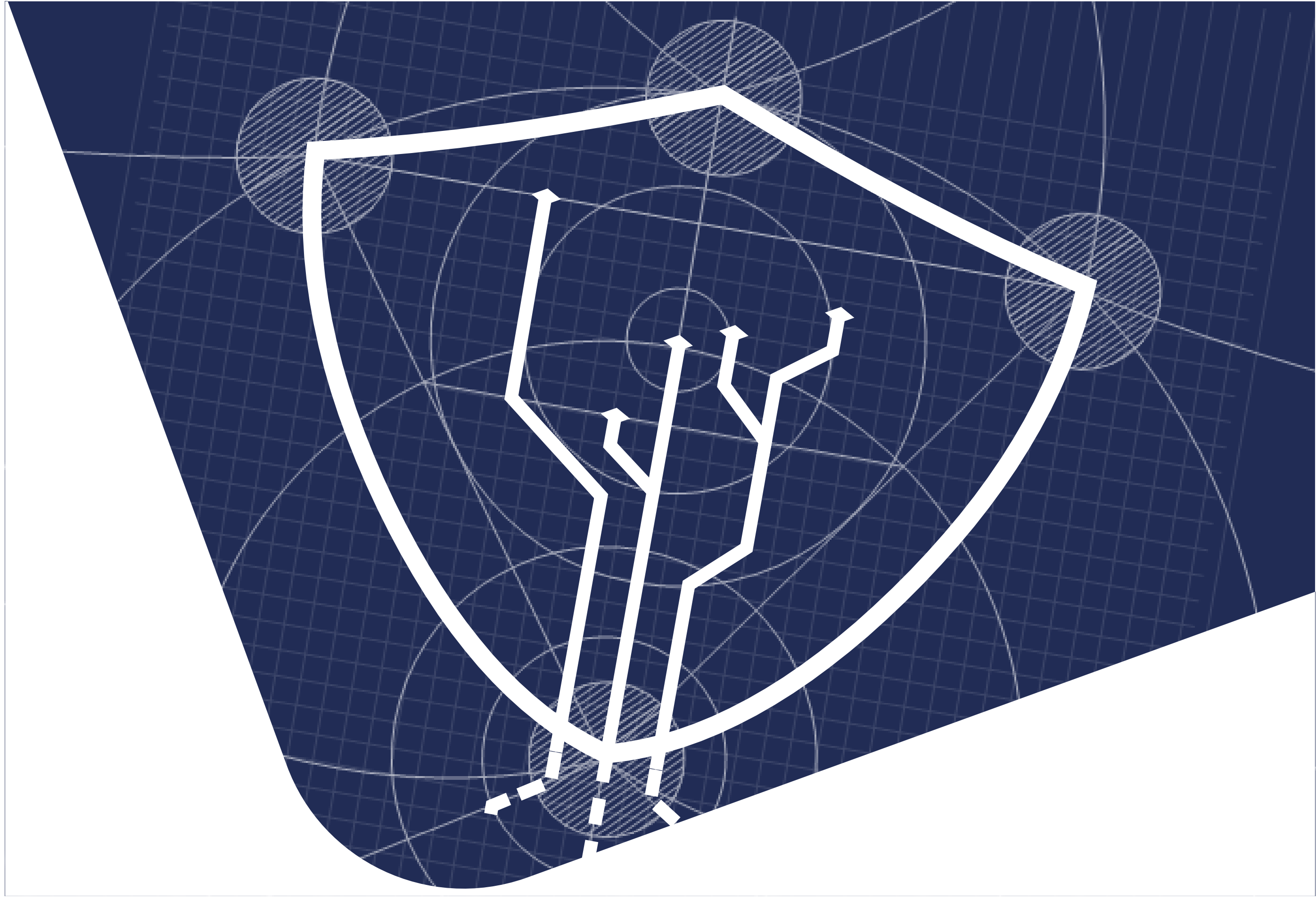
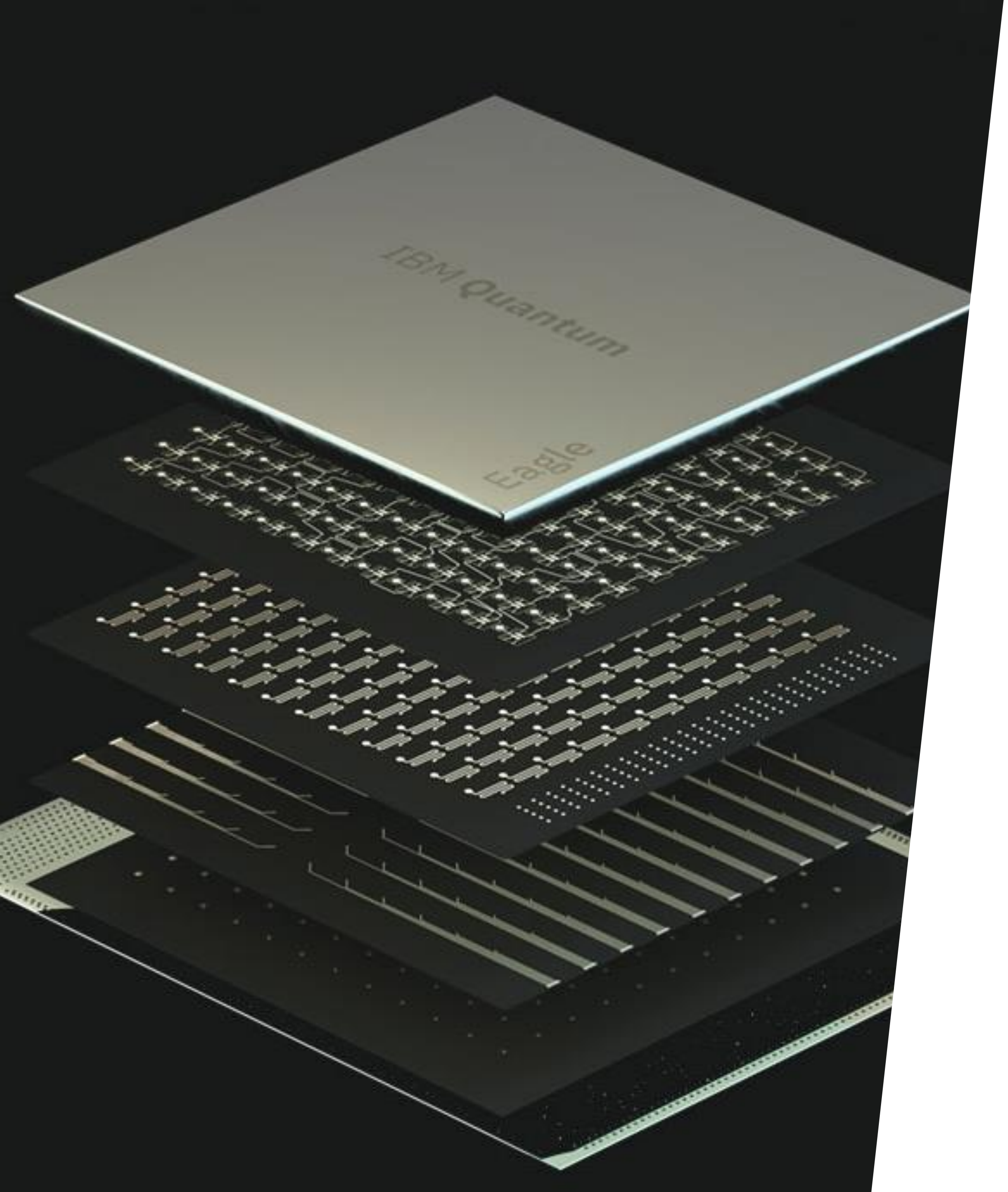


