“Lattice PQC Candidates: A Side-Channel and Fault Analysis on Dilithium”

Andersson Calle Viera

Previous Results, 13 june 2022

Outline

Outline

A few details about the current situation

- **Current cryptography**:
  Integer factorization ex: **RSA**
  Discrete Logarithm problem on finite fields ex: **DSA, DH**
  Discrete Logarithm problem on eliptic curves ex: **ECDSA**

- **Rise of Quantum computing**:
  Shor's Algorithm breaks current systems in polynomial time
  First quantum computer 10 to 15 years

- **NIST international PQC competition**:
  Currently in the last round with Codes, Multivariate and Lattices
  3 out of 4 PKE/ KEM schemes and 2 out of 3 Signature schemes are lattice based

- **Embedded Constraints**:
  Reduced Memory size (RAM and flash) and Limited processor clock frequency
  Slow Communication rates: < 100 kB/s (for contactless, time: < 300 ms)

PQC Requirements

NIST Security Level	Equivalent type of security
I	Key search on a block cipher with 128-bit key (**AES-128**)
II	Collision search on a 256-bit hash function (**SHA256**/ **SHA3-256**)
III	Key search on a block cipher with 192-bit key (**AES-192**)
IV	Collision search on a 384-bit hash function **SHA384**/ **SHA3-384**)
V	Key search on a block cipher with 256-bit key (**AES-256**)

Table: NIST Levels of security

- **Side Channel Attacks**: Instead of directly attacking a cryptosystem one can use different techniques to infer data of an implementation of such an algorithm
- On embedded devices the user can be the attacker !
- **From the original NIST PQC call for proposals in 2016**:
  "Schemes that can be made resistant to side-channel attacks at minimal cost are more desirable than those whose performance is severely hampered by any attempt to resist side-channel attacks."

Internship goal

- State of the Art selection of relevant Attacks/ Countermeasures papers
- Side Channel/ Fault Attack Analysis of CRYSTALS package
- Selecting countermeasures with as little overhead as possible
- Develop High level and Embedded implementation of these countermeasures
- Perform Tests on a protected code

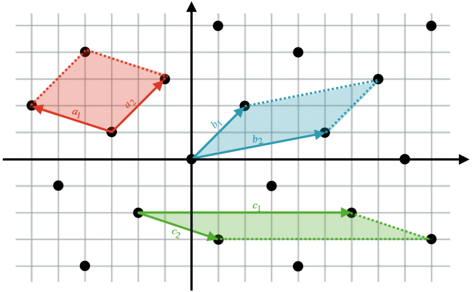
Outline

THALES

Lattices

Let $(b_1, \ldots, b_d) \in \mathbb{R}^n$ be a set of vectors:

$$\mathcal{L}(b_1, \ldots, b_d) = \left\{ \sum_{i=1}^{d} \mu_i b_i \ : (\mu_1, \ldots, \mu_d) \in \mathbb{Z} \right\}$$



Degree 2 lattice generated by:
$b_1 = (2,\ 2)$ and $b_2 = (5,\ 1)$

Shortest Vector Problem (SVP):

Given $(b_1, \ldots, b_n)$ a basis of $\mathcal{L} \in \mathbb{R}^n$.
Decision: $\forall r > 0$, decide if there is $x \neq 0 \in \mathcal{L}$ such that $||x|| \leq r$.
Search: Find such a vector x.

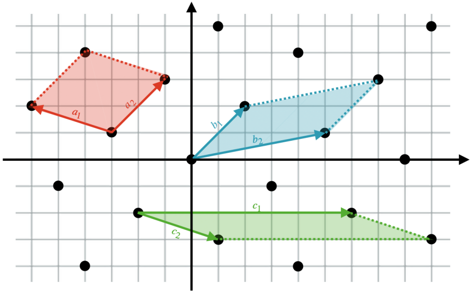
Closest Vector Problem (CVP):

Given $(b_1, \ldots, b_n)$ a basis of $\mathcal{L} \in \mathbb{R}^n$ and $y \in \mathbb{R}^n$.
Decision: $\forall r > 0$, decide if there is $x \neq 0 \in \mathcal{L} \mid ||y - x|| \leq r$.
Search: Find such a vector x.

Lattices

Let $(b_1, \ldots, b_d) \in \mathbb{R}^n$ be a set of vectors:

$$\mathcal{L}(b_1, \ldots, b_d) = \left\{ \sum_{i=1}^{d} \mu_i b_i \;:\; (\mu_1, \ldots, \mu_d) \in \mathbb{Z} \right\}$$



Degree 2 lattice generated by:
$b_1 = (2,\ 2)$ and $b_2 = (5,\ 1)$

Shortest Vector Problem (SVP):

Given $(b_1, \ldots, b_n)$ a basis of $\mathcal{L} \in \mathbb{R}^n$.
Decision: $\forall r > 0$, decide if there is $x \neq 0 \in \mathcal{L}$ such that $||x|| \leq r$.
Search: Find such a vector x.

