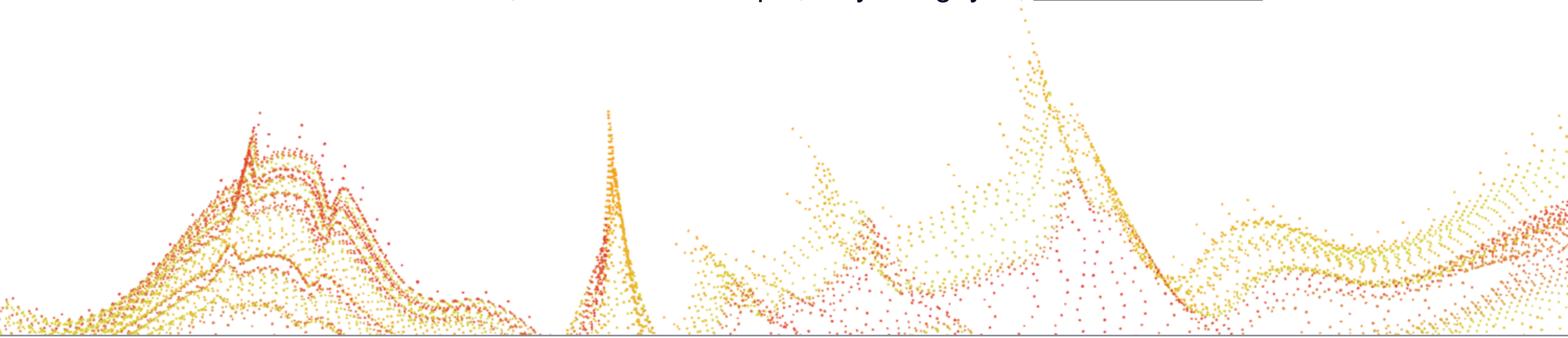
# Avengers assemble!
# Supervised learning meets lattice reduction

*TCHES 2025, 17/09/2025, Kuala Lumpur, Malaysia*

Damien Marion, Pierre-Alain Fouque, Quyen Nguyen, Alexandre Wallet



# A single trace attack against Kyber's KeyGen

Sum-up in 4 items:

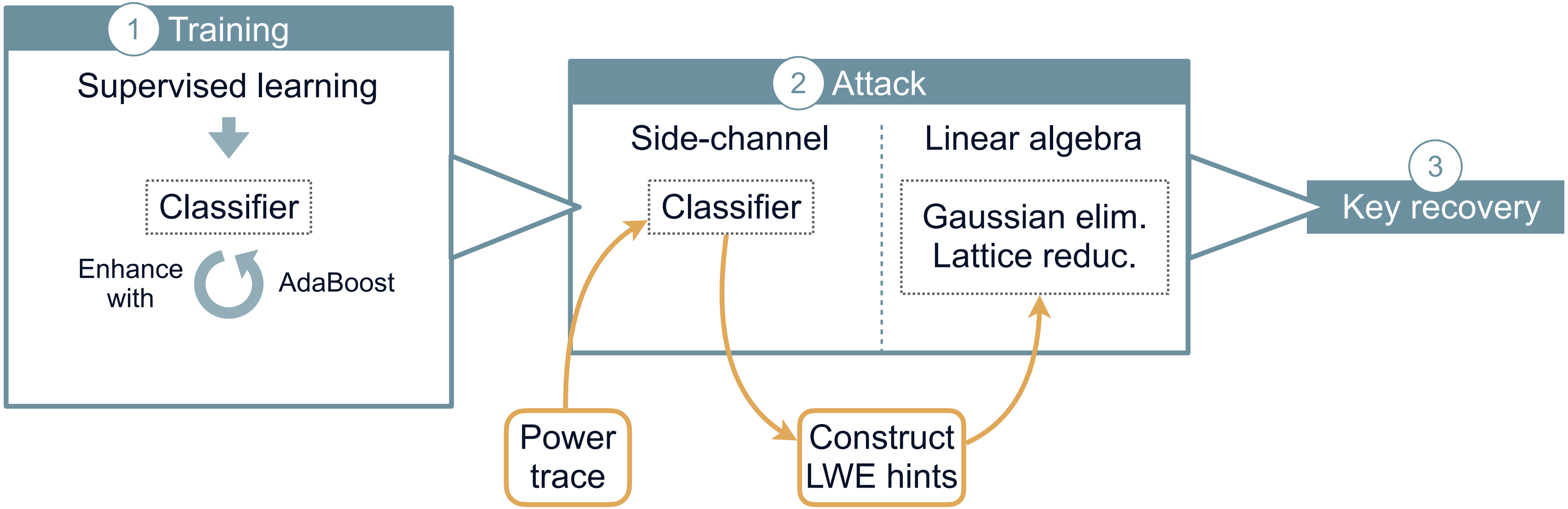
1. Target: an unprotected CBD sampler in the KeyGen.
   Method: power analysis.