COSADE 2022, KU Leuven, Belgium

Tuesday, 12 April 2022

Single-trace Clustering Power Analysis of The Point Swapping Procedure in the Three Point Ladder of Cortex-M4 SIKE

Aymeric Genêt, Novak Kaluđerović



https://research.ibm.com/blog/127-qubit-quantum-processor-eagle

Introduction

Quantum computers

- IBM Eagle: 127-qubit quantum computer

Post-quantum cryptography

- Resists quantum computing (in theory)

Side-channel attacks

- Attacks on electronic devices

Outline

1. **Supersingular isogeny key exchange (SIKE)**
2. **Clustering power analysis of SIKE**
3. **Countermeasure**

Section 1

Supersingular isogeny key exchange (SIKE)

Elliptic curves

Point to infinity ("zero" point)

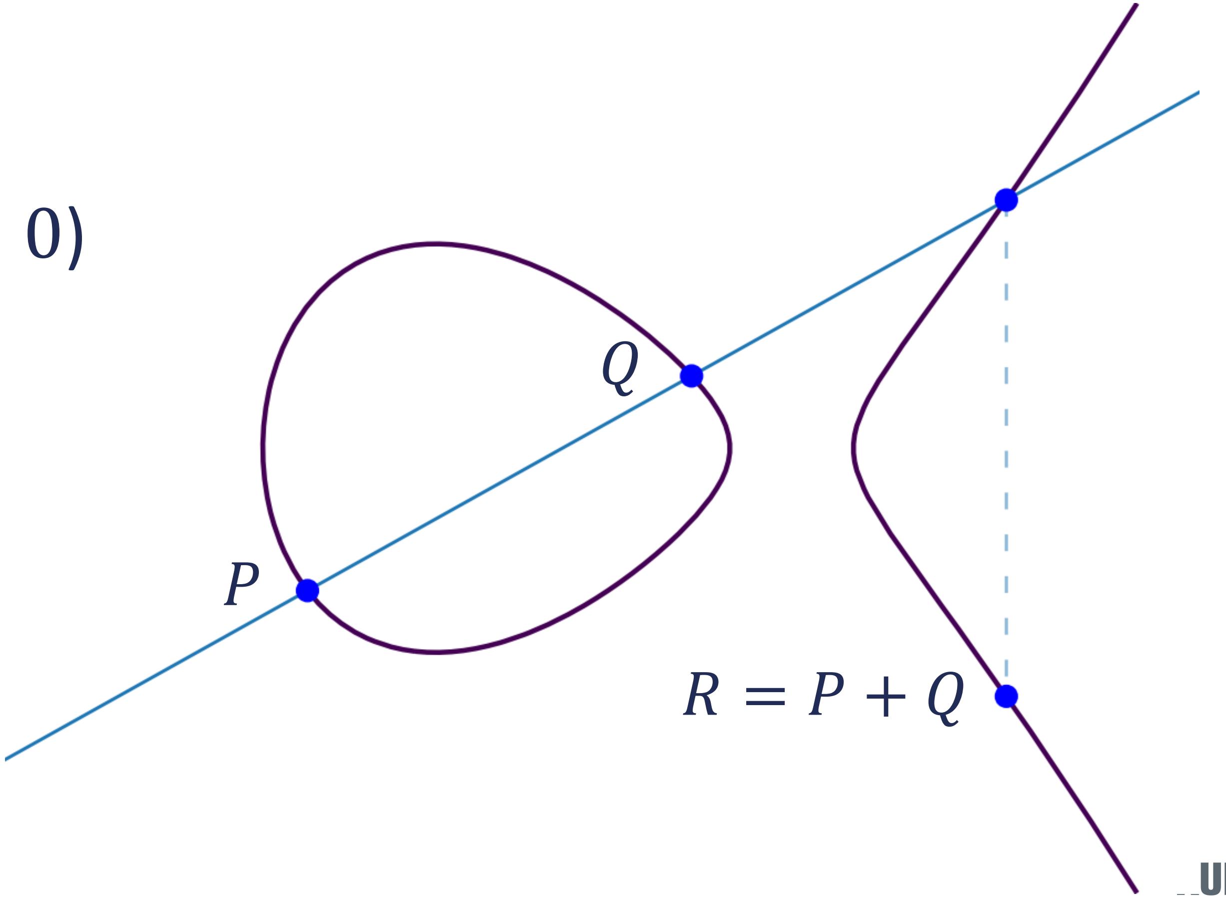
$$E_a(\mathbb{K}) = \{ (x, y) \in \mathbb{K}^2 : y^2 = x^3 + ax^2 + x \} \cup \{\mathcal{O}\}$$