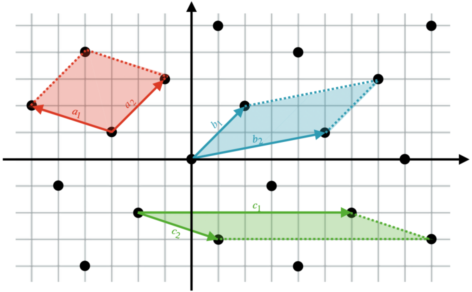
Closest Vector Problem (CVP):

Given $(b_1, \ldots, b_n)$ a basis of $\mathcal{L} \in \mathbb{R}^n$ and $y \in \mathbb{R}^n$.
Decision: $\forall r > 0$, decide if there is $x \neq 0 \in \mathcal{L} \mid ||y - x|| \leq r$.
Search: Find such a vector x.

Lattices

Let $(b_1, \dots, b_d) \in \mathbb{R}^n$ be a set of vectors:

$$\mathcal{L}(b_1, \dots, b_d) = \left\{ \sum_{i=1}^{d} \mu_i b_i \ : (\mu_1, \dots, \mu_d) \in \mathbb{Z} \right\}$$



Degree 2 lattice generated by:
$b_1 = (2,\ 2)$ and $b_2 = (5,\ 1)$

Shortest Vector Problem (SVP):

Given $(b_1, \dots, b_n)$ a basis of $\mathcal{L} \in \mathbb{R}^n$.
Decision: $\forall r > 0$, decide if there is $x \neq 0 \in \mathcal{L}$ such that $||x|| \leq r$.
Search: Find such a vector x.

Closest Vector Problem (CVP):

Given $(b_1, \dots, b_n)$ a basis of $\mathcal{L} \in \mathbb{R}^n$ and $y \in \mathbb{R}^n$.
Decision: $\forall r > 0$, decide if there is $x \neq 0 \in \mathcal{L} \mid ||y - x|| \leq r$.
Search: Find such a vector x.

Lattice Based Cryptography

Learning With Error (LWE):

Let $m, n, q > 0$, $\chi \leftarrow \mathbb{Z}$ and $A \underset{\$}{\leftarrow} \mathbb{Z}_q^{m \times n}$.

Let $s_1 \underset{\$}{\leftarrow} \mathbb{Z}_q^n$, $s_2 \underset{\$}{\leftarrow} \chi^m$ such that:

$t := As_1 + s_2$ mod q

Decision: Distinguish (A,t) from (A,u)

Search: Recover s_1 with small s_2

Short Integer Solution (SIS):

Let m, n and q be positive integers, $\gamma > 0$ be an integer and$A \underset{\$}{\longleftarrow} \mathbb{Z}_q^{m \times n}$.

Search: Find $s | As = 0$ mod q and $||s|| \leq \gamma$

Can be instantiate on different mathematical objects, Rings, Modules ...

$$A = \begin{pmatrix} \begin{matrix} a_0 & -a_3 & -a_2 & -a_1 \\ a_1 & a_0 & -a_3 & -a_2 \\ a_2 & a_1 & a_0 & -a_3 \\ a_3 & a_2 & a_1 & a_0 \end{matrix} & \begin{matrix} a_8 & -a_{11} & -a_{10} & -a_9 \\ a_9 & a_8 & -a_{11} & -a_{10} \\ a_{10} & a_9 & a_8 & -a_{11} \\ a_{11} & a_{10} & a_9 & a_8 \end{matrix} \\ \begin{matrix} a_4 & -a_7 & -a_6 & -a_5 \\ a_5 & a_4 & -a_7 & -a_6 \\ a_6 & a_5 & a_4 & -a_7 \\ a_7 & a_6 & a_5 & a_4 \end{matrix} & \begin{matrix} a_{12} & -a_{15} & -a_{14} & -a_{13} \\ a_{13} & a_{12} & -a_{15} & -a_{14} \\ a_{14} & a_{13} & a_{12} & -a_{15} \\ a_{15} & a_{14} & a_{13} & a_{12} \end{matrix} \end{pmatrix}$$

$$= \begin{pmatrix} a_{0,0}(x) & a_{0,1}(x) \\ a_{1,0}(x) & a_{1,1}(x) \end{pmatrix}$$

Lattice Based Cryptography

Learning With Error (LWE):

Let $m, n, q > 0$, $\chi \leftarrow \mathbb{Z}$ and $A \underset{\$}{\leftarrow} \mathbb{Z}_q^{m\times n}$.

Let $s_1 \underset{\$}{\leftarrow} \mathbb{Z}_q^n$, $s_2 \underset{\$}{\leftarrow} \chi^m$ such that:

$t := As_1 + s_2$ mod q

Decision: Distinguish (A,t) from (A,u)

Search: Recover s_1 with small s_2

Short Integer Solution (SIS):

Let m, n and q be positive integers, $\gamma > 0$ be an integer and$A \underset{\$}{\longleftarrow} \mathbb{Z}_q^{m\times n}$.

