IACR Transactions on Cryptographic Hardware and Embedded Systems
ISSN 2569-2925, Vol. 2020, No. 3, pp. 307–335. DOI:10.13154/tches.v2020.i3.307-335

# Generic Side-channel attacks on CCA-secure lattice-based PKE and KEMs

Prasanna Ravi[1,2], Sujoy Sinha Roy[3], Anupam Chattopadhyay[1,2] and Shivam Bhasin[1]

[1] Temasek Laboratories, Nanyang Technological University, Singapore
[2] School of Computer Science and Engineering, Nanyang Technological University, Singapore
prasanna.ravi@ntu.edu.sg, sbhasin@ntu.edu.sg, anupam@ntu.edu.sg
[3] School of Computer Science, University of Birmingham, United Kingdom
s.sinharoy@cs.bham.ac.uk

**Abstract.** In this work, we demonstrate generic and practical EM side-channel assisted chosen ciphertext attacks over multiple LWE/LWR-based Public Key Encryption (PKE) and Key Encapsulation Mechanisms (KEM) secure in the *chosen ciphertext model* (IND-CCA security). We show that the EM side-channel information can be efficiently utilized to instantiate a *plaintext checking* oracle, which provides binary information about the output of decryption, typically concealed within IND-CCA secure PKE/KEMs, thereby enabling our attacks. Firstly, we identified EM-based side-channel vulnerabilities in the error correcting codes (ECC) enabling us to distinguish based on the value/ validity of decrypted codewords. We also identified similar vulnerabilities in the Fujisaki-Okamoto transform which leaks information about decrypted messages applicable to schemes that do not use ECC. We subsequently exploit these vulnerabilities to demonstrate practical attacks applicable to *six* CCA-secure lattice-based PKE/KEMs competing in the second round of the NIST standardization process. We perform experimental validation of our attacks on implementations taken from the open-source *pqm4* library, running on the ARM Cortex-M4 microcontroller. Our attacks lead to complete key-recovery in a matter of *minutes* on all the targeted schemes, thus showing the effectiveness of our attack.

**Keywords:** Lattice-based cryptography · EM-based side-channel attack · LWE/LWR · Chosen Ciphertext Attack · Public Key Encryption · Key Encapsulation Mechanism

## 1 Introduction

NIST is currently conducting a global standardization process for post-quantum public-key cryptographic algorithms which are secure against attacks from quantum computers [NIS16]. This process started in 2017 with about 69 candidates in the first round, based on a variety of hard problems considered to be intractable by quantum computers. After intense scrutiny from the cryptographic community, about 26 candidates (17 Public-key Encryption (PKE) and Key Encapsulation Mechanisms (KEM) and 9 Digital Signature schemes (DS)) were selected for the second round. While the main selection criterion for the first round had been theoretical security and uniqueness of the schemes, the second round will also consider implementation aspects such as performance on both hardware and software platforms, bandwidth and resistance to side-channel attacks (SCA). Twelve out of the 26 candidates in the second round are based on computationally hard problems from the lattice theory and in particular, variants of the Learning With Errors (LWE) or Learning With Rounding (LWR) problem. Several research works have investigated

Received: 2020-01-15 Accepted: 2019-03-15 Published: 2020-06-19

the efficiency aspects of these schemes through implementations on a wide-range of hardware [PDG14, HMO+16] and software platforms [OPG14, POG15, KMRV18, BKS19]. But, the side-channel security aspect of these lattice-based cryptographic schemes has received only limited attention [PPM17, BP18, OSPG18, Pes16]. *In this paper, we analyze side-channel security of several LWE/LWR-based PKE/KEMs from the second round of the NIST's standardization project.*

It is well known that LWE/LWR-based PKE schemes that are secure in the chosen plaintext model (IND-CPA secure) are vulnerable to chosen ciphertext attacks when the secret key is reused [DCQ19, QCD19, BDHD+19, BGRR19, Flu16]. These attacks mainly work by assuming the presence of a *key mismatch* or *plaintext checking* oracle, a weaker assumption compared to a *decryption* oracle. The *plaintext checking* oracle typically provides a binary response (correct or wrong) about the attacker's guess of the decrypted message (resp. shared secret key) of a PKE (resp. KEM) for a given chosen ciphertext. The attacker carefully chooses his query ciphertexts in such a way that the corresponding oracle's responses reveal the secret key. However, LWE/LWR-based PKE/KEMs can be transformed to be secure against such adaptive chosen ciphertext attacks (IND-CCA secure) by applying well known CCA conversions such as the Fujisaki-Okamoto (FO) transform [FO99]. One of the main requirements when applying CCA conversions for IND-CCA security is to have negligible decryption failure rate (about $2^{-\lambda}$ for $\lambda$-bit security). All LWE/LWR schemes are associated with a certain failure rate due to the inherent presence of noise. While some schemes [BDK+18, ADPS16] tune their parameter set to achieve negligible decryption failure rates, others [BBF+19, LLZ+18] however rely on additional forward Error Correcting Codes (ECC) to increase robustness against errors.