Notations

- $(X : Y : Z) \leftrightarrow \left(\frac{X}{Z}, \frac{Y}{Z}\right) \quad (Z \neq 0)$
- $[n]P = \underbrace{P + P + \cdots + P}_{n}$
- $\langle P \rangle = \{[i]P : i \in \mathbb{N}\}$

Eventually, $[n]P = \mathcal{O}$ for some $n \in \mathbb{N}^*$ (the **order** of P)

As a result, this subgroup is finite



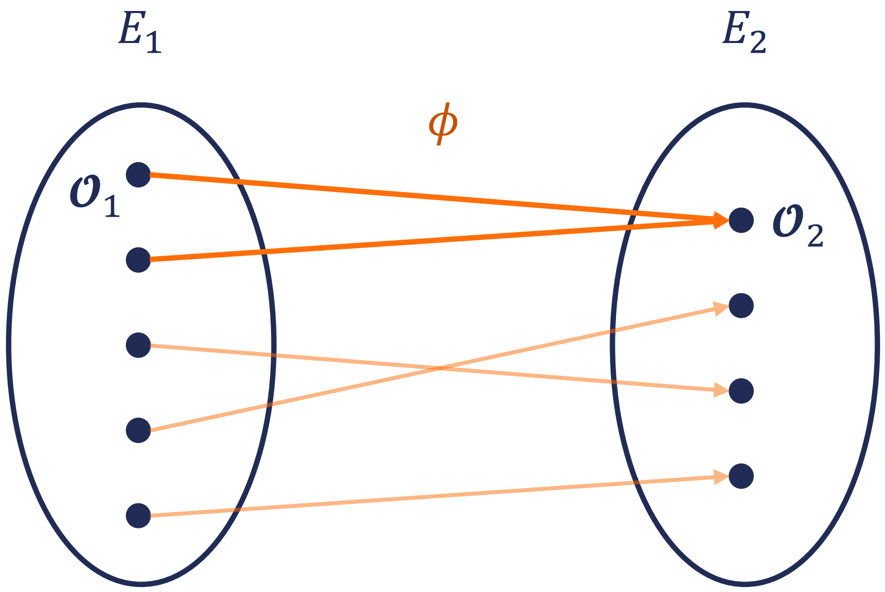
Isogenies

Isogeny: surjective mapping of finite kernel between two elliptic curves

Isogeny computation

Pick $\langle P \rangle = \{P_0, P_1, \dots\}$ so that $\phi(P_i) = \mathcal{O}$

$\Rightarrow \quad \phi : E \to E/\langle P \rangle$ is **defined** by that!



Degree

$$\deg(\phi) = \#\langle P \rangle = \mathrm{ord}(P)$$

Smallest $n \in \mathbb{N}^*$ s.t. $[n]P = \mathcal{O}$

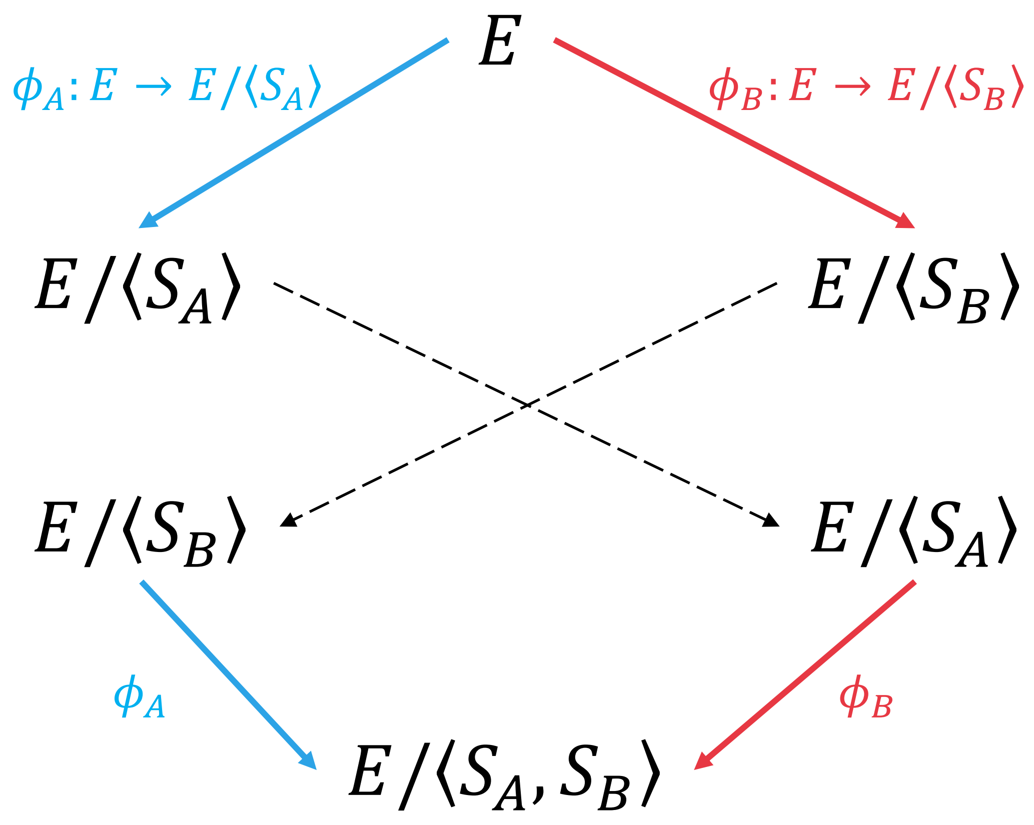
Hard problem: Given E_a and $E_a/\langle P \rangle$, find $\phi_f : E_a \to E_a/\langle P \rangle$

