Fast Side-Channel Key-Recovery on Dumbo

Louis Vialar

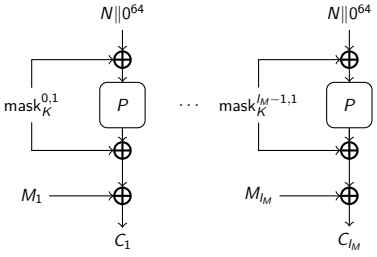
EPFL

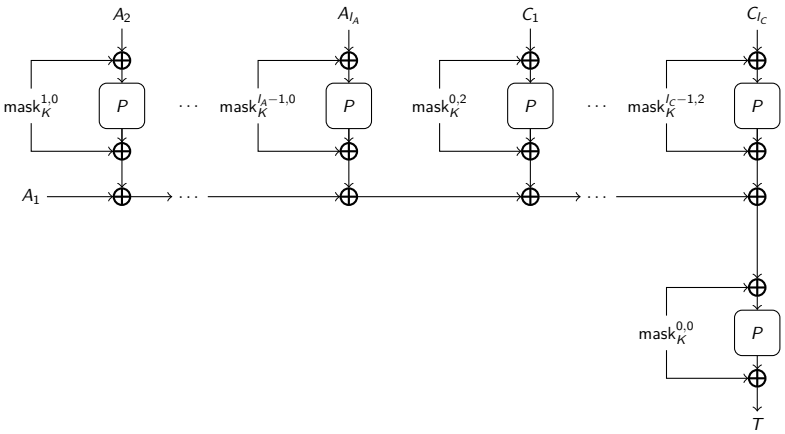
Kudelski Security Research Team

May 2022

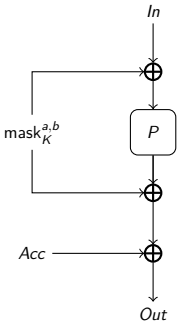
The Design of Elephant-160 (Dumbo)

Encryption

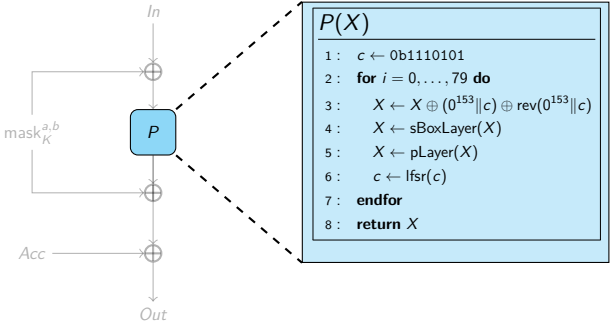



A_2
A_{l_A}
C_1
C_{l_C}
$\text{mask}_K^{1,0}$
$\text{mask}_K^{l_A-1,0}$
$\text{mask}_K^{0,2}$
$\text{mask}_K^{l_C-1,2}$
$\text{mask}_K^{0,0}$
P
A_1
T

Building blocks



Building blocks



Building blocks

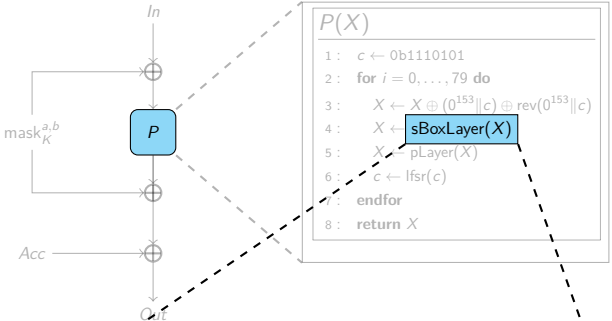
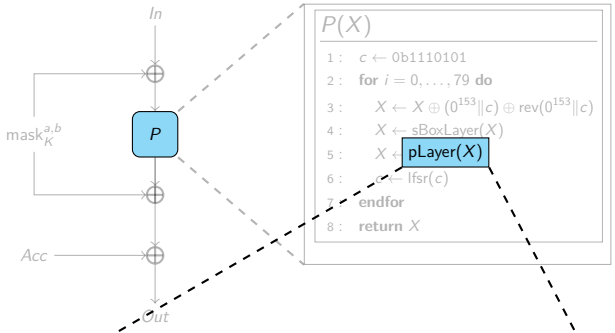