The FO transform mainly involves re-encryption after decryption which enables to detect invalid or maliciously formed ciphertexts and return failure upon detection. Thus malicious/invalid chosen ciphertexts will always be rejected by the decapsulation and hence the attacker cannot gain any meaningful information from decryption of chosen ciphertexts, thereby removing the presence of the *plaintext checking* oracle. However, this is only true in a classical *black box* setting since any cryptographic algorithm implemented on a real-device leaks information about intermediate values through side-channels such as timing, power consumption or Electromagnetic (EM) emanation. In fact, D'Anvers in [DTVV19] showed that difference in the execution times of variable-time error correcting procedures can be used to distinguish validity of decrypted output codewords with very high probability. They utilized this timing vulnerability as a *plaintext checking* oracle to demonstrate practical chosen ciphertext attacks over IND-CCA secure KEMs LAC [LLZ+18] and RAMSTAKE [Sze17]. While their attack can be prevented through implementation of constant-time ECC procedures [WR19, BBF+19], it still raises critical questions about existence of other side-channel vulnerabilities within CCA transformed decryption procedures that could lead to practical chosen ciphertext attacks.

In this work, we extend and generalize chosen ciphertext attacks over IND-CCA secure lattice-based PKE/KEMs to the EM side-channel setting demonstrating practical key recovery attacks applicable to *six* such schemes. We identify two types of side-channel vulnerabilities to gain information about the decryption output. They are present (1) within constant time error correcting procedures in schemes that use ECC and (2) within operations of the FO transform in schemes that do not use ECC. We exploit these vulnerabilities to instantiate a practical *plaintext checking* oracle to perform full key-recovery over multiple IND-CCA secure lattice-based PKE/KEMs. Our work reiterates the need for concrete masking countermeasures for lattice-based schemes against side-channel attacks whose significance has also been stressed by several previous works [RdCR+16, RRVV15, OSPG18].

## Contributions

The main contributions of this work are as follows:

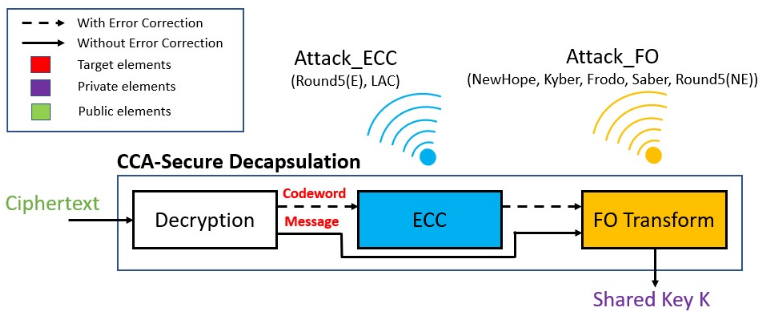


**Figure 1:** Pictorial representation of our proposed attacks on the IND-CCA secure decapsulation procedure[2]

1. We demonstrate generic and practical EM side-channel assisted chosen ciphertext attacks applicable to six IND-CCA secure LWE/LWR based PKE/KEMs which are also round 2 candidates in the ongoing NIST standardization process. We demonstrate very efficient strategies to instantiate the EM side-channel as a precise *plaintext checking* oracle, which facilitates our attacks over such unprotected schemes.

2. We identify EM-based side channel vulnerabilities within constant-time decoding procedures of ECC used in the Round5 and LAC PKE/KEMs which are exploited to identify the validity of decrypted codewords (Attack_ECC). We identify similar vulnerabilities in the FO transform of the decapsulation procedure which are exploited to gain information about the decrypted messages, leading to full key-recovery in schemes that do not utilize ECC such as Kyber, NewHope, Saber, Frodo and variants of Round5 that do not use ECC (Attack_FO).

3. We adopt a simple and generic approach towards generating chosen ciphertexts to uniquely identify the secret key candidates based on decryption outputs and show that our technique is adaptable to multiple LWE/LWR-based PKE/KEMs. Our approach also leads to a key recovery attack over NewHope that works with a 100% recovery rate in the presence of a perfect oracle.

4. We perform experimental validation of our attacks on the implementations of the aforementioned NIST candidates obtained from the *pqm4* public library, a testing and benchmarking framework for post quantum cryptographic schemes on the ARM Cortex-M4 microcontroller [KRSS]. The pre-processing phase of our attack only requires to distinguish two values of decrypted messages or codewords and subsequent key recovery can be performed on all the targeted schemes in a matter of *minutes*. Please refer to Fig.1 for the pictorial description of both our attacks (Attack_ECC and Attack_FO) targeting the IND-CCA secure decapsulation operation[1].

5. While prior works have stressed on the need to protect the re-encryption procedure within the decapsulation operation against possible side-channel attacks [RdCR$^+$16, RRVV15, OSPG18, BGRR19], our work however provides substantial evidence of easy susceptibility to side-channel assisted chosen ciphertext attacks resulting in full key-recovery. Our work thus reiterates the need for protection of IND-CCA secure lattice-based schemes through concrete masking countermeasures against side-channel attacks.