2. SCA model: classifier on Hamming weight.
   Linear algebra tools: Gaussian elimination or (black-box) lattice reduction, with « LWE hints ».

3. Principle: **Classifier + Linear algebra = secret keys.**

4. **Results: full key recovery at all security levels, with average success rate > 96%**

# Flow of the attack = roadmap of this talk




(LWE: Learning With Errors)

# Kyber's KeyGen, secret keys and CBD

**Algorithm 1:** CRYSTALS-Kyber key generation algorithm

**Input:** Secret key $sk \in \mathcal{B}^{12 \cdot k \cdot n/8}$
**Result:** Public key $pk \in \mathcal{B}^{12 \cdot k \cdot n/8+32}$

$d \leftarrow \mathcal{B}^{32}$
$(\rho, \sigma) = G(d)$ // G
$\{\hat{\mathbf{A}}_{j,i} = \text{Parse}(\text{XOF}(\rho, i, j))\}_{i<k-1, j<k-1}$ // $\hat{\mathbf{A}} \in R_q^{k\times}$
$\{\mathbf{s}_i = \text{CBD}_{\eta_1}(\text{PRF}(\sigma, i))\}_{i<k}$
$\{\mathbf{e}_i = \text{CBD}_{\eta_1}(\text{PRF}(\sigma, i+k))\}_{i<k}$
$\hat{\mathbf{e}}, \hat{\mathbf{s}} = \text{NTT}(\mathbf{e}), \text{NTT}(\mathbf{s})$
$\hat{\mathbf{t}} = \hat{\mathbf{A}} \circ \hat{\mathbf{s}} + \hat{\mathbf{e}}$
$pk = \text{Encode}_{12}(\hat{\mathbf{t}} \bmod^+ q) || \rho$
$sk = \text{Encode}_{12}(\hat{\mathbf{s}} \bmod^+ q)$
**return** $pk, sk$

**Algorithm 2:** CRYSTALS-Kyber CBD function from [8].

**Input:** Byte array $B = (b_0, b_1, \ldots b_{64\eta-1}) \in \mathcal{B}^{64\eta}$
**Result:** Polynomial $f \in R_q$

$(\beta_0, \ldots, \beta_{512\eta-1}) = \text{BytesToBits}(B)$
**for** ( $i = 0;\ i < 256;\ i{+}{+}$ ) {
    $a = \sum_{j=0}^{\eta-1} \beta_{2i\eta+j}$
    $b = \sum_{j=0}^{\eta-1} \beta_{2i\eta+\eta+j}$
    $\boxed{f_i = a - b}$
}
**return** $\sum_0^{255}(f_i X^i)$

- $k \in \{2,3,4\}$. All computations modulo $3329$.
- $\texttt{sk} = (\mathbf{s}, \mathbf{e})$ : two vectors of $256k$ **small** coefficients.
- $\texttt{pk} = (\mathbf{A}, \mathbf{t})$ with:

  $\mathbf{A}$ : $256k \times 256k$ public matrix, large coefficients.
  $\mathbf{t} = \mathbf{A}\mathbf{s} + \mathbf{e}$, $256k$ large coefficients.