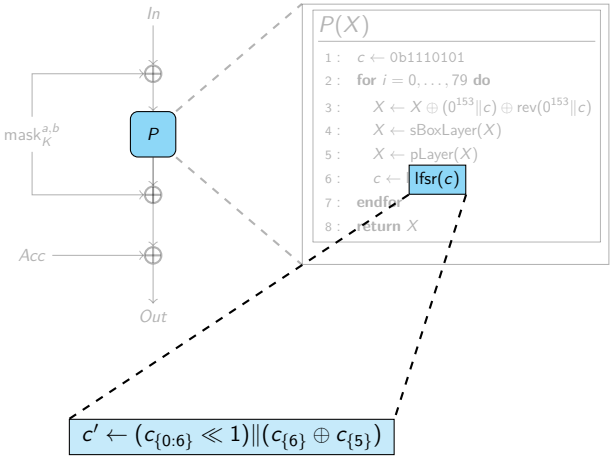
Table 1: Elephant S-Box

X	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
S(X)	E	D	B	0	2	1	4	F	7	A	8	5	9	C	3	6

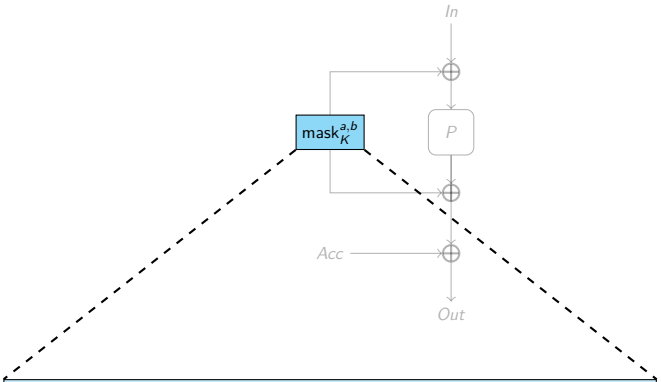
Building blocks



Building blocks

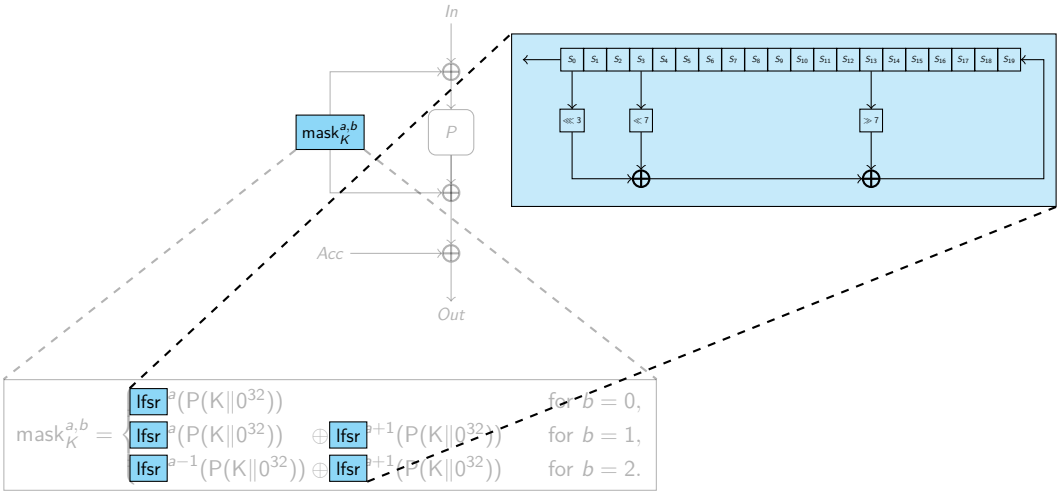


Building blocks



$$\mathrm{mask}_K^{a,b} = \begin{cases} \mathrm{lfsr}^a(\mathrm{P}(\mathrm{K}\|0^{32})) & \text{for } b = 0, \\ \mathrm{lfsr}^a(\mathrm{P}(\mathrm{K}\|0^{32})) \oplus \mathrm{lfsr}^{a+1}(\mathrm{P}(\mathrm{K}\|0^{32})) & \text{for } b = 1, \\ \mathrm{lfsr}^{a-1}(\mathrm{P}(\mathrm{K}\|0^{32})) \oplus \mathrm{lfsr}^{a+1}(\mathrm{P}(\mathrm{K}\|0^{32})) & \text{for } b = 2. \end{cases}$$

Building blocks



Associated Data is authenticated with

$$\mathsf{mask}_K^{a,0} = \mathsf{lfsr}^a(\mathsf{P}(\mathsf{K} \| 0^{32}))$$

Associated Data is authenticated with

$$\mathsf{mask}_K^{a,0} = \mathsf{lfsr}^a(\mathsf{P}(\mathsf{K} \| 0^{32}))$$

lfsr is invertible: we can recover $\mathsf{mask}_K^{a-1,0}$ from $\mathsf{mask}_K^{a,0}$.

Associated Data is authenticated with