[1]It is common practice for LWE/LWR schemes to include the decoding procedure of ECC within the decryption operation. But for better understandability of the concepts explained in the paper, we decouple the decoding procedure of ECC from the decryption operation and treat it as a separate function.

[2]We denote Round5(E) (Round5(NE)) as those variants of Round5 which utilize (don't utilize) ECC after decryption respectively.

#### Availability of software

All softwares utilized for this work is placed into public domain. They are available at `https://github.com/PRASANNA-RAVI/Generic-SCA-CCA-Lattice-Schemes`.

#### Organization of the Paper

This paper is organized as follows. Section 2 covers the necessary background by introducing the required concepts, Section 3 presents our side-channel-assisted chosen-ciphertext attack on the IND-CCA secure Round5 algorithm, through exploitation of EM-leakage from its ECC as a *plaintext checking* oracle. Section 4 presents a very similar attack on the IND-CCA secure LAC algorithm. Section 5 demonstrates that EM-leakage from the FO transform can also be used as an oracle to attack multiple LWE/LWR-based schemes that do not use ECC. Section 6 discusses potential countermeasures against our proposed attacks and finally, Section 7 concludes our paper.

## 2 Lattice Preliminaries

### 2.1 Notation

We denote the ring of integers modulo a prime $q$ as $\mathbb{Z}_q$. The polynomial ring $\mathbb{Z}_q(x)/\phi(x)$ is denoted as $R_q$ where $\phi(x)$ is its reduction polynomial. We denote $\mathbf{r} \in R_q^{k\times\ell}$ as a *module* of dimension $k \times \ell$. Polynomials in $R_q$ and modules in $R_q^{k\times\ell}$ are denoted in bold lower case letters. The $i^{th}$ coefficient of a polynomial $\mathbf{a} \in R_q$ is denoted as $\mathbf{a}[i]$ and the $i^{th}$ polynomial of a given module $\mathbf{x}$ as $\mathbf{x}_i$. Matrices/vectors in $\mathbb{Z}_q^{k\times l}$ are denoted in bold upper case letters. Multiplication of two polynomials $\mathbf{a}$ and $\mathbf{b}$ in the ring $R_q$ is denoted as $\mathbf{c} = \mathbf{a} \times \mathbf{b}$. We define two operations $\mathsf{Rotr}(\mathbf{a}, p)$ and $\mathsf{Anti\_Rotr}(\mathbf{a}, p)$ which rotate the polynomial $\mathbf{a} \in R_q$ by $p$ positions to the left in a cyclic and anti-cyclic fashion respectively, as described in Eqn.1-2.

$$\mathsf{Rotr}(\mathbf{a}, p)[i] = \begin{cases} \mathbf{a}[n-p+i], & \text{for } 0 \le i < p \\ \mathbf{a}[i-p], & \text{for } p \le i \le n-1 \end{cases} \tag{1}$$

$$\mathsf{Anti\_Rotr}(\mathbf{a}, p)[i] = \begin{cases} -\mathbf{a}[n-p+i], & \text{for } 0 \le i < p \\ \mathbf{a}[i-p], & \text{for } p \le i \le n-1 \end{cases} \tag{2}$$

A negative value for $p$ denotes corresponding cyclic and anti-cyclic rotations towards the right. An element $\mathbf{x} \in R_q$ sampled from the distribution $\mathcal{D}$ with standard deviation $\sigma$ is denoted as $\mathbf{x} \leftarrow \mathcal{D}(R_q)$. Byte arrays of length $n$ are denoted as $\mathcal{B}^n$. The $i^{th}$ bit in an element $x \in \mathbb{Z}_q$ is denoted as $x_i$. The acquisition of a side-channel trace $t$ corresponding to a particular operation $\mathcal{X}$ using an input $p$ is denoted as $t \Longleftarrow \mathcal{X}(p)$.

