Correlation electromagnetic analysis on an FPGA implementation of CRYSTALS-Kyber

Rafael Carrera Rodriguez, Florent Bruguier, Emanuele Valea, Pascal Benoit

Correlation Electromagnetic Analysis on an FPGA Implementation of CRYSTALS-Kyber

Rafael Carrera Rodriguez*†, Florent Bruguier*, Emanuele Valea†, Pascal Benoit*
*LIRMM, University of Montpellier, CNRS, Montpellier, France
{rafael.carrera-rodriguez, florent.bruguier, pascal.benoit}@lirmm.fr
†Univ. Grenoble Alpes, CEA, List, F-38000 Grenoble, France
{rafael.carrerarodriguez, emanuele.valea}@cea.fr

***Abstract*—Post-quantum cryptography represents a category of cryptosystems resistant to quantum algorithms. Such schemes are under the scrutiny of their mathematical security in the context of the NIST standardization process, but they are not side-channel secure at the algorithm level. That is why their side-channel vulnerabilities must be assessed by the research community. In this paper, we present a non-profiled correlation electromagnetic analysis against an FPGA implementation of the standard key-encapsulation mechanism, CRYSTALS-Kyber. The attack correlates an electromagnetic radiation model of the polynomial multiplication execution with the captured traces. With 166,620 traces, this attack correctly recovers 100% of the subkeys. Furthermore, a countermeasure is presented for securing the target implementation against the presented attack.**

***Index Terms*—correlation power analysis, FPGA, PQC**

I. Introduction

CRYSTALS-Kyber [1] is a recently standardized quantum resistant key-encapsulation mechanism (KEM) by the National Institute of Standards and Technologies (NIST) [2]. Similarly to all Post-Quantum Cryptography (PQC) primitives, Kyber implementations can be vulnerable to side-channel analyses (SCA), that take advantage of physical quantities such as power consumption and electromagnetic radiation to obtain sensitive information, namely the plaintext or the secret key.

Since the opening of the NIST PQC standardization effort, many SCA against lattice-based cryptography (LBC) schemes have been presented in the literature, mainly on software implementations. However, most of them are either profiling attacks [3], [4] and/or attacks against the Chosen-Ciphertext-Attack (CCA) security [5], [6]. Some of these works assume an attacker model with increased capabilities, such as being able to communicate directly with the attacked server and send carefully crafted invalid ciphertexts, like the works from [3], [5]–[7]. In [8], the authors perform a correlation power analysis (CPA) [9], targeting an ARM Cortex-M4 software implementation of Kyber. A CPA is an unprofiled, divide-and-conquer type of SCA that attempts to find a correlation between captured power traces of the device and a leakage model, depending on a key hypothesis.

In this work, a variation of such attack is presented, targeting the FPGA hardware implementations of Kyber from [10], using a power model that leverages the attacker's knowledge of the device's algorithm and using electromagnetic traces. Furthermore, a lightweight countermeasure, specific to this implementation and to this attack, is also proposed. Unlike the work from [11] that also attacks the same implementation, no profiling or model training stage is needed nor the ability for an attacker to send specially crafted ciphertexts.

This paper is organized as follows: Section II presents the background necessary for the understanding of the contributions of this paper. Section III presents the proposed correlation electromagnetic attack on Kyber and its results. Section IV presents the lightweight countermeasure. Finally, in Section V, conclusions are drawn.