$$\mathsf{mask}_K^{a,0} = \mathsf{lfsr}^a(\mathrm{P}(\mathrm{K}\|0^{32}))$$

lfsr is invertible: we can recover $\mathsf{mask}_K^{a-1,0}$ from $\mathsf{mask}_K^{a,0}$.

P is invertible: we can recover $K\|0^{32}$ from $\mathrm{P}(\mathrm{K}\|0^{32})$.

Attack idea

Recover $\mathsf{mask}_K^{1,0}$ **by CPA on the authentication of Associated Data. Invert** *lfsr* **and** P **to recover** K**.**

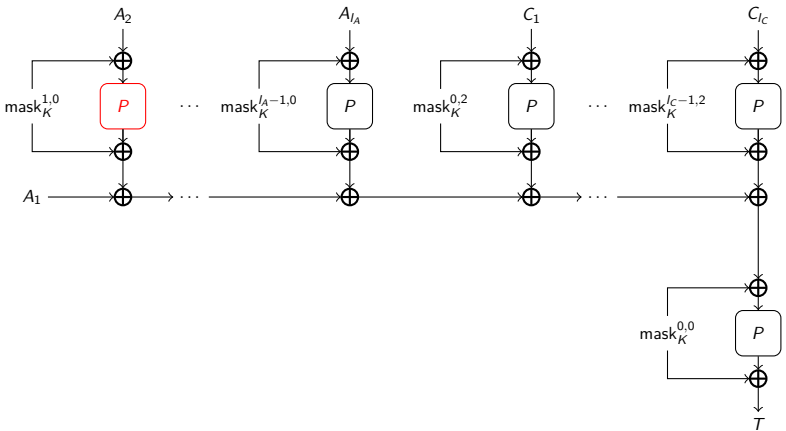


Figure 1: Dumbo Authentication With Attack Point

We attack the sBoxLayer of P by CPA.

$P(X)$

1 : $c \leftarrow \texttt{0b1110101}$
2 : **for** $i = 0, \ldots, 79$ **do**
3 : $\quad X \leftarrow X \oplus (0^{153} \| c) \oplus \mathrm{rev}(0^{153} \| c)$
4 : $\quad X \leftarrow \mathrm{sBoxLayer}(X)$
5 : $\quad X \leftarrow \mathrm{pLayer}(X)$
6 : $\quad c \leftarrow \mathrm{lfsr}(c)$
7 : **endfor**
8 : **return** X

Our CPA attack

Intuition: changing a bit value does not consume the same power if the bit is 1 or 0.

Intuition: changing a bit value does not consume the same power if the bit is 1 or 0.