$\texttt{sk}$ is **sampled out of CBD**$_\eta$ $(\eta_{512} = 3, \eta_{768,1024} = 2)$

CBD: <u>C</u>entered <u>B</u>inomial <u>D</u>istribution, $|\texttt{sk}_i| \leq \eta$

Main focus

# Settings for the training phase

## Data collection and sorting

- 4 different chips
- 20.000 traces/chips/implementation
- Isolation of 256 subtraces by traces

Total: 8 datasets, for a total of > 5M subtraces.

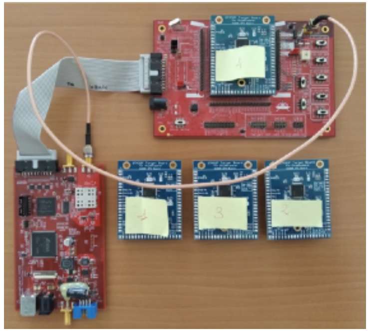
Training/Testing sets: 80/20 splits

- 16 classifiers trained (on Hamming weight)

See also our artifacts:

https://gitlab.inria.fr/capsule/avengers-assemble

Acquisition on a ChipWhisperer CW1200



```c
static void cbd2(poly *r,
                 const unsigned char *buf){
unsigned int i, j;
uint32_t t, d;
int16_t a, b;

for (i = 0; i < n/8; i++) {
  t  = load32_littleendian(buf + 4 * i);
  d  = t & 0x55555555;
  d += (t >> 1) & 0x55555555;

  for (j = 0; j < 8; j++) {
    // in {0, 1, 2}
    a = (d >> (4 * j + 0)) & 0x3;
    // in {0, 1, 2}
    b = (d >> (4 * j + 2)) & 0x3;
    // in {-2, -1, 0, 1, 2}
    r->coeffs[8 * i + j] = a - b;}}
}
```

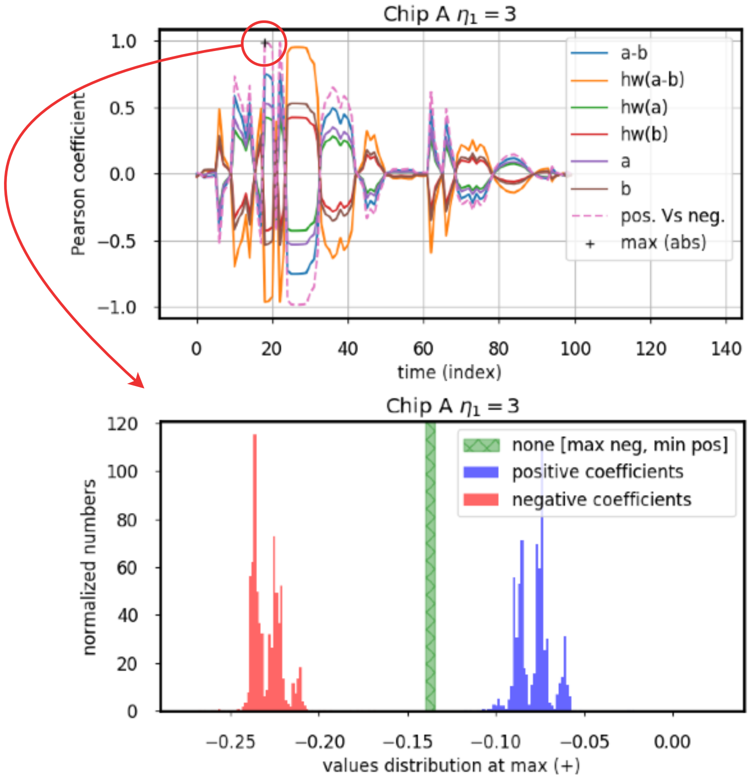
```c
void cbd3(poly *r, int add,
          const unsigned char *buf) {
unsigned int i,j;
uint32_t t,d;
int16_t a,b;

for(i = 0; i < n/4; i++) {
  t  = load24_littleendian(buf + 3 * i);
  d  = t & 0x00249249;
  d += (t >> 1) & 0x00249249;
  d += (t >> 2) & 0x00249249;

  for(j=0; j<4; j++) {
    // in {0, 1, 2, 3}
    a = (d >> (6 * j + 0)) & 0x7;
    // in {0, 1, 2, 3}
    b = (d >> (6 * j + 3)) & 0x7;
    // in {-3, -2, -1, 0, 1, 2, 3}
    r->coeffs[4 * i + j] = a - b;}}
}
```