II. Background

A. CRYSTALS-Kyber, Implementation and its Vulnerability

Kyber is a KEM, used to establish a shared private message between two parties, e.g., a client and a server. At first, the server generates a public and a secret key. The client receives the public key and uses it to encrypt the generated message, producing a ciphertext. Finally, the server decrypts the received ciphertext using the its secret key. In Kyber, the decryption function contains the operation $m = v - \mathbf{s} \cdot \mathbf{u}$, where $\mathbf{s}$ is the secret key, the tuple $(\mathbf{u}, v)$ is the ciphertext and m is the decrypted message. m and v are polynomials of degree 256 with integer coefficients. $\mathbf{u}$ and $\mathbf{s}$ are *vectors of polynomials* of size k, where $k \in [2, 4]$ that depends on the security level. Thus, each polynomial of the $\mathbf{s}$ vector can be represented as $\mathbf{s}[j]$, for $j \in [0, k-1]$. Meanwhile, the coefficients of $\mathbf{s}[j]$ can be represented as $\mathbf{s}[j]_i$, for $i \in [0, 255]$. The same is valid for the $\mathbf{u}$ vector. For these reasons, the scalar product requires the execution of several polynomial multiplications. The complexity of the polynomial multiplication can be reduced from $O(n^2)$ to $O(n \log n)$ using the Number Theoretic Transform (NTT), which works like a Fast-Fourier Transform over the ring of integers. The multiplication of two polynomials, transformed in the NTT domain, can be solved by a *point-wise multiplication* (PWM), whose complexity is linear with the number of coefficients, obtaining the following expression:

$$m = v - \text{NTT}^{-1}(\text{NTT}(\mathbf{s}) \circ \text{NTT}(\mathbf{u})) = v - \text{NTT}^{-1}(\hat{\mathbf{s}} \circ \hat{\mathbf{u}}) \quad (1)$$

where the operator $\circ$ represents the inner product in the NTT domain, i.e., executed with a PWM of each $\hat{\mathbf{s}}[j]$ and $\hat{\mathbf{u}}[j]$ polynomials.

In the FPGA implementation that has been targeted by the proposed attack [10], the arithmetic operations are performed in the NTT core. This core contains two parallel and pipelined butterfly units used for NTT and inverse NTT, as well as for other operations, including the PWM. The result of each PWM is accumulated, in order to complete the inner product $\hat{\mathbf{s}} \circ \hat{\mathbf{u}}$. The PWM operation is executed in two clock cycles. In the first cycle, i.e., PWM0, the following operations are executed in parallel:

$$\begin{aligned} n_0 = \hat{\boldsymbol{s}}[j]_{2i} + \hat{\boldsymbol{s}}[j]_{2i+1},\ n_1 = \hat{\boldsymbol{u}}[j]_{2i} + \hat{\boldsymbol{u}}[j]_{2i+1}, \\ m_0 = \hat{\boldsymbol{s}}[j]_{2i} \cdot \hat{\boldsymbol{u}}[j]_{2i},\ m_1 = \hat{\boldsymbol{s}}[j]_{2i+1} \cdot \hat{\boldsymbol{u}}[j]_{2i+1} \end{aligned} \quad (2)$$

Since the products $\hat{\boldsymbol{s}}[j]_{2i} \cdot \hat{\boldsymbol{u}}[j]_{2i}$ and $\hat{\boldsymbol{s}}[j]_{2i+1} \cdot \hat{\boldsymbol{u}}[j]_{2i+1}$ are performed between a part of the ciphertext (i.e., a value that can be known by an attacker) and the secret key (i.e., information that the attacker wants to retrieve), this are the vulnerable operations that we used as the target of the presented side-channel attack.

B. Correlation Power/Electromagnetic Analysis

A correlation power analysis (CPA) [9] and its analog, correlation electromagnetic analysis (CEMA), is a divide-and-conquer type of attack that targets an operation with a known value and a part of the secret key, using a leakage model. The steps of such an analysis are the following:

1) The attacker retrieves n traces $\boldsymbol{t}_0, \boldsymbol{t}_1, \ldots, \boldsymbol{t}_{n-1}$ of m sample points corresponding to the execution of a targeted operation, with $i_0, i_1, \ldots, i_{n-1}$ known inputs. For simplicity, the set of traces will be referred as the matrix $\boldsymbol{T}$ of size $n \times m$, where each row is the trace of an execution.
2) The attacker targets an unknown subkey s_0 and makes a guess g_0 from the set of l possible values of s_0.
3) According to his knowledge of the device, the attacker uses a power consumption model denoted as $M(i_j, g_0)$, depending on a certain known input i_j, for each point k of the trace $\boldsymbol{T}[j, \cdot]$. Then, the attacker will calculate the vector $\boldsymbol{h}_0$ of n points, where $\boldsymbol{h}_0 = M(i_j, g_o)$ for each input given to the device.
4) The attacker uses the Pearson's correlation coefficient $\rho(X, Y)$ to obtain a vector $\boldsymbol{c}_0$ of size m, where each point is obtained as $\rho(\boldsymbol{T}[\cdot, j]^T, \boldsymbol{h}_0)$ and $\boldsymbol{T}[\cdot, j]$ is equal to the column vector $(\boldsymbol{T}[0, j], \boldsymbol{T}[1, j], \ldots, \boldsymbol{T}[n-1, j])$. That is, the vector $\boldsymbol{c}_0$ contains the correlation coefficient of each of the sample points with the model vector $\boldsymbol{h}_0$.
5) The attacker repeats steps 2-4 for guesses $g_1, g_2, \ldots, g_{l-1}$, obtaining the vectors $\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_{l-1}$.
6) The attacker uses a strategy to choose the most likely guess. For example, the attacker can choose the guess g_j that yields the vector $\boldsymbol{c}_j$ that contains the point with the maximum absolute value.
7) The attacker repeats steps 2-5 for the rest of unknown subkeys until the whole key is retrieved.

C. Usage of Hamming distance as leakage model

As it has been anticipated in Section II-B, a leakage model is required. If z is an intermediate targeted value and r is the previous value of the circuit part that produces z, called reference value, a consumption model that represents well the switching activity of a device is the Hamming distance model, $HW(z \oplus r)$, where $\oplus$ is a bitwise XOR and $HW(\cdot)$ is the Hamming weight, that is the number of 1's in the binary representation of the input.

For using such model, a vector of reference values must be known or guessed. In the target implementation [10], the reference values for attacking $\hat{\boldsymbol{s}}[j]_0$ and $\hat{\boldsymbol{s}}[j]_1$ can be known by the attacker. In fact, these values are the result produced by the multipliers in the last stage of the execution of the NTT of $\boldsymbol{u}[j]$ and before starting the first PWM0 cycle of the PWM operation. Then, for obtaining such result, it suffices to revert the last steps used to obtain $\hat{\boldsymbol{u}}[j]$.

The rest of the reference values depend on the success of the attack on the previous subkeys. They correspond to the previous result $m_0 = \hat{\boldsymbol{u}}[j]_{2(i-1)} \cdot \hat{\boldsymbol{s}}[j]_{2(i-1)}$ for the even values and to $m_1 = \hat{\boldsymbol{u}}[j]_{2(i-1)+1} \cdot \hat{\boldsymbol{s}}[j]_{2(i-1)+1}$ for the odd values in cycle PWM0 in (2), for $i \in [1, 127]$. After executing the acquisition campaign, the analysis phase must be carried out sequentially, obtaining at first the subkeys $\hat{\boldsymbol{s}}[j]_0$ and $\hat{\boldsymbol{s}}[j]_1$, in order to obtain the next reference values for establishing the Hamming distance model.

III. CEMA on Kyber Hardware Implementation

A. Attack scenario and setup

An FPGA evaluation general-purpose board was programmed with the implementation from [10]. A UART interface for communication with the computer and a trigger to start the capture of the traces on the desired PWM operation have been added to the target implementation. The evaluation board is a Digilent Basys-3 with a Xilinx Artix-7 XC7A35TCPG236 FPGA chip. The signal is shaped by a Langer RF-U 5-2 EM probe coupled with a Femto HSA-X-2-40 amplifier and captured with a Tektronix MSO64 oscilloscope configured to use a low-pass filter with a cut-off frequency of 200 MHz. The sample rate of the oscilloscope was fixed to 1.25 GS/s, whereas the FPGA was programmed to use a clock of 62.5 MHz generated by the Artix-7 chip. Therefore, the number of samples per clock cycle is 20. 15 sets of about 11k traces are used (166,620 traces in total), calculating the correlation incrementally with each set. Instead of using the full traces for attacking each subkey, it was only selected the clock cycle where the result of the multiplication with the subkey is being updated. This was done to reduce the runtime. A scheme of the setup is shown in Figure 1.