$\Longrightarrow$ **Assumption**: the power consumption of a microchip depends on the data it processes.

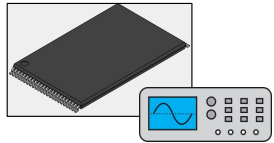
Model(a, k, i)

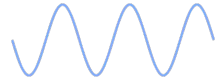
1 : $S \leftarrow a \oplus k$
2 : // First operation of the first round: add c to the first and last bytes
3 : **if** $i = 0$ **then**
4 : $S \leftarrow S \oplus$ 0x75 // 0b1110101
5 : **elseif** $i = 19$ **then**
6 : $S \leftarrow S \oplus$ 0xae // 0b1110101 in reverse order
7 : **endif**
8 : // Second operation of the first round: sBoxLayer
9 : $S \leftarrow \mathrm{SBox}(S)$
10 : **return** $\mathrm{HammingWeight}(S)$

Idea

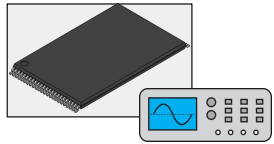
If we can know the real value of Model(a, k, i) for any a, k, i, we can compute the model with all 256 possibilities for k until we find the one that always matches the real value.

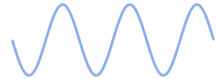
MkOPWdzNblDDaIF+fA





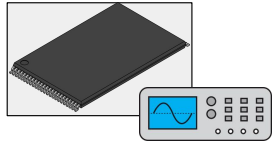
QroQNMe2gzpJDc5LPt

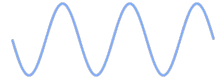




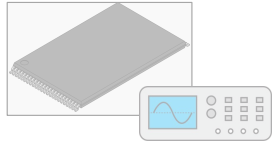
...