Especially when degree is large…

Supersingular isogeny Diffie–Hellman (SIDH)

Party computations

1. $S = P + [sk]Q$ of order $\begin{cases} 2^{e_A} \text{ (Alice)} \\ 3^{e_B} \text{ (Bob)} \end{cases}$
2. Obtain ϕ from $\langle S \rangle$
3. Send $E/\langle S \rangle, \phi(P), \phi(Q)$



SIKE SIDH + Fujisaki–Okamoto

Transform that makes sure that no one cheats

Three point ladder

Given $(Q, P, Q-P)$, compute efficiently $S = P + [sk]Q$ on E_a

Three point ladder (simplified)

```
for (i = 0; i < nbits; i++) {
    bit = sk[i];
    swap = bit ^ prevbit;
    prevbit = bit;

    swap_points(P, QmP, swap);
    xDBLADD(Q, QmP, P, Ea);
}
```

$(Q, P, Q-P) \leftarrow ([2]Q, P+Q, Q-P)$

swap_points(P, Q, swap) (simplified)

```
mask = 0 - swap;

for (i = 0; i < NWORDS_FIELD; i++) {
    temp = (mask & (P->X[i] ^ Q->X[i]);
    P->X[i] = temp ^ P->X[i];
    Q->X[i] = temp ^ Q->X[i];
    ⋮
}
```

$\textbf{mask} = \begin{cases} \texttt{0x00000000} \text{ if swap} = 0 \\ \texttt{0xFFFFFFFF} \text{ if swap} = 1 \end{cases}$

KUDELSKI IOTHINGS

Photo by Alexander Dummer (Pexels)

Section 2

Clustering power analysis of SIKE

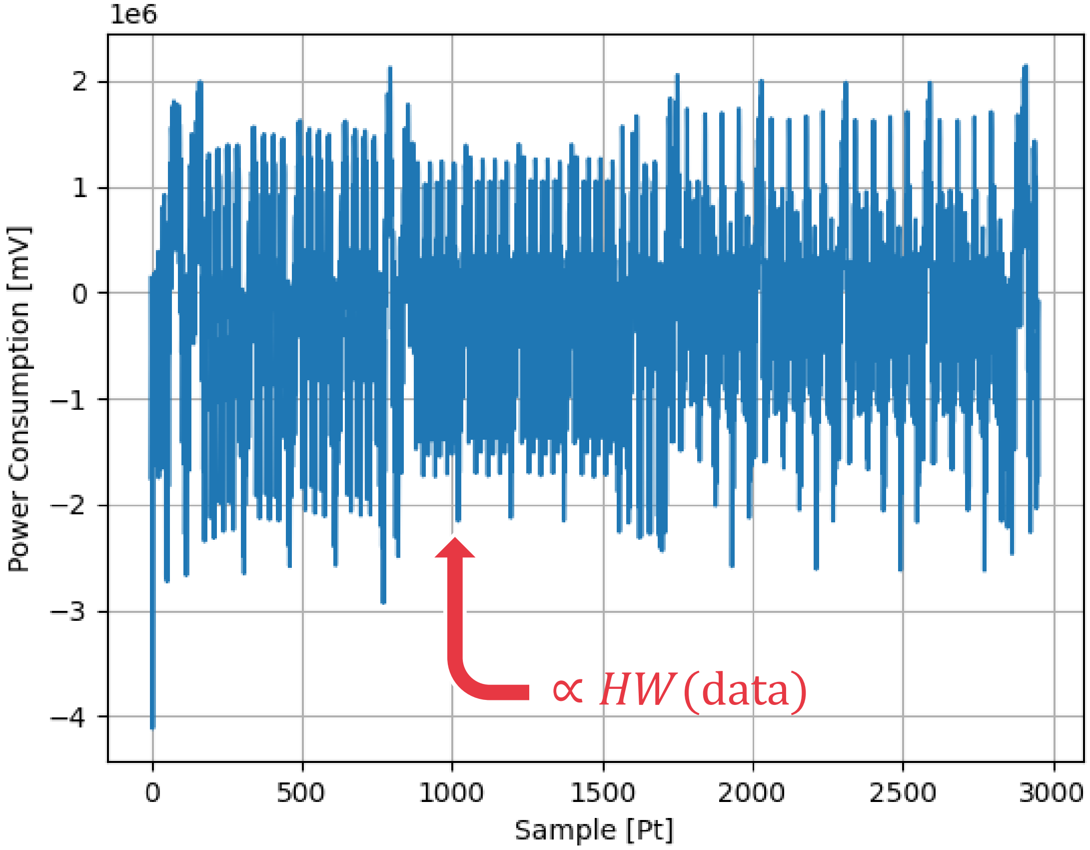
Side-channel power analysis

Passive known-text attack

Power consumption linked to **processed data**

Exploit link to recover secrets

- Simple power analysis
- Differential power analysis
- ...