The security parameter k from Kyber was set to 2, the lowest one, for ease of analysis. It should be noted that increasing the security parameter should not make the attack more difficult. Because of the divide-and-conquer nature of this attack, increasing k would only lead to a linear increase of the attack runtime.

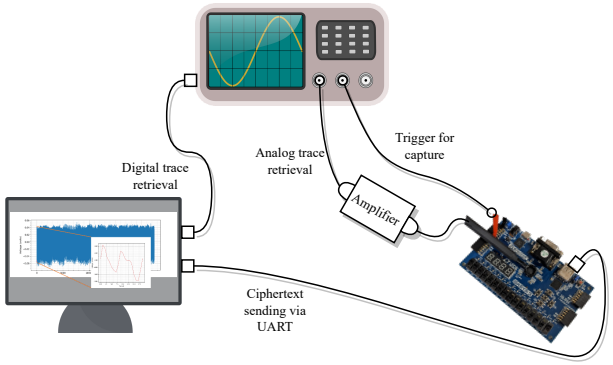

Figure 1: A scheme of the setup, with the connections between computer, oscilloscope, FPGA board, EM probe and amplifier.

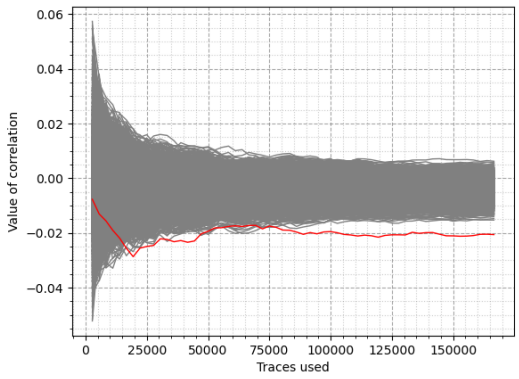

Figure 2: Highest correlation of the first sample for each key guess in function of the number of sets of traces used for the subkey $\hat{\boldsymbol{s}}[0]_0$.

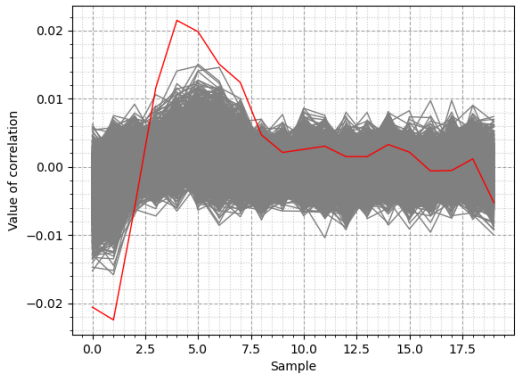

Figure 3: Correlation coefficient for all samples considered for subkey $\hat{\boldsymbol{s}}[0]_0$ for each key guess after 166,620 traces are used. In red, correct key guess.

B. Results

The trace capture campaign took around 6 hours and 45 minutes, with around 2 hours and 45 minutes of runtime for the analysis part, using a laptop with an Intel Core i7-11850H processor. The success rate SR, when executing the test attack in a non-sequential way is 100%. In fact, for the sake of simplicity, in this test attack, the secret key and all reference values are known beforehand. The SR is the accuracy of finding the correct subkeys when only selecting the guess with the highest correlation coefficient. Even if the attack had been sequential, SR would still hold at 100%, because there is not an incorrect guess that would induce to have false reference values and prevent the analysis of further subkeys.

In Fig. 2, the highest correlation for the first sample, of each key guess in function of the number of sets used for the subkey $\hat{\boldsymbol{s}}[0]_0$ is shown. In red, the highest correlation of the correct key guess is shown, surpassing visibly the other key guesses. In Fig. 3, the correlation coefficient of all samples considered for subkey $\hat{\boldsymbol{s}}[0]_0$ after 166,620 traces is shown. The correct guess is shown in red, and it is visibly above other key guesses in multiple points.