### 2.2 Learning With Error/Learning With Rounding Problem

The Learning With Errors (LWE) problem, introduced by Regev [Reg09] is one of the most well known average-case hard problems upon which several lattice-based NIST candidates have been constructed. Solving the LWE problem on random lattices in the *average* case is at least as hard as solving the related Bounded Distance Decoding (BDD) problem on the same lattices in the *worst* case. There are two versions of the LWE problem - Search LWE and Decisional LWE. The search variant of the LWE problem requires the attacker to solve for a secret $\mathbf{S} \in \mathbb{Z}_q^{\ell\times n}$ given polynomially many LWE instances of the form $(\mathbf{A}, \mathbf{T} = \mathbf{A} \times \mathbf{S} + \mathbf{E}) \in (\mathbb{Z}_q^{k\times\ell} \times \mathbb{Z}_q^{k\times n})$, with $\mathbf{A} \leftarrow \mathcal{U}(\mathbb{Z}_q^{k\times\ell})$ and $\mathbf{E} \leftarrow \mathcal{D}_\sigma(\mathbb{Z}_q^{k\times n})$ where $\mathcal{U}$ represents uniform distribution and $\mathcal{D}_\sigma$ represents a discrete error distribution with standard deviation $\sigma$. The decisional variant of this problem requires the attacker to distinguish similarly structured ordered pairs of LWE instances $(\mathbf{A}, \mathbf{T}) \in (\mathbb{Z}_q^{k\times\ell} \times \mathbb{Z}_q^{k\times n})$ from uniformly random pairs $\mathcal{U}(\mathbb{Z}_q^{k\times\ell} \times \mathbb{Z}_q^{k\times n})$. While errors in the LWE problem are sampled from an error distribution $\mathcal{D}_\sigma(\mathbb{Z}_q^{k\times n})$ and explicitly added to the linear system of equations $(\mathbf{A} \times \mathbf{S}) \in \mathbb{Z}_q^{k\times n}$, Banerjee *et al.* [BPR12] showed that it is possible to implicitly add errors to the linear system $\mathbf{A} \times \mathbf{S}$ by scaling down the coefficients from $\mathbb{Z}_q$ to $\mathbb{Z}_p$ with $p < q$ resulting in instances such as $(\mathbf{A}, \mathbf{T} = \lfloor \frac{p}{q}(\mathbf{A} \times \mathbf{S})\rceil) \in (\mathbb{Z}_q^{k\times\ell} \times \mathbb{Z}_p^{k\times n})$. Such ordered pairs are referred to as the Learning With Rounding (LWR) instances which are also associated with their correspondingly hard search and decisional problems [LPR10].

However, most of the LWE/LWR based NIST candidate PKE/KEMs, except for Frodo are based on algebraically structured variants of the standard LWE and LWR problem known as the Ring/Module-LWE (RLWE/MLWE) and Ring/Module-LWR (RLWR/MLWR) problems respectively. The ring variant of the LWE/LWR problem (RLWE/RLWR) [LPR10] deals with computation over polynomials in polynomial rings $R_q$ with $\mathbf{s}, \mathbf{e} \leftarrow \mathcal{D}_\sigma(R_q)$ such that the corresponding RLWE instance is defined as $(\mathbf{a}, \mathbf{t} = \mathbf{a}\times\mathbf{s}+\mathbf{e}) \in (R_q \times R_q)$ and the RLWR instance is defined as $(\mathbf{a}, \mathbf{t} = \lfloor \mathbf{a} \times \mathbf{s}\rceil_{q\to p}) \in (R_q \times R_p)$. The module variant deals with computations over vectors/matrices of polynomials in $R_q^{k_1\times k_2}$ with $(k_1, k_2) > 1$. With $\mathbf{a} \leftarrow \mathcal{U}(R_q^{k_1\times k_2})$ and $\mathbf{s} \leftarrow \mathcal{D}_\sigma(R_q^{k_2})$ and $\mathbf{e} \leftarrow \mathcal{D}_\sigma(R_q^{k_1})$. the corresponding MLWE instance is defined as $(\mathbf{a}, \mathbf{t} = \mathbf{a} \times \mathbf{s} + \mathbf{e}) \in (R_q^{k_1\times k_2}, R_q^{k_2})$ and the MLWR instance is defined as $(\mathbf{a}, \mathbf{t} = \lfloor \mathbf{a} \times \mathbf{s}\rceil_{q\to p}) \in (R_q^{k_1\times k_2}, R_p^{k_2})$.

Most PKE/KEMs based on the LWE/LWR problem contain in their core an IND-CPA secure public-key encryption scheme based on a general paradigm/framework proposed by Lyubashevskey, Peikert and Regev in 2010 [LPR10], now known as the "LPR Encryption scheme". All these schemes follow a standard *encryption-based* approach of generating the message (key) at one end and hiding it within the ciphertext. However, there also exists another parallel approach for key exchange called the *reconciliation-based* approach used in schemes such as BCNS [BCNS15], NewHope-USENIX [ADPS16] and Frodo-CCS [BCD+16], which can be seen as a direct analogue of the Diffie-Hellman key exchange scheme. However, most of the round 2 round 2 NIST PKE/KEMs have adopted the *encryption-based* approach due to its simplicity compared to the *reconciliation-based* approach. We refer the reader to [DVV18] for a generalized description of the LPR encryption scheme. Schemes following this framework only differ in the low level details such as structure of the underlying ring ($R_q$, $R_q^{k\times k}$, $\mathbb{Z}_q^{n\times n}$), choice of LWE/LWR, usage of ECC, secret/error distribution, relative sizes of the rounding moduli etc.

## 2.3 Error Correcting Codes in LWE/LWR-based schemes

Almost all semantically secure lattice-based PKE schemes based on the LWE/LWR problem are associated with a certain decryption failure probability. Thus, the associated PKE/KEMs schemes may encounter a failure event when the two involved parties fail to establish a shared secret key. Designers strive to achieve a non-negligible failure probability if not zero, as it serves as a key-requirement to achieve security against adaptive chosen-ciphertext attacks [Cac04]. Moreover, it also prevents attacks which mainly work by triggering decryption failures [DVV18].