Example of a **power trace**

KUDELSKI IOTHINGS

Power analysis of SIKE

Attack on the ECC part of SIKE

Double-and-add procedure

- Correlation power analysis
- Template attack
- …

$(X:Y:Z) = (\lambda X : \lambda Y : \lambda Z) \quad (\lambda \neq 0)$

Coordinate randomization

Swapping procedure

$$\text{swap}_i = sk_i \oplus sk_{i-1} = \begin{cases} 0 \\ 1 \end{cases}$$

A single trace contains the n iterations

Three point ladder (simplified)

```
for (i = 0; i < nbits; i++) {
    coord_randomize(Q, P, QmP);

    bit = sk[i];
    swap = bit ^ prevbit;
    prevbit = bit;

    swap_points(P, QmP, swap);
    xDBLADD(Q, QmP, P, Ea);
}
```


KUDELSKI IoT THINGS

swap_points Difference of Means (swap=0 vs. swap=1)

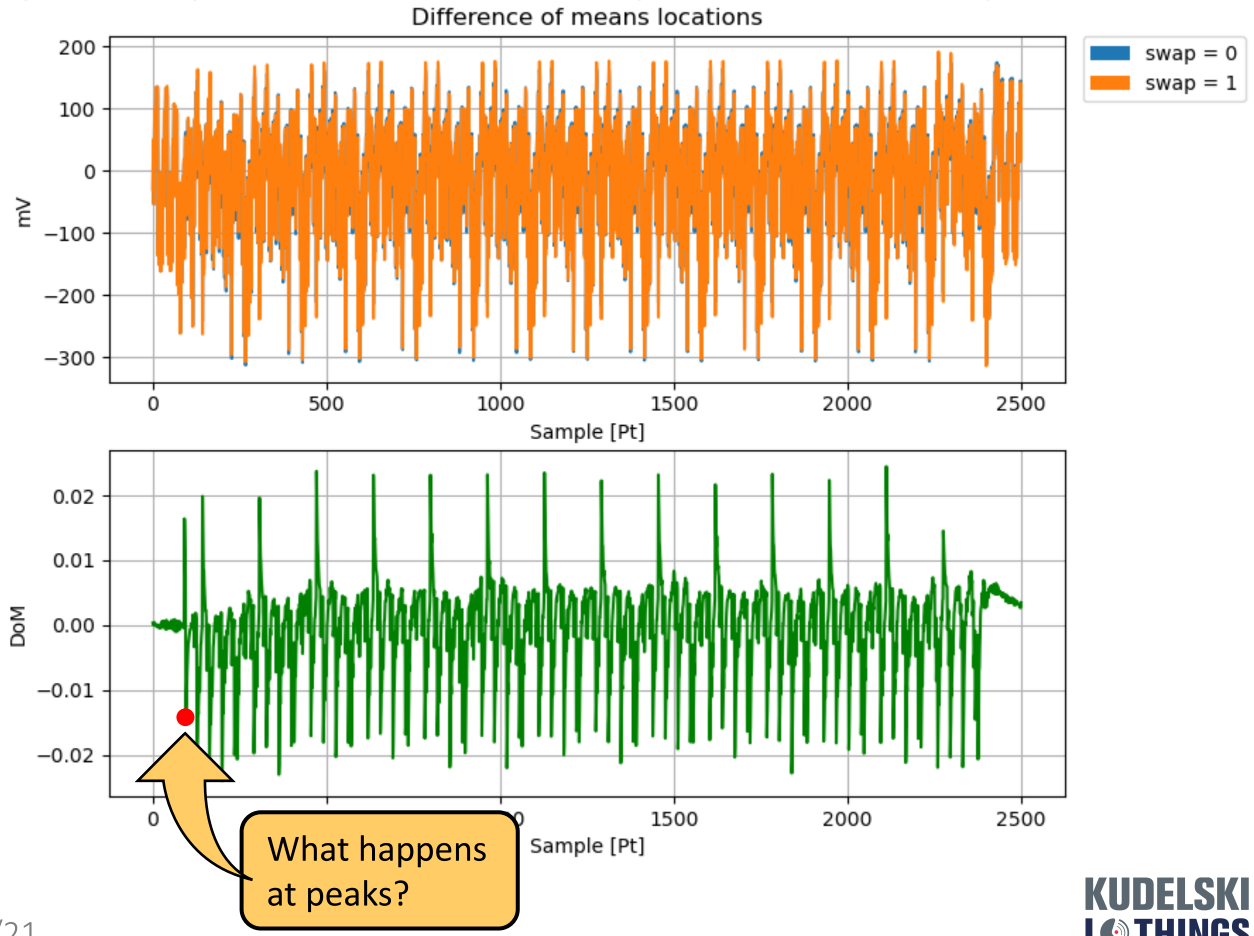
ChipWhisperer (29.54 [MHz]) + LNA (20dB) on STM32F3 (7.37 [MHz])

swap_points(P, Q, swap) (simplified)

```
mask = 0 - swap;

for (i = 0; i < NWORDS_FIELD; i++) {
    temp = (mask & (P->X[i] ^ Q->X[i]);
    P->X[i] = temp ^ P->X[i];
    Q->X[i] = temp ^ Q->X[i];
    ⋮
}
```