IV. Low-area countermeasure

This attack leverages the hypothesis that the device is leaking the Hamming distance of the results of the attacked multiplier. The attacker knows the first reference values for the first subkeys, because the NTT of the ciphertext is performed before the PWM between the secret vector and the ciphertext in the NTT domain. If a random dummy operation is performed in the multiplier after the NTT and before the PWM, the attacker will not know the reference value and hence, the attack as it is presented here will not work. In this case, the decision was made to use a simple linear-feedback shift register (LFSR) to provide the dummy inputs of the multiplier. The LFSR used is a maximal LFSR of degree 24. It is initialized at `0xaaaaaa`, started when the device is powered up and never stopped. The first 12 bits of the LFSR state are used as the first input of both of the butterfly units of the target implementation. The last 12 bits are used as the second input of such units, as well as the twiddle factor inputs. This random dummy multiplication is performed just k times, before the start of each PWM sequence. This countermeasure has a time overhead of k clock cycles. When looking at area, it uses 7694 LUT and 4953 flip-flops, with an overhead of 3.80% and 6.65% respectively, compared to the original implementation in [10].

For evaluating the countermeasure, the same attack procedure is used. In this case, instead of using 15 sets of about 11k traces, 150 sets are used, for a total of 1,666,200 traces. That is, ten times more traces. Also, the attack is only done against subkeys $\hat{\boldsymbol{s}}[j]_0$ and $\hat{\boldsymbol{s}}[j]_1$, for $j \in [0, k-1]$, using the samples in the traces where the multiplication with these coefficients is being done. The attack is not successful for any of the analyzed subkeys, even when using this number of samples. In Fig. 4, the highest correlation, for all considered samples, of each key guess in function of the number of sets used for the subkey $\hat{\boldsymbol{s}}[0]_0$ is shown. In red, the highest correlation of the correct key guess is shown, being visibly similar to other key guesses.

In Fig. 5, the correlation coefficient of all samples considered for subkey $\hat{\boldsymbol{s}}[0]_0$ is shown. The correct guess is shown in red, and it does not surpass other key guesses at any point.

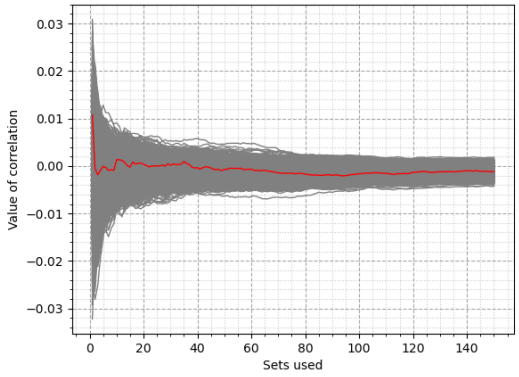

Figure 4: Highest correlation of all samples for each key guess in function of the number of sets of about 11k traces used for the subkey $\hat{\boldsymbol{s}}[0]_0$ after countermeasure. In red, correct key guess.

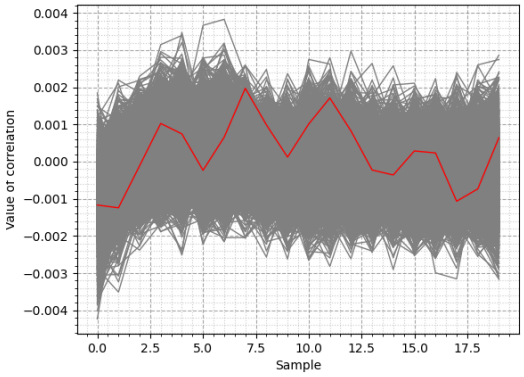

Figure 5: Correlation coefficient for all samples considered for subkey $\hat{\boldsymbol{s}}[0]_0$ for each key guess after all traces are used. In red, correct key guess.