But, designing schemes with negligible decryption failures while also guaranteeing high security could impose a significant penalty over performance. Fritzmann *et al.* [FPS18] performed a systematic study of utilizing forward error correcting codes to artificially reduce decryption failures by correcting errors in the decrypted message. The message $m$ to be encrypted is first input to the ECC's encoding procedure to create a codeword $c$. This codeword is subsequently encrypted to generate the ciphertext $ct$. The decryption operation decrypts $ct$ to yield $c'$, which could have some erroneous bits. The ECC's decoding procedure subsequently extracts the message $m'$ from the possibly erroneous codeword. If the number of bit errors in $c'$ is less than the ECC's maximum error correcting capability, then $m' = m$ and hence will result in correct decryption. The LAC KEM utilizes heavy BCH [HP10] error correcting code, while Round5 PKE utilizes a constant-time linear parity code XEf [BBF+19] ECC.

## 2.4 CCA Transformation

An LWE/LWR-based PKE/KEM secure in the CPA model can be converted into a IND-CCA secure PKE/KEM using the well known Fujisaki-Okamoto transform [FO99]. Variants of the same transformation are used by multiple lattice-based NIST candidates to achieve IND-CCA security. It utilizes a pair of hash functions $\mathcal{H}$ and $\mathcal{G}$ and operates over the top of the encryption and decryption schemes and Alg.1 describes a generic FO transform of a IND-CPA secure PKE to a IND-CCA secure KEM. Upon decapsulation failures, some schemes simply return a failure symbol $\perp$, while certain schemes such as Kyber [BDK+17] and Saber [DKRV18] use a variant that returns a pseudo random value as the shared secret key, generated by hashing the ciphertext with the secret key.

In theory, the FO transform helps protect KEMs against chosen-ciphertext attacks since the validity of ciphertexts are checked through the re-encryption procedure during decapsulation (Line 8 of KEM.Decaps). Thus, the attacker only sees decapsulation failures for any handcrafted (invalid) ciphertext. Moreover, the decryption procedure is encapsulated within the decapsulation procedure and hence the attacker cannot directly observe the output of the decryption module. This provides strong theoretical security guarantees against chosen-ciphertext attacks which are possible over IND-CPA secure PKE/KEMs. But in this work, we show that side-channel information from operations within the FO transform leak enough information about the decrypted message, that can be used to mount chosen-ciphertext attacks over CCA secure PKE/KEMs.

**Algorithm 1:** FO transform of a IND-CPA secure PKE into a IND-CCA secure KEM

```
1  Procedure KEM.Encaps($pk$, Ecc_Used)
2      $\rho \leftarrow \mathcal{U}(\mathcal{B}^{32})$
3      $m = \mathcal{H}(\rho)$
4      $r = \mathcal{G}(m, pk)$
5      if Ecc_Used = 1 then
6          $c$ = Ecc_Enc($m$)
7      end
8      else
9          $c = m$
10     end
11     $ct$ = PKE.Encrypt($pk, c, r$)
12     $K = \mathcal{H}(r)$
13     return $ct, K$
14 ----------------------------------------------------------------
1  Procedure KEM.Decaps($sk, pk, ct$, Ecc_Used)
2      $c'$ = PKE.Decrypt($sk, ct$)
3      if Ecc_Used = 1 then
4          $m'$ = Ecc_Dec($c'$)
5      end
6      else
7          $m' = c'$
8      end
9      $r' = \mathcal{G}(m', pk)$
10     $ct'$ = PKE.Encrypt($pk, m', r'$)
11     if $ct' = ct$ then
12         return $K = \mathcal{H}(r' \| ct')$
13     end
14     else
15         return $K = \perp$ or $K = \mathcal{H}(z \| ct')$ /* $z \in \mathcal{B}^{32}$ is a random secret */
16     end
```

## 2.5 Chosen Ciphertext Attacks on LWE/LWR based schemes

Key-reuse or Key-caching is a commonly adopted technique in secure protocols such as TLS and IKE (Internet Key Exchange). Security of IND-CPA secure public key cryptographic schemes in a key-reuse scenario in the presence of a *plaintext chekcing* oracle has been studied for a long time, dating back to the *reaction attacks* such as Bleichenbacher's attack over RSA PKCS#1 in 1998 [Ble98] and attacks over standard instantiations of the Diffie-Hellman key exchange [HGS99, MU10]. Security of LWE/LWR-based schemes in a similar scenario was first studied by Fluhrer in [Flu16]. He proposed chosen ciphertext attacks over IND-CPA secure *reconciliation-based* key exchange schemes. This attack assumes reuse of secret key by the initiator of the key exchange procedure and the adversary (responder) gathers information about key mismatches for specially crafted ciphertexts leading to full key recovery. Ding *et al.* [DAS+17] proposed another attack that assumes the initiator to be the adversary who observes changes in the *reconciliation* hints for chosen public key shares to recover the secret key, thus showing that such attacks work in both directions. Subsequent works proposed different approaches towards key recovery in *reconciliation-based* key exchange schemes [DFR18, LZZ18, DXL12, Pei14].