$n = 218$ sub-traces of swap_points



Single-sample swap_points leakage

Significant locations

- `mask = 0 – swap`
- `temp = (mask & (P->X[i] ^ Q->X[i]);`
- `P->X[i] = temp ^ P->X[i];`

Thresholding

$$\theta = \frac{\min x_i + \max x_i}{2}$$

Key validation

1. Move the threshold
2. Pick the most frequent candidate of all the locations

KUDELSKI IoTHINGS

Clustering power analysis

Cluster relative **power samples** according to their **secret value** [HIM+13, PITM14, NaC17,…]

Methodology (Perin et al. 2014)

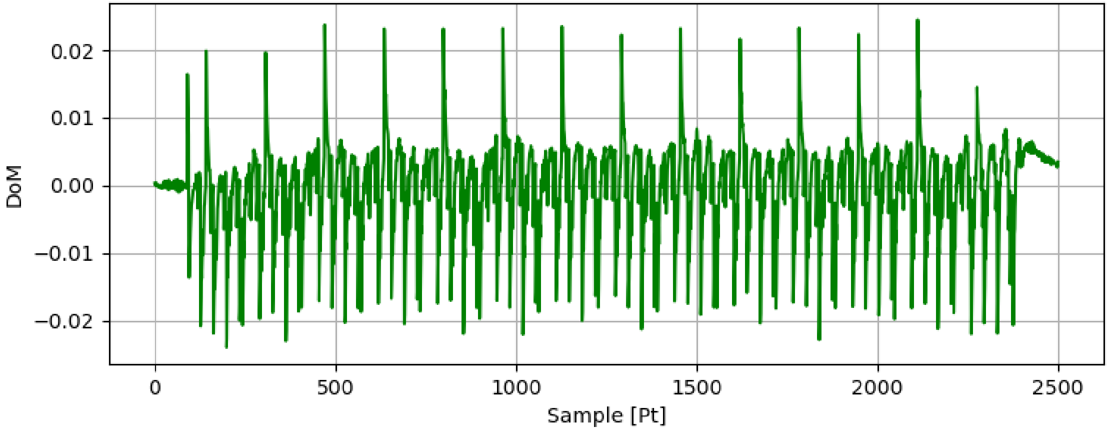
1. Leakage assessment (k-means)
2. Points of interest selection
3. Key recovery (fuzzy k-means)
4. Key validation

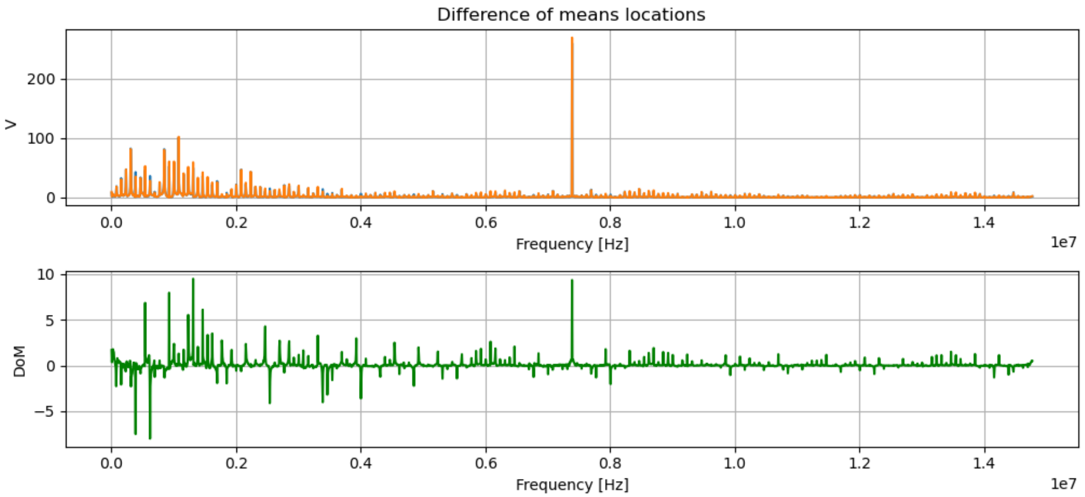
k-means algorithm	
Input: $\{x_i \in \mathbb{R}\}$ – Collection of samples	
1.	Assign x_i to cluster $0 \leq j < k$ at random
2.	**repeat**
3.	Compute the mean of all clusters μ_j
4.	Assign x_i to cluster $j = \mathrm{argmin}_{0 \leq j < k} \lvert x_i - \mu_j \rvert$
5.	**until** no μ_j changes
6.	**return** final cluster arrangement of x_i

k-means is used with $k = 2$
(i.e., swap = 0, swap = 1)

Clustering power analysis in **frequency**

Pre-process traces with a Fourier transform





Leaking frequencies

Frequencies at which clustering is 100% distinct

0.74 [MHz], 0.93 [MHz], 2.41 [MHz], 3.33 [MHz], 4.07 [MHz], 8.13 [MHz], 8.32 [MHz], 10.72 [MHz], 11.46 [MHz]

