.
- Apply to words that areinput or output of an XOR-network
- applied before the non-linear encodings

Annexes-Mixing Bijection