Search: Find $s|As = 0$ mod q and $||s|| \leq \gamma$

Can be instantiate on different mathematical objects, Rings, Modules ...

$$A = \begin{pmatrix} \boxed{\begin{matrix} a_0 & -a_3 & -a_2 & -a_1 \\ a_1 & a_0 & -a_3 & -a_2 \\ a_2 & a_1 & a_0 & -a_3 \\ a_3 & a_2 & a_1 & a_0 \end{matrix}} & \boxed{\begin{matrix} a_8 & -a_{11} & -a_{10} & -a_9 \\ a_9 & a_8 & -a_{11} & -a_{10} \\ a_{10} & a_9 & a_8 & -a_{11} \\ a_{11} & a_{10} & a_9 & a_8 \end{matrix}} \\ \boxed{\begin{matrix} a_4 & -a_7 & -a_6 & -a_5 \\ a_5 & a_4 & -a_7 & -a_6 \\ a_6 & a_5 & a_4 & -a_7 \\ a_7 & a_6 & a_5 & a_4 \end{matrix}} & \boxed{\begin{matrix} a_{12} & -a_{15} & -a_{14} & -a_{13} \\ a_{13} & a_{12} & -a_{15} & -a_{14} \\ a_{14} & a_{13} & a_{12} & -a_{15} \\ a_{15} & a_{14} & a_{13} & a_{12} \end{matrix}} \end{pmatrix}$$

$$= \begin{pmatrix} a_{0,0}(x) & a_{0,1}(x) \\ a_{1,0}(x) & a_{1,1}(x) \end{pmatrix}$$

CRYSTALS Package

- 2 schemes on the final Round
- Based on Module Lattices
- Quotient Ring $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^n + 1)$ with $n = 2^8 = 256$
  - $X^{256} + 1$ cyclotomic polynomial
  - Efficient multiplication using NTT: $\mathcal{O}(n)$ (point-wise)

NTT in practice: Butterfly operation

- Atomic operation in the loop is called ***Butterfly operation***

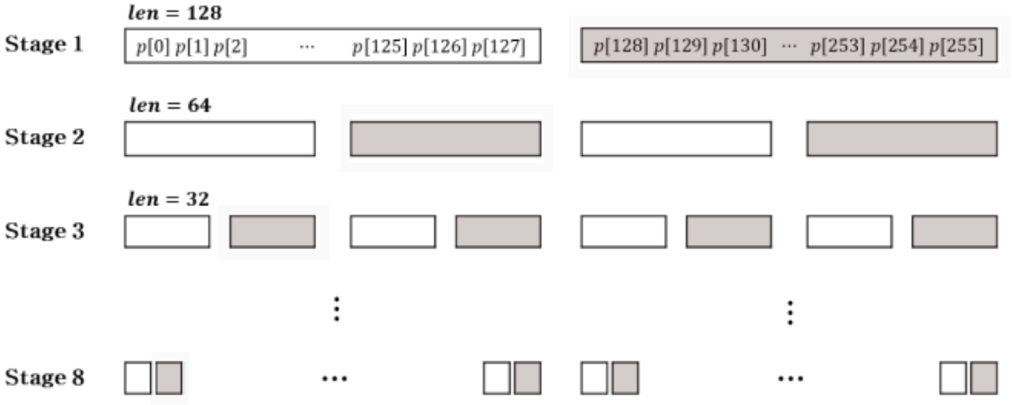
```
for(len = 128; len > 0; len >>1) {
    for(start = 0; start < N; start = j + len){
        w = zetas[k++];
        for(j = start; j < start + len; ++j){
            t = Montgomeryreduce(w * p[j + len]);
            p[j + len] = p[j] + 2*Q - t;
            p[j] = p[j] + t;
        }
    }
}
```

$$c = a + bw \quad c = a + b$$
$$d = a - bw \quad d = (a - b)w$$

- $n-1$ degree polynomial
- $log(n)$ stages
- $n/2$ butterflies
- $\mathcal{O}(nlog(n))$ complexity

- There are 2 types of Butterfy:
- Cooley-Tukey(CT) for the NTT and Gentleman-Sande (GS) for the INNT $\mathcal{O}(nlog(n))$ complexity

NTT in practice: Butterfly operation



Dilithium

- Signature scheme
- Simple to securely implement
- Minimal `pk` size + `sig` size
- Make adjusting security levels simple
- $q = 2^{23} - 2^{13} + 1 = 8380417$, a 24-bit prime number
  - $2n \mid (q-1)$
  - w a primitive 256-th root of unity in $\mathbb{Z}_q$,i.e., $w^n \equiv 1 \mod q$
  - $\phi = 1753$ a primitive 512-th root of unity in $\mathbb{Z}_q$ such that $\phi^2 = w$
- Rejection Sampling to make the signature independant on `sk`

CRYSTALS - Dilithium

KeyGen:

1- $(s_1,\ s_2) \in S_\eta^l \times S_\eta^k$
2- $\mathbf{A} \in \mathcal{R}_q^{k\times l} := \mathtt{ExpandA}(\rho)$
3- $t := \mathbf{A}s_1 + s_2$
4- $(t_1,\ t_0) := \mathtt{Power2Round}_q(t,\ d)$
5- $tr \in \{0,\ 1\}^{384} := \mathtt{CRH}(\rho \,||\, t_1)$
6- return $\mathtt{pk} = (\rho, t_1)$, $\mathtt{sk} = (\rho, s_1, s_2, t_0, tr)$

Verify $(\mathbf{pk}, M, \sigma)$**:**

1- $\mu \in \{0,\ 1\}^{384} := \mathtt{CRH}(\mathtt{CRH}(\rho \,||\, t_1) \,||\, M)$
2- $w_1' := \mathtt{UseHint}_q(h,\ \mathbf{A}z - ct_1,\ 2\gamma_2)$
3- if $||z||_\infty < \gamma_1 - \beta$ and $c == \mathtt{H}(\mu \,||\, w_1')$
and $|h|_{h_i=1} \leq \omega$:
4- return *True*
5- return *False*