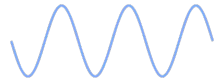
fTnwamMcjL9WzEl5Ad

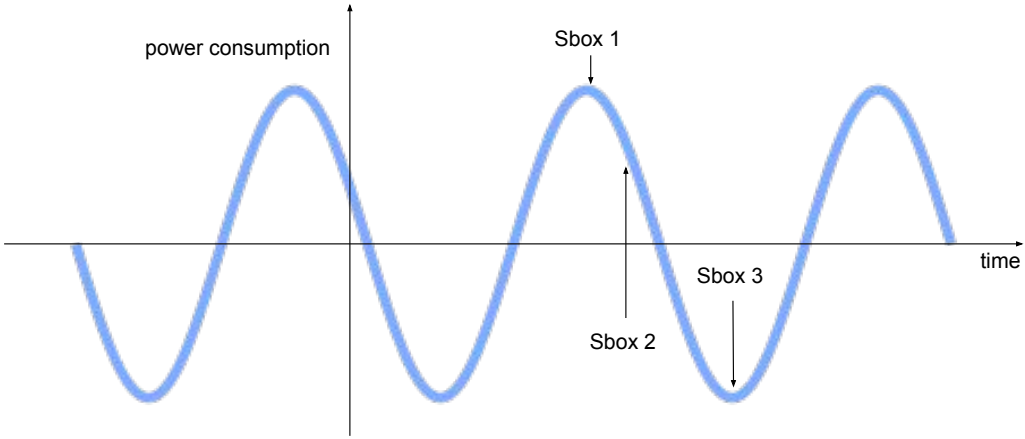


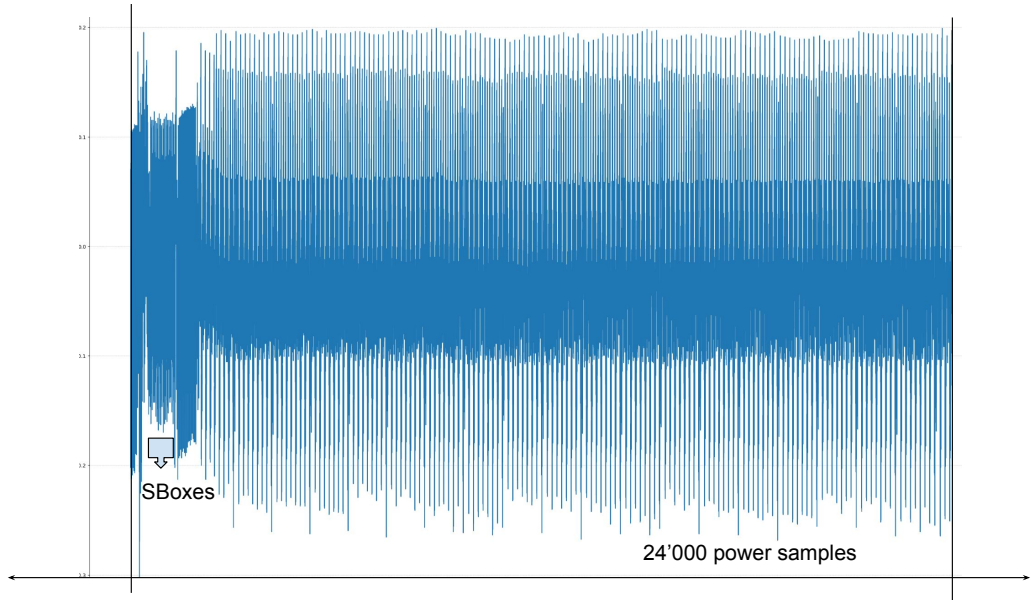


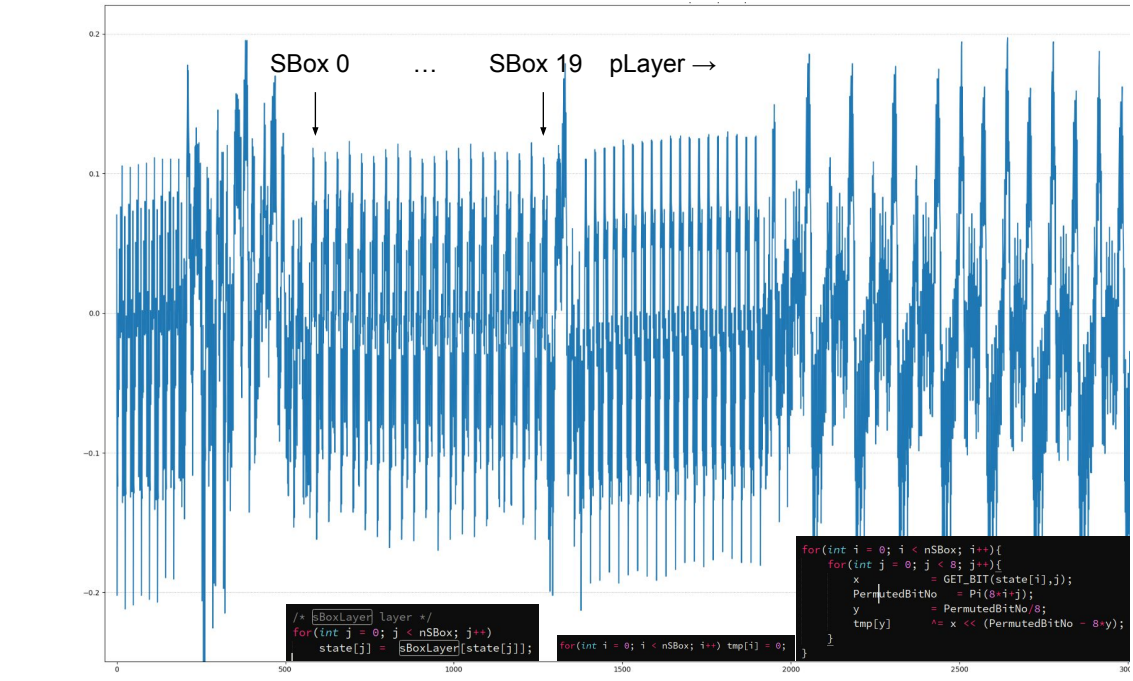
MkOPWdzNblDDaIF+fA

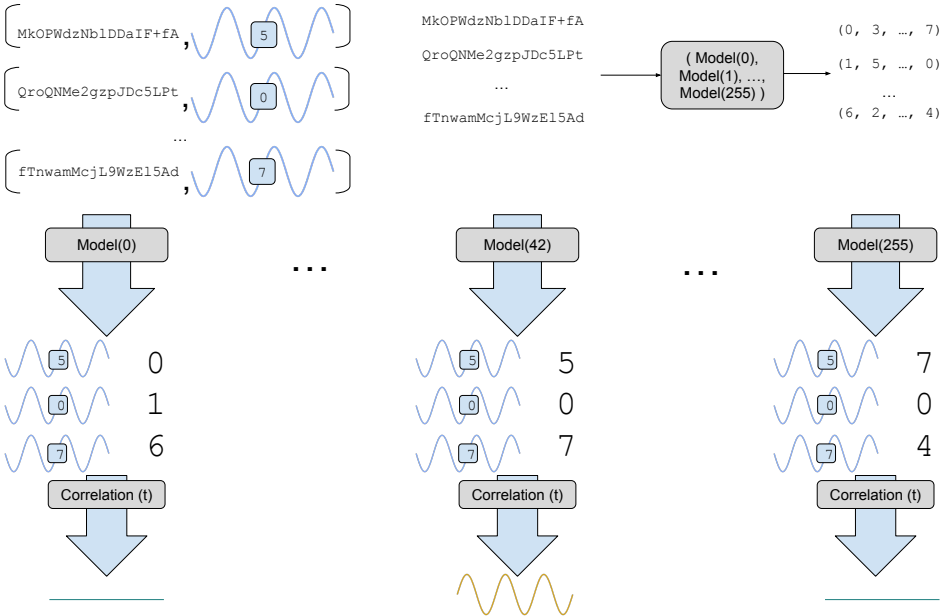


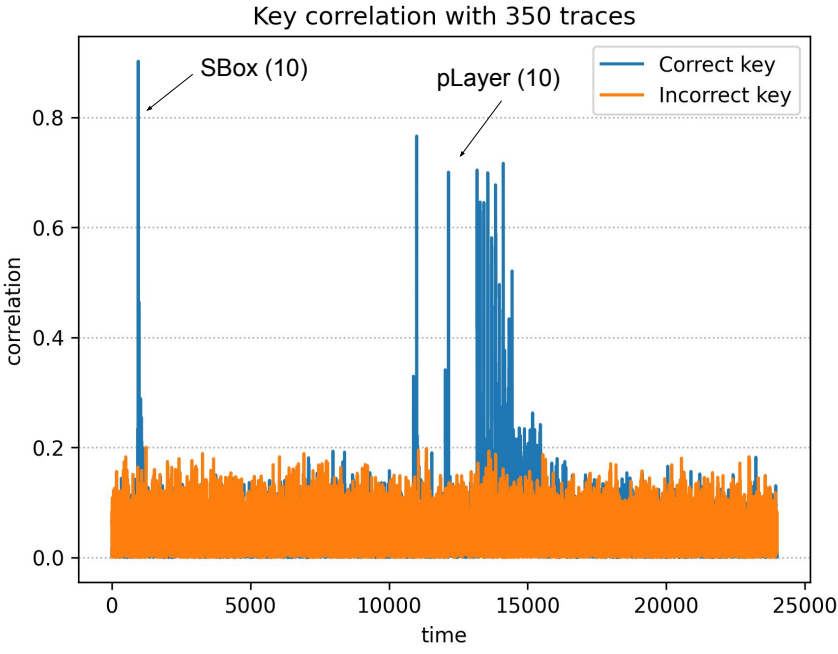


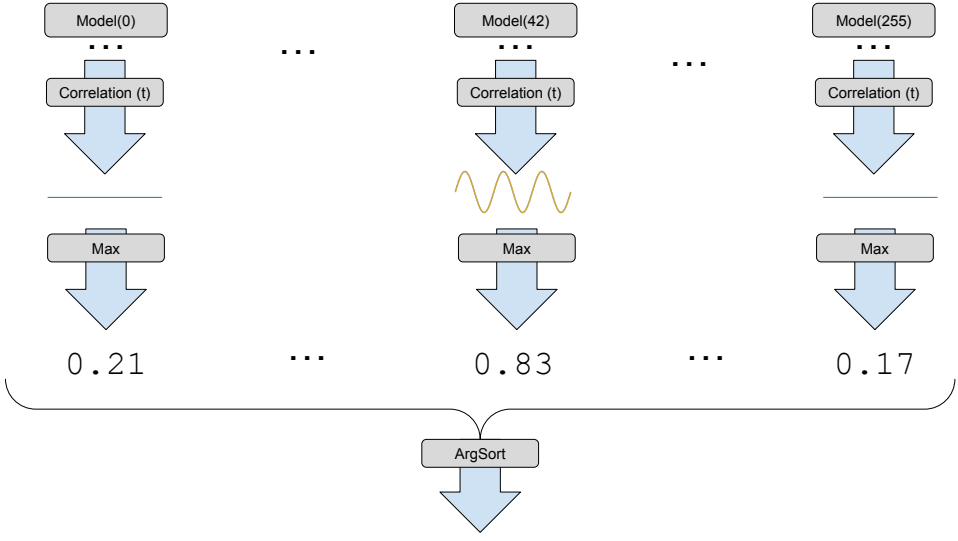



SBoxes
24'000 power samples


SBox 0
…
SBox 19
pLayer →
0.2
0.1
0.0
−0.1
−0.2
0
500
1000
1500
2000
2500
/* sBoxLayer layer */
for(int j = 0; j < nSBox; j++)