Code from the pqm4 open source implementation

# Correlations, positive/negative separation




Pearson correlation coefficients of the leakage.

```
// in {0, 1, 2, 3}
a = (d >> (6 * j + 0)) & 0x7;
// in {0, 1, 2, 3}
b = (d >> (6 * j + 3)) & 0x7;
// in {-3, -2, -1, 0, 1, 2, 3}
r->coeffs[4 * i + j] = a - b;}}
```

Focus: $\mathtt{HW}(a-b)$ and « positive vs. negative »

« positive vs. negative » is perfectly distinguishable

(Similar plots for $\eta = 2$)

# A method to improve trust

**Observed limitations:**

Templates: high accuracy ( $> 90\%$) — Results not very trustable

→ **Cannot tolerate mistake** as it can prevent the key recovery

→ **Cannot sample new traces** in our setting

**Our mitigation:**

On testing sets, using a trained classifier:

1. Labels → Proba. of classes

$$\ell = (x, \text{pred}(x), \text{true}(x)) \longrightarrow (p_1, \ldots, p_{\#classes})$$

$\downarrow$

$q_c$ : highest probability that class $c$ is wrong

2. $$\text{assigned-value}(\ell) = \begin{cases} \text{pred}(x) \text{ if } p_{pred(x)} > q_{pred(x)} \\ \perp \text{ else.} \end{cases}$$

→ **Assumption:** *assigned-value gives the* **true** $\mathtt{HW}(a-b)$ (or nothing)

# From Hamming weights to values

**Assumption:** *for each classifier, assigned-value gives the* **true** $\mathtt{HW}(a - b)$ (or nothing)

Mapping to values: example for $\eta = 2$.

| | $a - b \geq 0$ | | $a - b < 0$ | |
|---|---|---|---|---|
| $\mathtt{HW}(a - b)$ | 0 | 1 | 15 | 14 |
| $a - b$ | **0** | 1,2 | **-1** | **-2** |

Can't know
= can't use

**Conclusion:** our classifiers give us **a proportion** of $\mathtt{sk}$'s coefficients (so, « LWE hints »).

PQ SHIELD

# Comparisons of the classifiers

Three methods: Templates, Decision Trees (DT), and DT+AdaBoost

All have high accuracy. Below we display their trustability (see $\eta = 2$ in the paper)

### Templates

$\eta_1 = 3$

| | Chip A | | Chip B | | Chip C | | Chip D | |
|---|---|---|---|---|---|---|---|---|
| Chip A | 9.20 | 6.12 | 10.79 | 7.86 | 9.04 | 2.49 | 10.40 | 5.50 |
| | 29.78 | 54.90 | 29.72 | 51.64 | 33.75 | 54.73 | 30.96 | 53.14 |
| Chip B | 9.72 | 6.64 | 12.97 | 10.19 | 9.56 | 2.64 | 9.98 | 5.69 |
| | 30.27 | 53.37 | 27.70 | 49.14 | 33.61 | 54.20 | 30.83 | 53.50 |
| Chip C | 3.83 | 1.83 | 1.76 | 0.86 | 9.68 | 6.81 | 9.56 | 0.54 |
| | 34.25 | 60.08 | 34.34 | 63.04 | 30.22 | 53.29 | 34.25 | 55.66 |
| Chip D | 8.84 | 5.14 | 4.22 | 6.10 | 5.55 | 4.95 | 11.84 | 9.15 |
| | 28.98 | 57.04 | 32.61 | 57.07 | 32.69 | 56.81 | 29.04 | 49.98 |