Sign $(M, \mathbf{sk})$ **:**

1- $\mathbf{A} \in \mathcal{R}_q^{k\times l} := \mathtt{ExpandA}(\rho)$
2- $\mu = \mathtt{CRH}(tr \,||\, M)$
3- while $(z, h) = \perp$ do
4- $y \xleftarrow[\$]{} [-\gamma_1,\ \gamma_1]^l$
5- $w = Ay$
6- $w_1 := \mathtt{HighBits}_q(w, 2\gamma_2)$
7- $c = \mathtt{H}(\mu \,||\, w_1)$
8- $z = y + cs_1$
9- $(r_1, r_0) = \mathtt{Decompose}(w - cs_2)$
10- if $||z||_\infty \geq \gamma_1 - \beta$ or $||r_0||_\infty \geq \gamma_2 - \beta$:
$(z, h) = \perp$
11- $h := \mathtt{MakeHint}_q(w - cs_2 + ct_0,\ 2\gamma_2)$
12- return $\sigma = (c, z, h)$

Outline

State of the art attacks on Dilithium

Type	Description	Nb of samples	Ref.
	Dilithium		
FA	KeyGen, Force s_1, s_2 with, Nonce re-use & EM Key Recovery	$k + l$ fault complexity	[1]
DPA [1]	Sign, cs_1 both on textbook and Sparse multiplication	Vertical DPA Horizontale + Vertical	[2]
SPA [1]	Sign, 1 bit leakage in y plus analytical reconstruction	10000 traces 10000 traces	[3]
ML	Sign, Unmasked : $\mathtt{NTT}\ (s_1,\ s_2,\ t_0)$ Masked : Multiplication cs_1	Attack phase : 8000 traces Learning phase : 2000 traces Learning phase : 9000 traces	[4]
CPA	Sign, cs_1 or cs_2 on all type of multiplicaton	100 traces	[5]
DFA	Sign, modify c (μ, w)	2 executions	[6]
LFA	Sign, fault 1 coeff in addition of cs_1 and y	$2 \times N$	[7]
SASCA	Dec, $\mathtt{NTT}^{-1}$ on su	1 trace 196 intermediate values 100 million traces TM 20 iterations of BP	[8]
SASCA [1]	Enc, $\mathtt{NTT}$ on r	213 templates 2 304 intermediates	[9]

Table: Matrix of Attack Paths of Dilithium

Countermeasures

- Operations more suited for boolean masking: rejection sampling, random sampling
- Other parts for arithmetic masking: multiplications and additions modulo q
- Maybe conversion from both type of masks

	Dilithium		
Type	DPA	ML	CPA
Boolean Masking	✓✓	✗	✓
Shuffling	✗	✓	✓
Blinding	✓	✓	✓

Outline

1 Introduction

2 Theoretical Background

3 State Of the art
- Attacks
- Countermeasures

4 Constructive Results on Dilithium
- Leakage Identification
- Fault Simulation

Leakage Identification

Dilithium:

- Round 3 signature size even larger: 2420 bytes
- Round 2 Deterministic Dilithium1-AES: 1387 bytes
- Sample Analysed
  - CPU: 32 bits
  - Total RAM: 12k

Without loss of generality analysis made on list of 1 round messages for a fixed key

- Side Channel:
  - Focus on leakage identification with EM traces
  - Previous Working algorithm in C on chip to collect side channel Data
  - Developed version in Sage/ Python to collect intermediate values/ to simulate faults
  - From there one can apply different analysis (DPA, CPA, Template, ML, SASCA ..)
- Fault Attack:
  - Focus on fault attack that can be simulated

Leakage Identification

Dilithium:

- Round 3 signature size even larger: 2420 bytes
- Round 2 Deterministic Dilithium1-AES: 1387 bytes
- Sample Analysed
  - CPU: 32 bits
  - Total RAM: 12k

Without loss of generality analysis made on list of 1 round messages for a fixed key

- **Side Channel:**
  - Focus on leakage identification with EM traces
  - Previous Working algorithm in C on chip to collect side channel Data
  - Developed version in Sage/ Python to collect intermediate values/ to simulate faults
  - From there one can apply different analysis (DPA, CPA, Template, ML, SASCA ..)
- **Fault Attack:**
  - Focus on fault attack that can be simulated

Leakage Identification

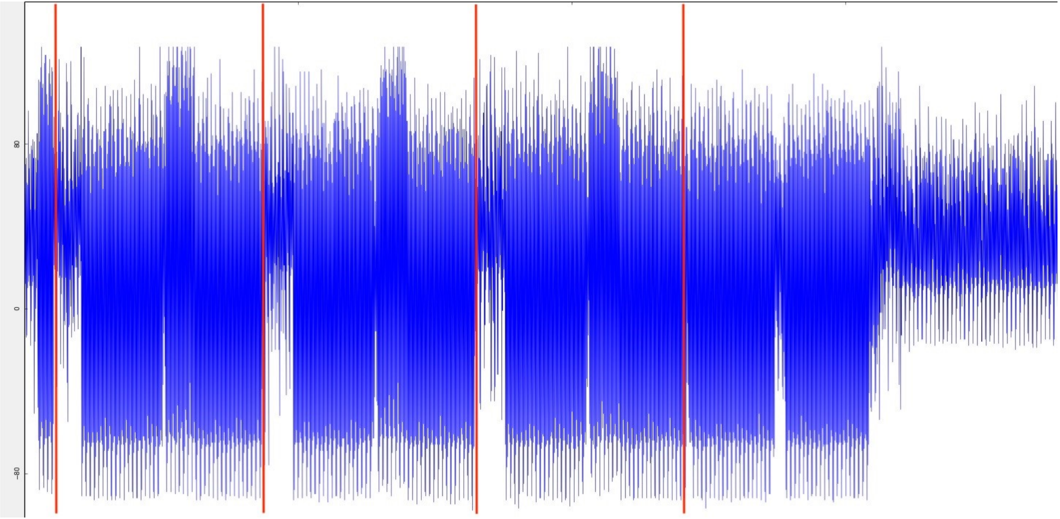
Dilithium:

- Round 3 signature size even larger: 2420 bytes
- Round 2 Deterministic Dilithium1-AES: 1387 bytes
- Sample Analysed
  - CPU: 32 bits
  - Total RAM: 12k

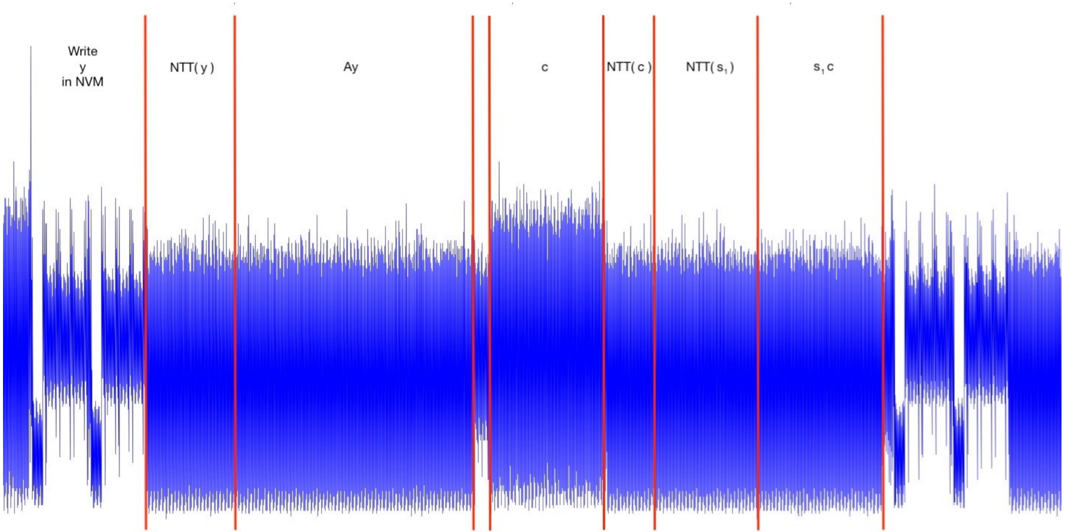
Without loss of generality analysis made on list of 1 round messages for a fixed key

- **Side Channel:**
  - Focus on leakage identification with EM traces
  - Previous Working algorithm in C on chip to collect side channel Data
  - Developed version in Sage/ Python to collect intermediate values/ to simulate faults
  - From there one can apply different analysis (DPA, CPA, Template, ML, SASCA ..)
- **Fault Attack:**
  - Focus on fault attack that can be simulated

Reverse Engineering

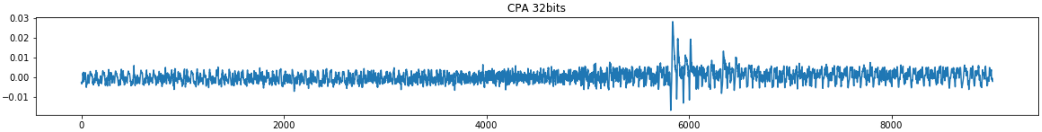


Reverse Engineering



What about the CPA ?

- Here let's focus on the first coefficient $\hat{c}\hat{s}_1$ with 270K traces
- Leakage even with considering a 32 bits values HW model



- Same thing with SNR, ANOVA, NICV
- Even leakage with Power Traces
- If implemented directly attack time: 16 years
- Simple parallel version using 32 CPUs in asynchronous mode: 10 months

What about the CPA ?

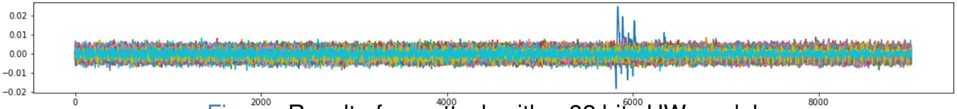


Figure: Result of our attack with a 32 bits HW model.