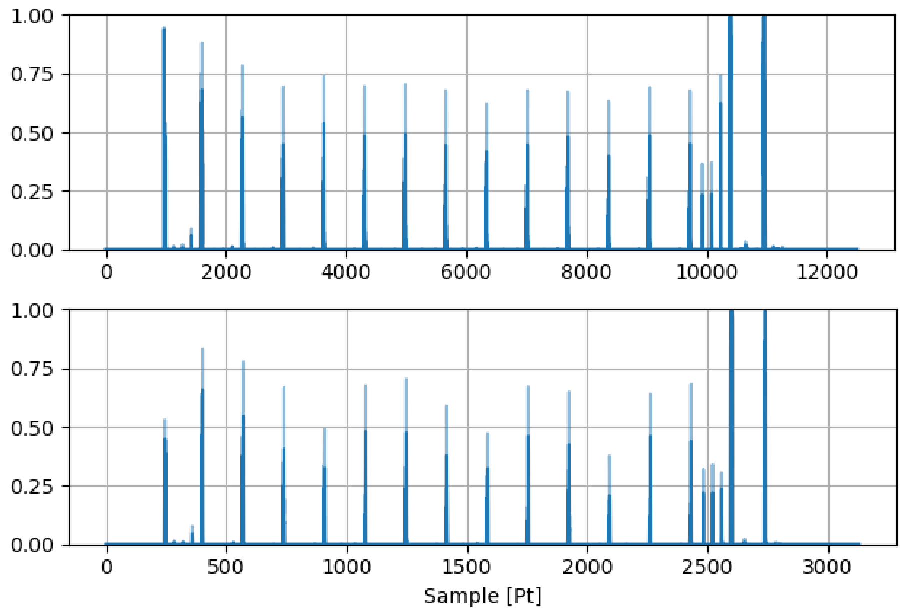
KUDELSKI IOTHINGS

Clustering power analysis with **wavelet**

Does the success rate improve with isolated frequency band?

No, but still fewer samples

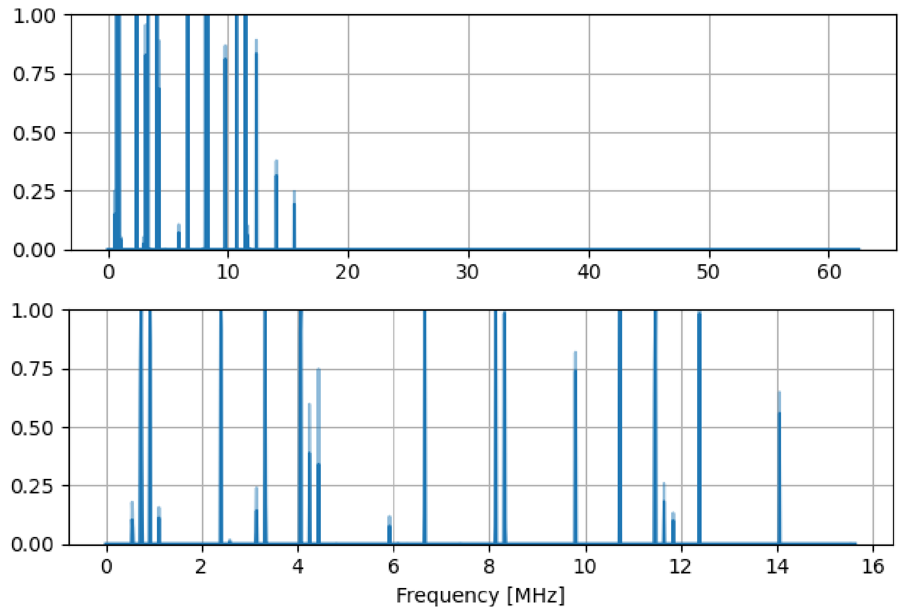
Wavelet transform as a **filter** to keep **lower frequencies** of signal



Success rate of clustering (at each **timing**)

⇔

⇔



Success rate of clustering (at each **frequency**)

KUDELSKI IOTHINGS

Section 3

Countermeasure

KUDELSKI IOTHINGS

Secure `swap_points`

Suppose a, b are words that need to be swapped depending on `swap`

Before

1. $\texttt{mask} = \begin{cases} \texttt{0x00000000} \text{ if } \texttt{swap} = 0 \\ \texttt{0xFFFFFFFF} \text{ if } \texttt{swap} = 1 \end{cases}$
2. $\texttt{temp} = \texttt{mask} \,\&\, (a \oplus b)$
3. $a = \texttt{temp} \oplus a$
4. $b = \texttt{temp} \oplus b$

After

$\texttt{m1} \oplus \texttt{m2} = \begin{cases} \texttt{0x00000000} \text{ if } \texttt{swap} = 0 \\ \texttt{0xFFFFFFFF} \text{ if } \texttt{swap} = 1 \end{cases}$

1. Draw `m1`, `m2` uniformly s.t. $\texttt{m2} = \begin{cases} \texttt{m1} \quad \text{if } \texttt{swap} = 0 \\ \neg\texttt{m1} \text{ if } \texttt{swap} = 1 \end{cases}$
2. $\texttt{temp1} = \texttt{m1} \,\&\, (a \oplus b)$
3. $\texttt{temp2} = \texttt{m2} \,\&\, (a \oplus b)$
4. $a = (\texttt{temp1} \oplus a) \oplus \texttt{temp2}$
5. $b = (\texttt{temp1} \oplus b) \oplus \texttt{temp2}$

Split mask into two random shares

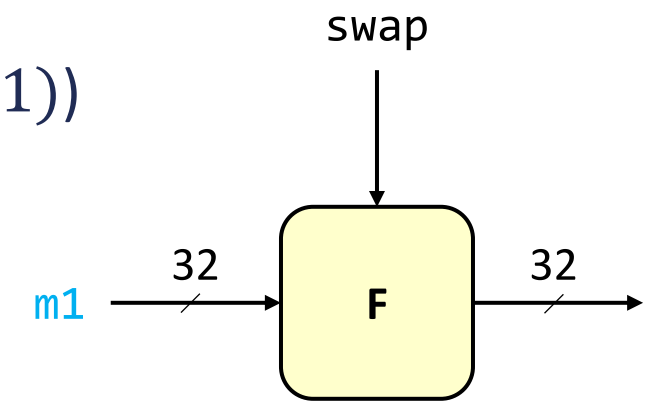
Secure mask computation

How do you obtain such $\texttt{m1}$ and $\texttt{m2}$?