### DT

$\eta_1 = 3$

| | Chip A | | Chip B | | Chip C | | Chip D | |
|---|---|---|---|---|---|---|---|---|
| Chip A | 0.00 | 0.00 | 0.00 | 0.00 | 6.93 | 0.00 | 0.00 | 0.00 |
| | 34.35 | 65.65 | 34.35 | 65.65 | 27.46 | 65.60 | 34.38 | 65.62 |
| Chip B | 0.00 | 0.00 | 0.00 | 0.00 | 6.93 | 0.00 | 0.00 | 0.00 |
| | 34.35 | 65.65 | 34.35 | 65.65 | 27.46 | 65.60 | 34.38 | 65.62 |
| Chip C | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| | 34.35 | 65.65 | 34.35 | 65.65 | 34.40 | 65.60 | 34.38 | 65.62 |
| Chip D | 0.00 | 0.00 | 5.89 | 0.00 | 0.01 | 0.01 | 0.00 | 0.00 |
| | 34.35 | 65.65 | 28.46 | 65.65 | 34.39 | 65.59 | 34.38 | 65.62 |

### DT+AdaBoost

$\eta_1 = 3$

| | Chip A | | Chip B | | Chip C | | Chip D | |
|---|---|---|---|---|---|---|---|---|
| Chip A | 56.08 | 43.47 | 55.82 | 43.09 | 49.79 | 35.59 | 55.77 | 29.04 |
| | 0.10 | 0.34 | 0.42 | 0.67 | 12.95 | 1.66 | 0.41 | 14.77 |
| Chip B | 55.63 | 42.76 | 55.99 | 43.24 | 39.14 | 32.11 | 52.58 | 32.76 |
| | 0.38 | 1.24 | 0.06 | 0.70 | 25.04 | 3.71 | 4.00 | 10.67 |
| Chip C | 45.63 | 23.89 | 37.32 | 31.98 | 56.19 | 43.02 | 51.21 | 21.67 |
| | 18.02 | 12.46 | 27.12 | 3.58 | 0.02 | 0.77 | 6.32 | 20.80 |
| Chip D | 55.23 | 42.47 | 54.78 | 36.48 | 46.90 | 39.81 | 55.68 | 43.60 |
| | 0.97 | 1.33 | 6.55 | 2.18 | 11.13 | 2.15 | 0.42 | 0.31 |

Vertical: trained chip; horizontal: tested chip. Percentages are number of remaining possibilities, sorted as

| | |
|---|---|
| 1 | 2 |
| 3 | 4 |

Upper left: proportion of recovered coefficients.

PQSHIELD

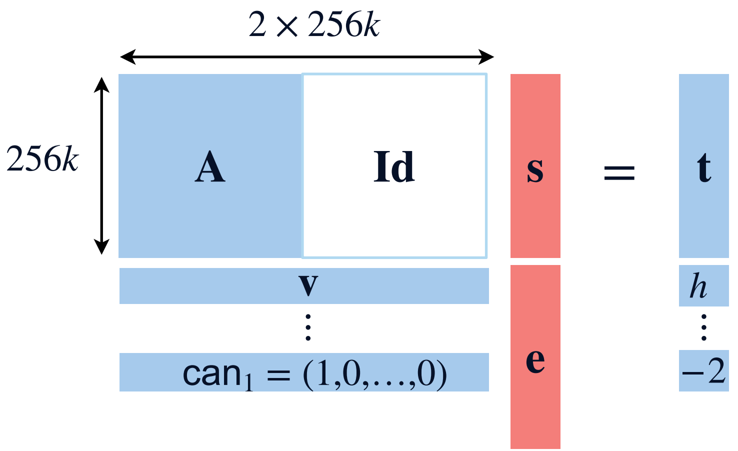
# Learning With Errors, hints, linear algebra

Hints = learned linear combination

- « perfect[1] »: $\langle \mathbf{v}, \mathtt{sk} \rangle = h$

From previous slide: we have perfect hints:

$\langle \text{can}_i, \mathtt{sk} \rangle = \text{assigned-value}(\ell)$




**Quick take 1: dimension of the problem = $256k$**

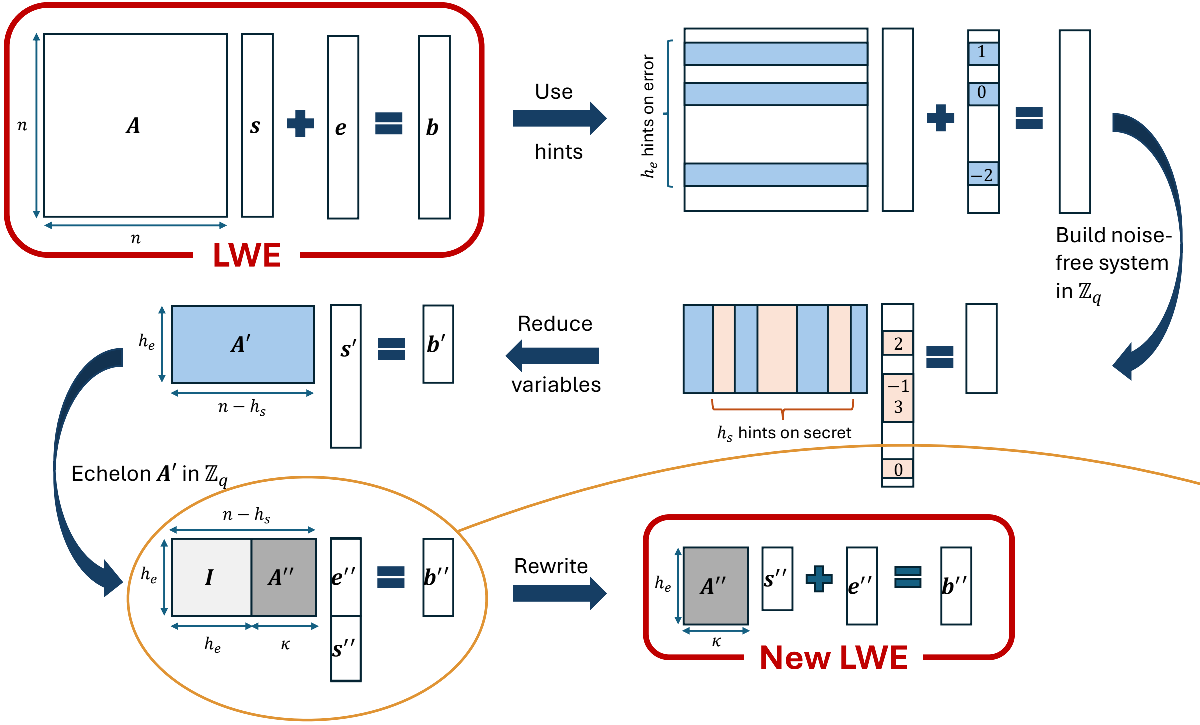
**Quick take 2: learning $\geq 50\,\%$ of $\mathtt{sk}$ = total break**

Often the situation in our attack

1: there are other types of « hints », not appearing in this work. See also this afternoon's talk on Hertzbleed and modular hints

PQSHIELD

# Depiction of hint processing




$h_\mathbf{s}$: #hints on $\mathbf{s}$

$h_\mathbf{e}$: #hints on $\mathbf{e}$

$n = 256k$

$\kappa = n - (h_\mathbf{s} + h_\mathbf{e})$

At the end:

$h_\mathbf{e}$ versus $n - h_\mathbf{s}$

# Sum-up, identification of two regimes

$h_\mathbf{e}$ versus $n - h_\mathbf{s}$

**« Way too many hints »**

$h_\mathbf{e} + h_\mathbf{s} \geq n$ : the linear system is (over)determined (quick take 2).