- “Attack“ mounted on the first coefficient of $\hat{s}_1$ with 1000 random keys
- Repeat $l \times n$ this procedure to complete the attack

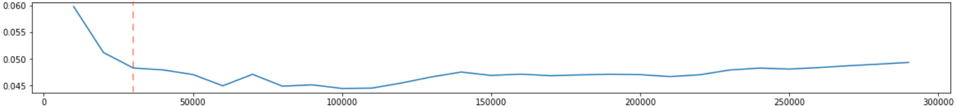
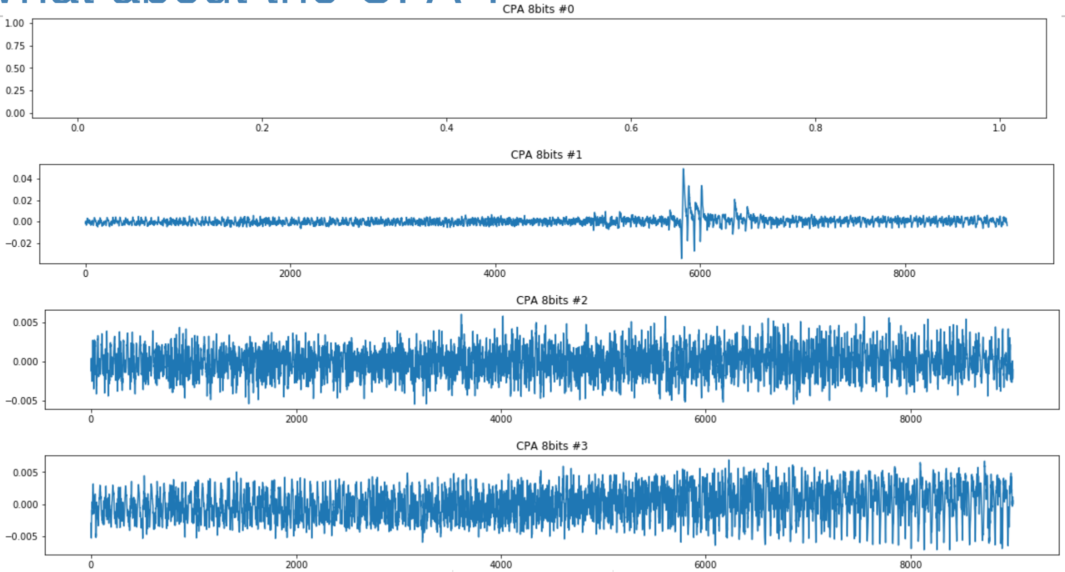


Figure: Correlation value as a function of number of traces.

- Less traces could still differentiate the correct key value

What about the CPA ?



Masking the NTT with a Twist

"On configurable SCA countermeasure against single trace attacks for the NTT'[10]':

The twiddle factors as a mask: because for $(a + bw_x)w_y = aw_y + bw_x w_y = aw_y + bw_{x+y}$

- N masks per stage: Mask space 2^{4196}
- One mask per stage: Mask space 2^{63} $\longleftarrow$ **Implemented version**
- Output unmasked:
  - The 8 masks need to sum up to a multiple of $2 \times n$
- Output masked:
  - We return the product of the 8 masks (another twiddle factor)
  - If we multiply two masked polynomials we add the masks (still fits on $uint\_32$)
  - INTT masked with masks that unmask the result

NTT version	Number of cycles
Unmasked	676665
Masked	1229961

Fault Model

- Possible to inject a single random Fault
- Instruction skips
- Arithmetic faults
- Glitches in storage
- Many more
- Not only restricted to single operations
- Can be applied during a large section of code

DFA on Deterministic Lattice Signatures [6]

- Force a nonce reuse
- Target the computation of c
- Why: Differential Fault Attack
  - First sign without fault
    $(c, z, h) = \textbf{Sign}\ (M, \mathbf{sk})$
  - Second time with fault
    $(c', z', h) = \textbf{Sign}\ (M, \mathbf{sk})$
- $z' = y + c's_1$ and $z = y + cs_1$
- $z - z' = \cancel{y} + cs_1 - \cancel{y} - c's_1$
  $= (c - c')s_1$
- $s_1 = (c - c')^{-1}(z - z')$

Sign $(M, \mathbf{sk})$ **:**

1- $A = \texttt{ExpandA}\ (\rho)$
2- $\mu = \mathrm{CRH}(tr \,||\, M)$
3- while $(z, h) = \perp$ do
4- $\quad y \xleftarrow{\$} [-\gamma_1,\ \gamma_1]^l$
5- $\quad w = Ay$
6- $\quad c = \mathrm{H}(\mu \,||\, \texttt{HighBits}(w)) \Longleftarrow$
7- $\quad z = y + cs_1$
8- $\quad (r_1, r_0) = \texttt{Decompose}(w - cs_2)$
9- $\quad$ if $||z||_\infty \geq \gamma_1 - \beta$ or $||r_0||_\infty \geq \gamma_2 - \beta$:
$\qquad (z, h) = \perp$
10- return $\sigma = (c, z, h)$

THALES

How to perform such a modification ?

- Change the value of c without changing other values and the number of rejections

Name	Description	Success Probability	Size of vulnerable code
fA$_\rho$	Corrupt ρ during import of **sk**	14.3	1.37
fA$_E$	Random fault in expansion A	54.4	20.1
fW	Random fault in multiplication w	25.4/ 90.3	3.35
fH	Random fault in call to H	91	1.07
fY	Random fault in sampling y	24.5	2

Table: Different ways of faulting the c polynomial.

- The scenario **fY** is discarded because uses partial nonce reuse
- Focus made on **fH** and **fW**

Scenario fH

- Inject a random fault into the computation of $c = \mathsf{H}(\mu \;||\; \texttt{HighBits}(w))$
- Faulting μ inside the function H or Faulting $\texttt{HighBits}(w)$ inside the function H
- Fault the function itself (change the value of a coefficient, fault in SHAKE)

▷ Faulting μ

- Three different hypothesis
  - Zero out a byte
  - Zero out a 4-bytes word
  - Zero out all the 48 bytes
- On our 200K messages corpus zeroing all the 48 bytes result in 99% of success rate !
- ... but can be hard to do
- On average there is ≈ 11 of the 12 4-bytes words that can be zeroed out and achieve a correct signature under the same number of rejections
- For a single byte ≈ 46 bytes can be targeted

How to check for correctness ?