However, the countermeasure has certain limitations. It works well because this attack assumes the leakage of the Hamming distance of the hardware multiplier result within the FPGA. If another attack were to use a different power model, this countermeasure would not work. Another limitation is that an attacker could try to guess the first two couples of coefficients together, $\hat{\boldsymbol{s}}[j]_0$ with $\hat{\boldsymbol{s}}[j]_2$and $\hat{\boldsymbol{s}}[j]_1$ with $\hat{\boldsymbol{s}}[j]_3$, for $j \in [0, k-1]$, for the first coefficient in each couple determines the reference value for the attack on the second coefficient. The number of possible guesses for both couples is $3329^2 < 2^{24}$, since all coefficients are in the interval $[0, 3328]$. After this strategy, the attacker can continue executing the attack as proposed in Section II-B. Therefore, even if this countermeasure do not offer a complete protection all alone, it could be used as a building block with other countermeasures in a real implementation.

V. Conclusion

An unprofiled correlation electromagnetic attack on a compact hardware implementation of Kyber is presented. It recovers the secret subkeys of the Kyber-512 version with a success rate of 100%, given the knowledge of register reference values. Furthermore, a lightweight countermeasure has been presented against this specific attack and platform. This work stresses the need of research for securing post-quantum cryptography implementations.

References

[1] R. Avanzi, J. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. M. Schanck, P. Schwabe, G. Seiler, and D. Stehlé, "CRYSTALS-Kyber: Algorithm Specifications and Supporting Documentation," 2020. [Online]. Available: https://pq-crystals.org/kyber/.

[2] National Institute of Standards and Technology, "Post-Quantum Cryptography Standardization Process." [Online]. Available: https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization.

[3] M. Hamburg, J. Hermelink, R. Primas, S. Samardjiska, T. Schamberger, S. Streit, E. Strieder, and C. V. Vredendaal, "Chosen Ciphertext k-Trace Attacks on Masked CCA2 Secure Kyber," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 88–113, 2021.

[4] R. Primas, P. Pessl, and S. Mangard, "Single-Trace Side-Channel Attacks on Masked Lattice-Based Encryption," in *Lecture Notes in Computer Science*, pp. 513–533, Springer International Publishing, 2017.

[5] S. Bhasin, J.-P. D'Anvers, D. Heinz, T. Pöppelmann, and M. V. Beirendonck, "Attacking and defending masked polynomial comparison for lattice-based cryptography," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 334–359, 2021.

[6] R. Ueno, K. Xagawa, Y. Tanaka, A. Ito, J. Takahashi, and N. Homma, "Curse of Re-encryption: A Generic Power/EM Analysis on Post-Quantum KEMs," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 296–322, 2021.

[7] Z. Xu, O. M. Pemberton, S. Sinha Roy, D. Oswald, W. Yao, and Z. Zheng, "Magnifying Side-Channel Leakage of Lattice-Based Cryptosystems with Chosen Ciphertexts: The Case Study of Kyber," *IEEE Transactions on Computers*, 2021.

[8] C. Mujdei, A. Beckers, J. Bermudo Mera, A. Karmakar, L. Wouters, and I. Verbauwhede, "Side-channel analysis of lattice-based post-quantum cryptography: Exploiting polynomial multiplication." Cryptology ePrint Archive, Paper 2022/474, 2022.

[9] E. Brier, C. Clavier, and F. Olivier, "Correlation Power Analysis with a Leakage Model," in *Lecture Notes in Computer Science*, pp. 16–29, Springer Berlin Heidelberg, 2004.

[10] Y. Xing and S. Li, "A Compact Hardware Implementation of CCA-Secure Key Exchange Mechanism CRYSTALS-KYBER on FPGA," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 328–356, 2021.

[11] Y. Ji, R. Wang, K. Ngo, E. Dubrova, and L. Backlund, "A side-channel attack on a hardware implementation of crystals-kyber." Cryptology ePrint Archive, Paper 2022/1452, Oct. 2022.