Subsequently, IND-CPA secure LWE/LWR-based PKE/KEMs following the *encryption* approach were also shown to be vulnerable to *plaintext checking* oracle based attacks by a number of works [DCQ19, BGRR19, BDHD+19, QCD19, DTVV19], albeit using different techniques compared to attacks over *reconciliation-based* schemes. Ding *et al.* [DCQ19] and Bŭaetu *et al.* [BDHD+19] proposed generic approaches towards chosen ciphertext attacks on multiple *encryption-based* PKE/KEMs. However, the aforementioned attacks particularly fail to attack or be trivially adaptable to schemes such as NewHope that utilize (1) redundancy-based message encoding and ciphertext compression functions and (2) large span of the secret coefficients, which complicates the interaction with the *plaintext checking* oracle. However, Bauer *et al.* [BGRR19] demonstrated the first chosen ciphertext attack over NewHope which recovers coefficients from a certain smaller range than the total span while brute-forcing the remaining coefficients and their attack was subsequently improved by Qin *et al.* [QCD19] which recovers all coefficients, albeit with a success rate of about 96.88%.

While the aforementioned attacks primarily targeted IND-CPA secure schemes, Bauer *et al.* [BGRR19] also suggested utilizing side-channel information to instantiate *plaintext checking* oracles and also performed preliminary experiments using EM side-channel information to argue their case. More concretely, D'Anvers *et al.* [DTVV19] demonstrated use of timing leakage information to perform chosen ciphertext attacks over the IND-CCA secure LAC KEM and also stated the possibility of using other side-channels to perform chosen ciphertext attacks. In this work, we extend and generalize the idea of side-channels being used as efficient *plaintext checking* oracles by demonstrating practical EM-side channel assisted key recovery attacks applicable to six IND-CCA secure LWE/LWR based PKE/KEMs. In fact, our attack approach is also adaptable to NewHope resulting in key recovery with 100% success rate in the presence of a perfect oracle.

## 2.6 Test Vector Leakage Assessment (TVLA)

The Test Vector Leakage Assessment (TVLA) [GGJR+11] is a popular conformance-based methodology in side-channel analysis which has been widely used by both academia and the industry to perform side-channel evaluation of cryptographic implementations. TVLA involves computation of the well known univariate Welch's $t$-test over two given sets of side-channel measurements to identify differentiating features in them. By testing for a null hypothesis such that the mean of two sets is identical, a *PASS/FAIL* decision is taken.

The formulation of TVLA over two sets of measurements $\mathcal{T}_r$ and $\mathcal{T}_f$ is given by:

$$TVLA = \frac{\mu_r - \mu_f}{\sqrt{\frac{\sigma_r^2}{m_r} + \frac{\sigma_f^2}{m_f}}} \quad , \tag{3}$$

where $\mu_r$, $\sigma_r$ and $m_r$ (resp. $\mu_r$, $\sigma_r$ and $m_r$) are mean, standard deviation and cardinality of the trace set $\mathcal{T}_r$ (resp. $\mathcal{T}_f$). The null hypothesis is rejected with a confidence of 99.9999% only if the absolute value of the $t$-test score is greater than 4.5 [GGJR+11]. A rejected null hypothesis implies that the two trace/data sets are different and might leak some side-channel information and hence is considered a *FAIL* test. While TVLA is mainly used as a metric for side-channel evaluation, it has been also been used as a tool for feature selection in multiple cryptanalytic efforts [RJJ+18]. In this work, we use TVLA as a tool for *feature selection* from side-channel measurements [GLRP06].

## 3 Side-Channel Analysis of Round5

We analyze the power/EM side-channel behaviour of constant-time decoding procedures of error correcting codes in IND-CCA secure Round5 PKE/KEM. It utilizes the XEf error correction scheme, which is an $f$-bit forward error correcting block code which can always correct at least $f$ errors in any given codeword. In the following discussion, we describe the functioning of XEf code and highlight its differential behaviour between valid and invalid codewords observable through the EM-side channel.

### 3.1 XEf error correction code

Let the $k$-bit message to be encoded be $m = (m_{k-1}, m_{k-2}, \ldots, m_0)$ and its corresponding binary polynomial be defined as $mp = m_{k-1}x^{k-1} + \ldots + m_1 x + m_0$. XEf is a linear parity code and works with a register set $r$ with $2f$ registers ($r_i$ for $i \in \{0, 2f-1\}$) each of size $l_i$ such that $r_i = mp \bmod (x^{l_i} - 1)$. The codeword $c = \mathsf{Ecc\_Enc}(m)$ consists of the message $m$ appended with the register set $r$ (i.e) $c = (m|r)$, which amounts to a total of $\mu$ bits where $\mu = k + \sum_{i=0}^{2f-1} l_i$. Let us denote the decrypted codeword output from the decryption logic as $c' = (m'|r')$. The decoding computes the register set $r''$ for the received message $m'$ and compares it with $r'$ in the codeword. Those bits $j$ in the received message $m'$ are flipped that satisfy the following condition that $\sum_{i=1}^{2f}((r''_{(i,j \bmod l_i)} - r'_{(i,j \bmod l_i)}) \bmod 2\,) \geq (f+1)$. One of the main advantages of XEf is that its encoding and decoding procedures can be easily implemented in constant-time and are thus resistant to timing attacks. However, we show in the following discussion that constant time implementations of XEf still leak exploitable information about the codeword through the EM-side channel.