- Check the computation time for the faulted variable
- Recover the theoretical value of s_1 and check if the values satisfy the distribution S_η^l
- Alternatively one can compute $||z - z'||$ and check if it is below a threshold
- What next? Modified Sign algorithm that produces valid signatures with only s_1

Countermeasures

- Double Computation
  - Doubles the runtime and adds storage space
  - Injecting the same fault twice can counter the countermeasure
- Verification after sign
  - Some faults result in incorrect signatures
  - Runtime cost of verifying a signature is $\approx$ one third of the signing one
- Use randomness
  - Use random version of Dilithium
  - Need of a good enough source of entropy
  - Will depend on the standard version chosen by the NIST

Conclusions and Future Work

- How realistic were the state of the art attacks ?
  - Side Channel: Leakage exploitation doable ... but not as easy to mount a whole attack
  - Fault Attacks: Actual version sensible to faults
    Maybe push for the probabilistic version as a standard
- Masking the NTT: With twiddle factors reasonable overhead for thwarting two attack paths
- Investigation of SASCA: Louvain University SCALib on Github (for AES)
  - Actual work of Thales (on AES) showes it to be less effecive than anticipated
  - Maybe results on Dilithium within a month
- Leakage assessment of masked Dilithium
- Exploiting possible attack path on s_2

Conclusion

Questions?

Bibliography I

[1] P. Ravi, D. B. Roy, S. Bhasin, A. Chattopadhyay, and D. Mukhopadhyay, *Number "not used" once - practical fault attack on pqm4 implementations of nist candidates*, Cryptology ePrint Archive, Report 2018/211, https://eprint.iacr.org/2018/211, 2018.

[2] P. Ravi, M. P. Jhanwar, J. Howe, A. Chattopadhyay, and S. Bhasin, *Side-channel assisted existential forgery attack on dilithium - a nist pqc candidate*, Cryptology ePrint Archive, Report 2018/821, https://eprint.iacr.org/2018/821, 2018.

[3] Y. Liu, Y. Zhou, S. Sun, T. Wang, R. Zhang, and J. Ming, *On the security of lattice-based fiat-shamir signatures in the presence of randomness leakage*, Cryptology ePrint Archive, Report 2019/715, https://eprint.iacr.org/2019/715, 2019.

[4] I.-J. Kim, T.-H. Lee, J. Han, B.-Y. Sim, and D.-G. Han, *Novel single-trace ml profiling attacks on nist 3 round candidate dilithium*, Cryptology ePrint Archive, Report 2020/1383, https://eprint.iacr.org/2020/1383, 2020.

Bibliography II

[5] A. P. Fournaris, C. Dimopoulos, and O. Koufopavlou, "Profiling dilithium digital signature traces for correlation differential side channel attacks," in *Embedded Computer Systems: Architectures, Modeling, and Simulation*, A. Orailoglu, M. Jung, and M. Reichenbach, Eds., Cham: Springer International Publishing, 2020, pp. 281–294, ISBN: 978-3-030-60939-9.

[6] L. G. Bruinderink and P. Pessl, *Differential fault attacks on deterministic lattice signatures*, Cryptology ePrint Archive, Report 2018/355, `https://eprint.iacr.org/2018/355`, 2018.

[7] P. Ravi, M. P. Jhanwar, J. Howe, A. Chattopadhyay, and S. Bhasin, *Exploiting determinism in lattice-based signatures - practical fault attacks on pqm4 implementations of nist candidates*, Cryptology ePrint Archive, Report 2019/769, `https://eprint.iacr.org/2019/769`, 2019.

[8] R. Primas, P. Pessl, and S. Mangard, *Single-trace side-channel attacks on masked lattice-based encryption*, Cryptology ePrint Archive, Report 2017/594, `https://eprint.iacr.org/2017/594`, 2017.

Bibliography III

[9] P. Pessl and R. Primas, *More practical single-trace attacks on the number theoretic transform*, Cryptology ePrint Archive, Report 2019/795, `https://eprint.iacr.org/2019/795`, 2019.

[10] P. Ravi, R. Poussier, S. Bhasin, and A. Chattopadhyay, *On configurable sca countermeasures against single trace attacks for the ntt - a performance evaluation study over kyber and dilithium on the arm cortex-m4*, Cryptology ePrint Archive, Report 2020/1038, `https://eprint.iacr.org/2020/1038`, 2020.