state[j] = sBoxLayer[state[j]];
for(int i = 0; i < nSBox; i++) tmp[i] = 0;
for(int i = 0; i < nSBox; i++){
for(int j = 0; j < 8; j++){
x = GET_BIT(state[i],j);
PermutedBitNo = Pi(8*i+j);
y = PermutedBitNo/8;
tmp[y] ^= x << (PermutedBitNo - 8*y);
}
}


MkOPWdzNblDDaIF+fA
5
QroQNMe2gzpJDc5LPt
0
fTnwamMcjL9WzEl5Ad
7
MkOPWdzNblDDaIF+fA
QroQNMe2gzpJDc5LPt
fTnwamMcjL9WzEl5Ad
(Model(0), Model(1), …, Model(255))
(0, 3, …, 7)
(1, 5, …, 0)
(6, 2, …, 4)
Model(0)
Model(42)
Model(255)
0
1
6
5
0
7
7
0
4
Correlation (t)
Correlation (t)
Correlation (t)


Key correlation with 350 traces
SBox (10)
pLayer (10)
Correct key
Incorrect key
correlation
0.0
0.2
0.4
0.6
0.8
0
5000
10000
15000
20000
25000
time


Model(0)
Model(42)
Model(255)
Correlation (t)
Correlation (t)
Correlation (t)
Max
Max
Max
0.21
0.83
0.17
ArgSort
[42, 37, 71, 90, 12, …]

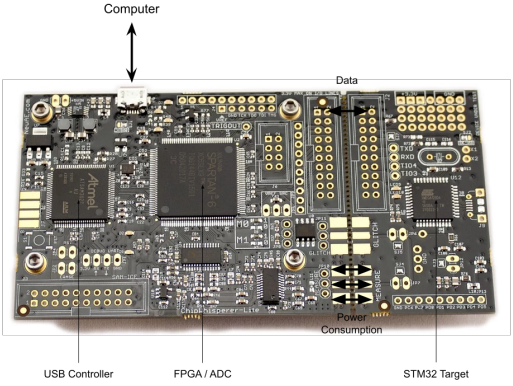


Figure 2: The CW Lite ARM Board

Common computers (i7-4790K, i7-8565U) for the CPA.