### 3.2 Side-channel vulnerability of XEf

We analyze the decoding of XEf in the optimized implementation of Round5 taken from the open-source *pqm4* library [KRSS]. We ran the optimized implementation of the IND-CCA secure Round5 KEM (`R5ND_1KEM_5d` variant in particular) on the STM32F4DISCOVERY board (DUT) housing the STM32F407, ARM Cortex-M4 microcontroller. The implementation (compiled with `-O3 -mthumb -mcpu=cortex-m4 -mfloat-abi=hard -mfpu=fpv4-sp-d16`) was running at a clock frequency of 24 MHz. We used the ST-LINK/v2.1 add-on board for UART communication with our DUT. We used the OpenOCD framework for flash configuration and on-chip hardware debugging with the aid of the GNU debugger for ARM (arm-none-eabi-gdb).

The XEf decoding involves a decision-making operation to decide upon the position of bits to be flipped in the recovered message. One of the steps in this decision-making operation involves computation of a majority logic operation, denoted as $\mathcal{M}$. It takes as input, a modified form of the register set $r$, which we denote as $r'$. Moreover, this is also the last operation within the decoding procedure that decides the bits to be corrected. We observe over multiple trials that $r' = 0$ when a codeword is valid, while it is non-zero for erroneous codewords with one or more erroneous bits (i.e) $r \neq 0$. Since the inputs to $\mathcal{M}$ differ based on the validity of the codeword, the computations also differ accordingly. In particular, we consider an all zero valid codeword $c = 0 \in \{0,1\}^*$ and an all zero codeword with a single bit error in its lsb (i.e) $c = 1 \in \{0,1\}^*$.Please refer to Fig.5 in the appendix for the screenshots from debugging using Openocd [Rat05] clearly showing the difference in the internal register values (at the same point during computation) for the two cases $c = 0$ and $c = 1$[3]. This differential behaviour based on the validity of the codeword can be easily detected through the EM side-channel. EM measurements were observed from the same DUT using a near-field probe and processed using a Lecroy 610Zi oscilloscope at a sampling rate of 500MSam/sec. Please refer to Fig.6 in Appendix B for our experimental setup used to perform EM trace acquisition. We adopt a Welch's $t$-test based reduced template approach to classify a given power/EM trace to either a valid ($c = 0$) or invalid ($c = 1$) codeword, which will be discussed in the following section.

## 3.3 Welch's *t*-test based template approach for classification

Our classification technique consists of two stages. The first is a *pre-processing* stage which involves generating a *reduced template* for each class and the second stage involves the template matching operation. For the *pre-processing* stage, we collect two sets of 50 measurements each, denoted as $\mathcal{T}_{\mathrm{O}}$ and $\mathcal{T}_{\mathrm{X}}$, corresponding to repeated executions of the decoding for codewords $c = 0$ and $c = 1$ respectively. Let $\mathcal{T} = \mathcal{T}_{\mathrm{O}} \cup \mathcal{T}_{\mathrm{X}}$ and we construct a *reduced template* for each trace set as follows:

- Center (normalize) each trace $t_i \in \mathcal{T}$ by removing the mean of each trace from itself such that $t'_i = t_i - \bar{t}_i$ where $\bar{t}_i$ is the mean of trace $t'_i$. This step is optional but can correct some environmental effects on measurements, such as DC shifts.
- We utilize the Welch's $t$-test to identify the points that clearly distinguish between $\mathcal{T}_{\mathrm{O}}$ and $\mathcal{T}_{\mathrm{X}}$. Refer to Fig.2 for the corresponding $t$-test plot. The observation of high peaks in the $t$-test plot (greater than the pass/fail threshold $\pm 4.5$) validates our hypothesis that the computations are significantly different at these points.
- We select only those points whose *absolute t-test value* (both $\pm$) is greater than a certain chosen threshold[4] $Th_{sel}$ as our set of Points of Interest (PoI) $\mathcal{P}$ and a reduced trace set $\mathcal{T}'_{\mathrm{O}}$ and $\mathcal{T}'_{\mathrm{X}}$ is constructed with these points within $\mathcal{P}$. We subsequently calculate their respective means denoted as $m_{\mathrm{O},\mathcal{P}}$ and $m_{\mathrm{X},\mathcal{P}}$ respectively, which are the *reduced templates* for each class.

Given a single trace $t$ for classification, an attacker normalizes the trace such that $t' = t - \bar{t}$ and obtains a reduced trace $t'_{\mathcal{P}}$. He can then compute the sum-of-squared difference $\Gamma_*$ of the trace with each reduced template as follows:

$$\Gamma_{\mathrm{O}} = (t'_{\mathcal{P}} - m_{\mathrm{O},\mathcal{P}})^T \cdot (t'_{\mathcal{P}} - m_{\mathrm{O},\mathcal{P}})$$
$$\Gamma_{\mathrm{X}} = (t'_{\mathcal{P}} - m_{\mathrm{X},\mathcal{P}})^T \cdot (t'_{\mathcal{P}} - m_{\mathrm{X},\mathcal{P}})$$

The attacker then chooses that class corresponding to the least sum-of-squared difference. Thus, a single power/EM trace of the targeted operation is enough to distinguish between

[3]Codewords/messages are bit vectors in $\{0,1\}^*$, but we deal with codewords/messages whose all but one bit at the LSB are always zero. Thus for simplicity, we refer to them using integer values.