Current attack path

- For performances we compute: $(r_1, r_0) = \texttt{Decompose}(w_0 - cs_2)$
- $\texttt{UseHint}(h, Az - ct_1 \cdot 2^d, 2\gamma_2) = \texttt{HighBits}(w - cs_2, 2\gamma_2) := w_1$
- $\texttt{UseHint}(h', Az' - c't_1 \cdot 2^d, 2\gamma_2) = \texttt{HighBits}(w' - c's_2, 2\gamma_2) := w'_1$
- $\tilde{c} = \mathsf{H}(\mu \,||\, w_1)$
- $\tilde{c}' = \mathsf{H}(\mu \,||\, w'_1)$

Sign_faulted $(M, \mathbf{sk})$:

1- $A = \texttt{ExpandA}\ (\rho)$
2- $\mu = \mathsf{CRH}(tr \,||\, M)$
3- while $(z, h) = \perp$ do
4- $\quad y \xleftarrow[\$]{} [-\gamma_1,\ \gamma_1]^l$
5- $\quad w = Ay$
6- $\quad c = \mathsf{H}(\mu \ \ ||\ \texttt{HighBits}(w))$
7- $\quad z = y + cs_1$
8- $\quad (r_1, r_0) = \texttt{Decompose}(w - cs_2) \Longleftarrow$
9- $\quad$ if $||z||_\infty \geq \gamma_1 - \beta$ or $\cancel{||r_0||_\infty \geq \gamma_2 - \beta}$:
$\quad\quad (z, h) = \perp$
10- return $\sigma' = (c', z', h')$

A few details about the papers

- **Title:** Number "Not Used" Once - Practical fault attack on pqm4 implementations of NIST candidates [1] :
- **Authors:** Prasanna Ravi and Debapriya Basu Roy and Shivam Bhasin and Anupam Chattopadhyay and Debdeep Mukhopadhyay
- **What:** Key Recovery and Message Recovery Attack
- **How:** EM Fault to skip store instruction
- **On what:** s_1 and s_2 (resp. s and e) sampling in KeyGen for Dilithium (resp. Kyber)
- **Setup:** pqm4 implementation on an ARM Cortex-M4 microcontroller
- **Results:** 100% repeatability with custom prob

A few details about the papers

- **Title:** Side-channel Assisted Existential Forgery Attack on Dilithium - A NIST PQC candidate [2] :
- **Authors:** Prasanna Ravi and Mahabir Prasad Jhanwar and James Howe and Anupam Chattopadhyay and Shivam Bhasin
- **What:** DPA
- **How:** Two stages DPA on sparse and DPA textbook multiplier version of Dilithium
- **On what:** cs_1 multiplication in Sign
- **Setup:** Simulated setting, uniform noise supposing 8 bit Hamming Weight leakage and linear regression model noise
- **Results:** HW model with up to noise in $[-6,\ 6]$ 75% coefficients retrieved . LR model up to same level of noise 90% retrieved with none to brute force (on average).

A few details about the papers

- **Title:** On the Security of Lattice-based Fiat-Shamir Signatures in the Presence of Randomness Leakage [3] :
- **Authors:** Yuejun Liu and Yongbin Zhou and Shuo Sun and Tianyu Wang and Rui Zhang and Jingdian Ming
- **What:** Generic Key Recovery attack supposing leakage of randomness
- **How:** Recovery of one bit of randomness, instance of FS-ILWE and analytical attack
- **On what:** $z = y + cs_1$ addition in Sign
- **Setup:** Certain and probabilistic leakage of the bit, profiling of power traces of sensitive operation without and with artificial noise
- **Results:** Up to 0.65 % even with $\sigma = 10$ noise.

A few details about the papers

- **Title:** Novel Single-Trace ML Profiling Attacks on NIST 3 Round candidate Dilithium [4] :
- **Authors:** Il-Ju Kim and Tae-Ho Lee and Jaeseung Han and Bo-Yeon Sim and Dong-Guk Han
- **What:** Single Trace Attack
- **How:** Target load, save and store instructions on operations involving private key
- **On what:** Sign
  - Unprotected version: Montgomery Reduction of NTT representation of sensitive variable
  - Masked Version: Sparse multiplication of challenge with sensitive variables
- **Setup:** ARM Cortex M4 microcontroller of Dilithium II
- **Results:** Success rate of 100%

A few details about the papers

- **Title:** Single-Trace Side-Channel Attacks on Masked Lattice-Based Encryption [8] :
- **Authors :** Robert Primas and Peter Pessl and Stefan Mangard
- **What:** Single Trace Attack
- **How:**
  - Side Channel Template Matching
  - Factor Graph of butterfly atomic operation, BP algorithm
  - Lattice decoding on reduced size of pk
- **On what:** Target $\mathtt{NTT}^{-1}(su)$ on Dec of generic lattice based PKE scheme
- **Setup:** EM measurement for real device experiment
- **Results:** 80% up to $\sigma = 0.5$ in the Noisy Hamming Weight leakage Model

A few details about the papers

- **Title:** More Practical Single-Trace Attacks on the Number Theoretic Transform [9] :
- **Authors:** Peter Pessl and Robert Primas
- **What:** Single Trace Attack
- **How:** New Factor Graph and SASCA method
- **On what:** Target save, store and load on NTT of r in Enc
- **Setup:** Hamming Weight Templates on pqm4 Kyber
- **Results:** Success rate of $\approx 57\%$ on a real device

A few details about the papers

- **Title:** Profiling Dilithium Digital Signature Traces for Correlation Differential Side Channel Attacks [5] :
- **Authors:**Apostolos P. Fournaris, Charis Dimopoulos and Odysseas Koufopavlou
- **What:** Correlation Power Attack
- **How:** Correlation Power Attack
- **On what:** Target cs_1 cs_2 multiplication in last round of in Sign
- **Setup:** Hamming Weight model on real noisy device
- **Results:** Polynomial Operation visible in the trace