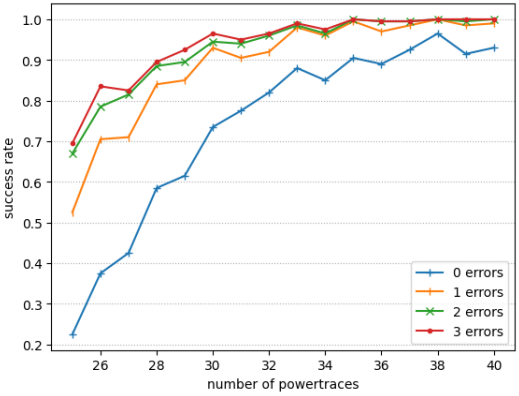


Figure 3: Attack Success Rate

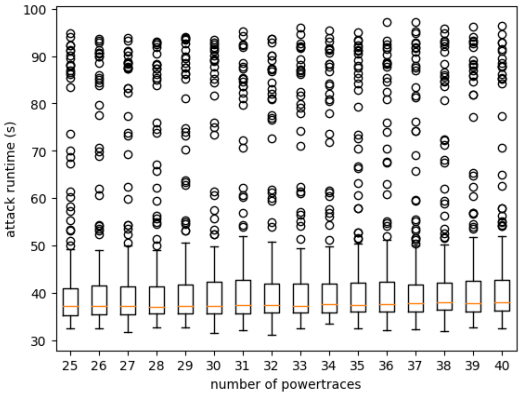


Figure 4: Attack Runtime

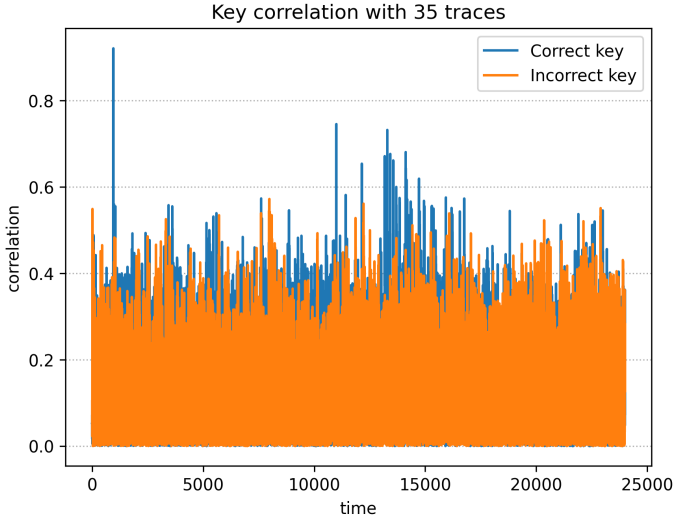


Figure 5: Example of correlation curve (35 traces)

Countermeasures were developed for hardware implementations by Abdulgadir et al. in **Side-Channel Resistant Implementations of Three Finalists of the NIST Lightweight Cryptography Standardization Process: Elephant, TinyJAMBU, and Xoodyak**.

Countermeasures were developed for hardware implementations by Abdulgadir et al. in **Side-Channel Resistant Implementations of Three Finalists of the NIST Lightweight Cryptography Standardization Process: Elephant, TinyJAMBU, and Xoodyak**.

Small spoiler: it has some impact on performance.

Conclusion

Conclusion

- Efficient CPA (< 40 traces) on Elephant
- Attacks the **reference software implementation**
  - Not protected, not optimized for our target hardware
  - Other unprotected implementations will likely require more traces, but still be vulnerable
- Countermeasures exist on hardware versions and have a performance impact

Questions?

Graphics Attribution

https://commons.wikimedia.org/wiki/File:CPT-sound-nyquist-thereom-raw.svg
https://commons.wikimedia.org/wiki/File:TSOP-48_Blank.svg
https://commons.wikimedia.org/wiki/File:Symbol_oscilloscope.svg