$\Rightarrow$ **Two's complement** (i.e., $\neg x = -(x+1)$)

Instructions

1. $\texttt{u1} = \texttt{randombytes}(4)\ \&\ \texttt{0xFFFFFFFD}$
2. $\texttt{m1} = \texttt{randombytes}(4)\ \&\ \texttt{0xFFFFFFFE}$
3. $\texttt{u2} = \texttt{u1} + \texttt{swap}$
4. $\texttt{r} = \texttt{m1} + \texttt{swap}$
5. $\texttt{u1} = \texttt{u1} + 1$
6. $\texttt{u1} = \texttt{u1} \times \texttt{r}$
7. $\texttt{u2} = \texttt{u2} + \texttt{swap}$
8. $\texttt{u2} = \texttt{u2} \times \texttt{r}$
9. $\texttt{m2} = \texttt{u1} - \texttt{u2}$



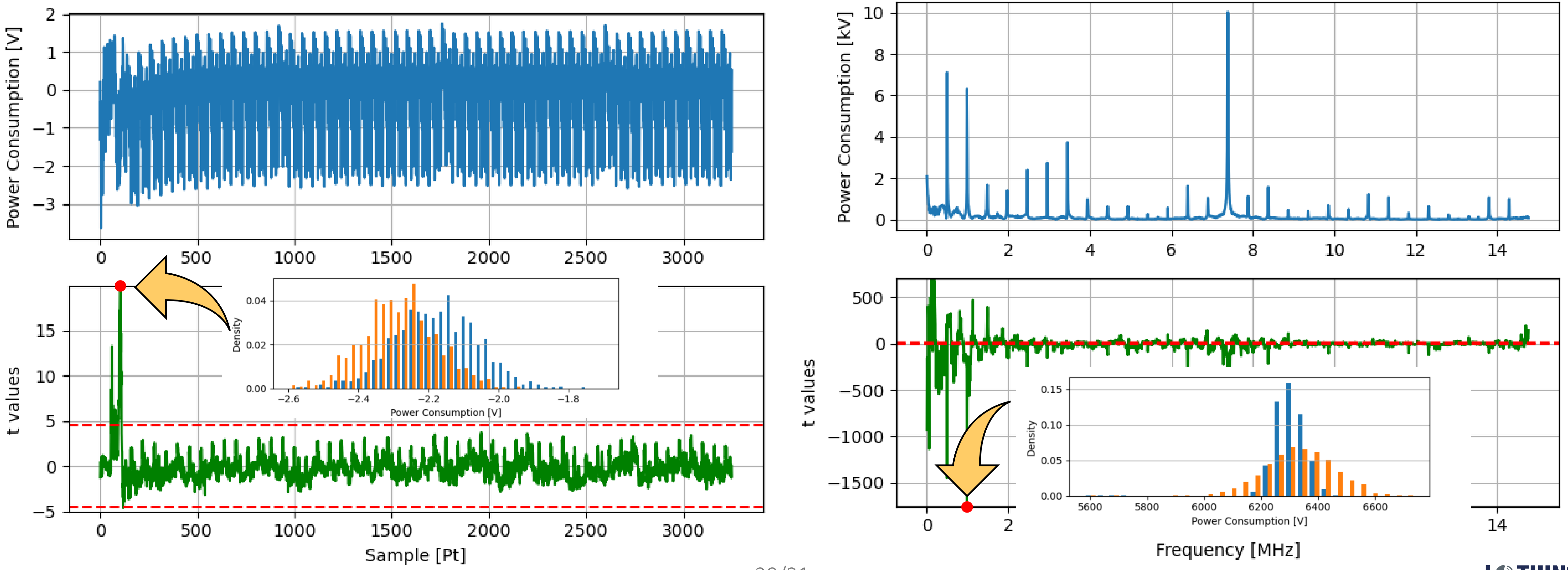
$$\texttt{m2} = \begin{cases} \texttt{m1} & \text{if } \texttt{swap} = 0 \\ \neg\texttt{m1} & \text{if } \texttt{swap} = 1 \end{cases}$$

$$\begin{aligned} \texttt{m2} &= (\texttt{m1} + \texttt{swap})(1 - 2\cdot\texttt{swap}) \\ &= \texttt{u1}(\texttt{m1} + \texttt{swap}) - \texttt{u2}(\texttt{m1} + \texttt{swap}) \end{aligned}$$

where $\texttt{u1} - \texttt{u2} = 1 - 2\cdot\texttt{swap}$

t-test (swap=0 vs. swap=1, $N = 1000$)

ChipWhisperer (29.54 [MHz]) + LNA (20dB) on STM32F3 (7.37 [MHz])



www.kudelski-iot.com

lasec.epfl.ch

Aymeric Genêt, aymeric.genet@nagra.com

Aymeric Genêt, aymeric.genet@epfl.ch