Solve it using Gaussian elimination in $\mathbb{Z}/3329\mathbb{Z}$ (takes $< 1$s)

Quick take 3: happens almost always for Kyber-$\{768, 1024\}$

**Normal cryptanalysis = « primal attack »**

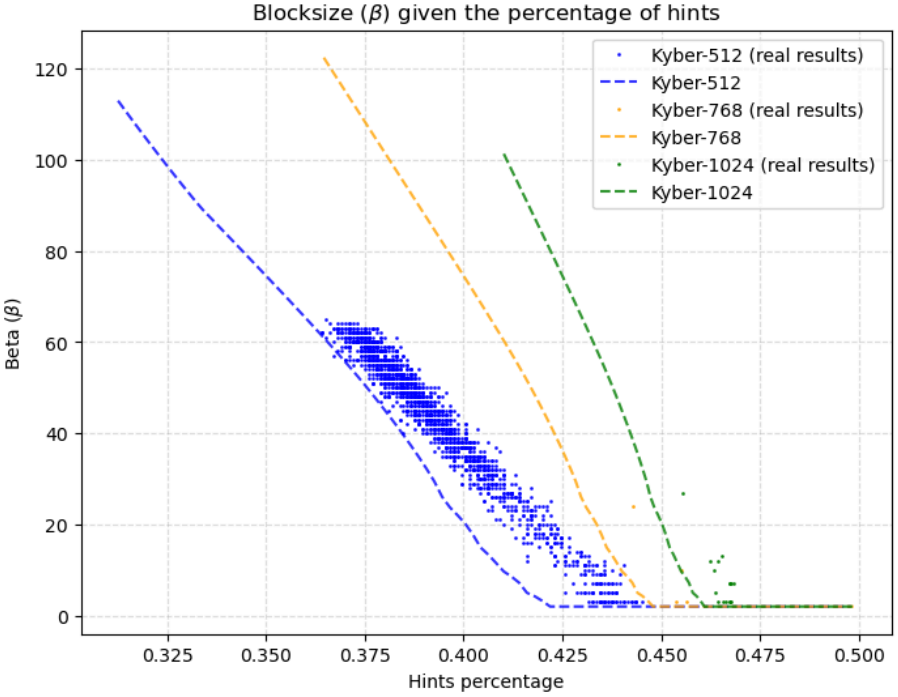
$\texttt{sk}$ is unusually short in a lattice of rank $n - h_\mathbf{s}$.
Use (black-box) lattice reduction, parameters depends on the proportion:

$$\rho := \frac{h_\mathbf{s} + h_\mathbf{e}}{2n}$$

Quick take 4: with $\rho \geq 35\ \%$ , a laptop recovers the full key on a weekend[1].

1 : trendy cryptanalysis unit

# Prediction vs. Experimental block-size



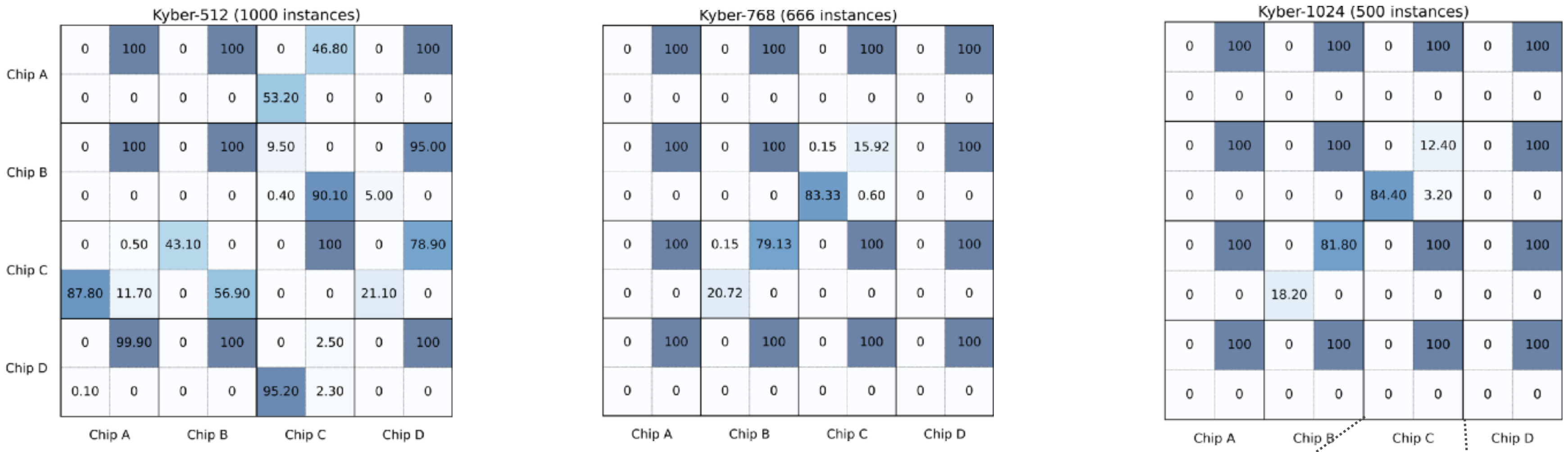

Tool: BKZ (block-lattice reduction) from sagemath
Complexity is a function of $2^\beta$ ($\beta$: block-size)

$\beta$ predicted w/ standard cost model
→ this gives a starting value;
→ increase the value until:

- $\mathtt{sk}$ is found;
- or we reach a threshold $(65)$

Kyber-$512$: prediction is a bit too optimistic
$\{768, 1024\}$: same, but we do not need so much lattice reduction.

# Larger scale experimental results




Key recovery percentages for the three security levels, depending on the method to complete.
Top left = percentage of unrecovered key.

# Comparison to the talk of Tuesday morning

*« Adaptative template attack against the Kyber binomial sampler », E.C.Y. Peng, M.G.Kuhn*

| | This work | Talk of Tuesday |
|---|---|---|
| Target | CBD in KeyGen$^*$ | Any CBD (KeyGen, Encaps, Decaps) |
| Classifier | $\mathtt{HW}(a-b)$, « pos vs. neg » | $\mathtt{HW}(a), \mathtt{HW}(b)$, « Buf » |
| Accuracy | ++ | +++ |
| Necessary $\rho$ | $\geq$35% for reasonable attack | 100% (or almost$^{**}$) |
| Success rate | High (close to 100%) | Moderate to high |
| Security level | Any | Kyber-768 |
| Noise tolerance | Medium | Low |

Natural approach:
Combine both to get best of both worlds.

$^*$: our models could be trained identically on Encaps/Decaps. $**$ : this could be reduced by combining with lattice reduction as in our work

# Differences with the talk of this afternoon

*« Improved Attacks Against Lattice-Based KEMs Using Hints From Hertzbleed », Z. Li et al.*

Quick take 5: two different attack styles, targeting distinct leakages, providing different hints, exploited in different lattices.

Common point: lattice reduction to complete the key recovery.

## About hints:

Hints = learned linear combination
- « perfect »: $\langle \mathbf{v}, \mathtt{sk} \rangle = h$
- « modular » : $\langle \mathbf{v}, \mathtt{sk} \rangle = h \bmod a$

Hertzbleed provides **modular** hints.

Li et al. use them in lattices of dimension 256, related to the NTT. See their talk for more infos!

# Summary of results, conclusion, thank you!

| Kyber | $\eta$ | Worst $\rho$ | Largest $\beta$ | Worst time | Smallest $\beta$ | Best time |
|---|---|---|---|---|---|---|
| 512 | 3 | $\approx 37\ \%$ | 65 | < 18h | 0 | < 1s |
| 768 | 2 | $\approx 43\ \%$ | 23 | < 18h | 0 | < 1s |
| 1024 | 2 | $\approx 46\ \%$ | 25 | < 18h | 0 | < 1s |

Key recovery, worst and best cases, three security level.

| Kyber | Worst | Average | Best |
|---|---|---|---|
| 512 | 56.9% | 96.71% | 100 % |
| 768 | 99.85% | 99.98% | 100 % |
| 1024 | 100 % | 100 % | 100 % |

Success rates, depending on security level

- <u>What:</u> a single trace attack against Kyber achieving full key recovery
- <u>How:</u> PA on the CBD sampler in the KeyGen + enhanced supervised learning + lattice reduction
- <u>Concrete results:</u> avg. success rates > 96% (over thousands of experiments).
- <u>Additional:</u> enhancement of trust for classifiers, stability wrt. multi-chip training (in paper).

**<u>Recommendation:</u>** use masking, shuffling and usual countermeasures even for the KeyGen.