[4]The chosen threshold $Th_{sel}$ shall be preferably slightly greater than $\pm 4.5$ for better distinguishability. For the $t$-test results in Fig.2, we choose a convenient threshold of $\pm 7$.

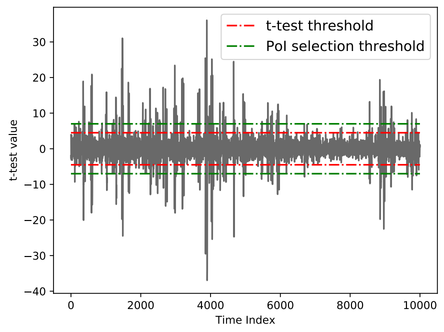


**Figure 2:** TVLA results for Round5 (R5ND_1KEM_5d) between decoding of valid ($c = 0$) and invalid ($c = 1$) codeword

the two cases $c = 0$ and $c = 1$. This can thus be used to instantiate a practical *plaintext checking* oracle to perform our chosen ciphertext attacks.

It is important to know that the *pre-processing* stage involving construction of *reduced templates* can be done directly on the target device in a *non-profiled* setting since the attacker does not require to gain information about the device for known secret keys. This is because an attacker can create chosen ciphertexts that can always decrypt to a specific codeword $c$ of his choice independent of the secret key, about which we will discuss in Sec.3.4.3. In the following section, we demonstrate practical chosen-ciphertext attacks over the IND-CCA secure Round5 PKE/KEM by exploiting the aforementioned side-channel vulnerability of XEf's decoding.

## 3.4 Chosen-Ciphertext Attack on IND-CCA secure Round5

### 3.4.1 Attacker Model

We consider an attacker who has physical access to a device performing decapsulation/decryption using a long term secret key. He/She can query the device with chosen ciphertexts arbitrarily many number of times and passively observe its corresponding side-channel behaviour and has no knowledge about corresponding outputs. He/She does not assume presence of a clone device and does not need to profile the device with known keys.

### 3.4.2 Decryption in Round5

Refer to Alg.2 for the decryption operation of the Round5 PKE scheme. We ignore several technical details in its description for simplicity and only focus on those relevant for our attack. The reader is however referred to [BBF+19] for more in-depth details. Since our attack only deals with decryption, we do not describe its corresponding encryption and key generation procedures. Round5 is based on the RLWR problem and hence both $\mathbf{u}$ and $\mathbf{v}$ are polynomials with degree $n-1$ and $\mu-1$ resp. in the polynomial ring $\mathbb{Z}_q(x)/\gamma(x)$ where $\gamma(x) = x^n + x^{n-1} + \ldots + x + 1$, with $n > \mu$. Furthermore, $\mathbf{s}$ is a ternary polynomial with degree $n-1$ with coefficients in $\{-1, 0, 1\}$. The decryption firstly computes the product $\mathbf{u} \times \mathbf{s}$ (Line 4 in Alg.2) modulo the cyclotomic polynomial $\phi(x) = (x^{n+1} - 1)$. Further, the product is truncated to the lowermost $\mu$ coefficients and subtracted from a rounded version of $\mathbf{v}$ (Line 5 in Round5.Decrypt). The result of this operation in $\mathbb{Z}_p^\mu$ is converted into the codeword $c'' \in \mathbb{Z}_2^\mu$ using $\mathsf{Round}_{p\rightarrow 2}$ and then it is decoded to the final message $m''$ using the Ecc_Dec decoding of XEf.

**Algorithm 2:** Round5 CPA.PKE scheme

```
1 Procedure Round5.Decrypt($ct, sk$)
2     $\mathbf{u}, \mathbf{v}$ = DecodeCT($ct$)
3     $\mathbf{s}$ = DecodeSK($sk$)
4     $\mathbf{w}$ = Truncate$_\mu$($\mathbf{u} \times \mathbf{s} \bmod (x^{n+1} - 1)$)
5     $c''$ = Round$_{p\to 2}$($(p/t) \cdot \mathbf{v} - \mathbf{w}, 1/2$)
```

### 3.4.3 Attack Methodology

Overview: Our attack recovers the secret polynomial one coefficient at a time and let us assume $X$ number of candidates for each coefficient (i.e) $(\mathbf{s}_0, \mathbf{s}_1, \ldots, \mathbf{s}_{X-1})$. Firstly, the attacker chooses a specific structure for the ciphertexts such that the value of decryption output (1) is solely dependent upon a single secret coefficient and (2) can only take two possible values (e.g) $c = 0$ or $c = 1$. Subsequently, he/she devises some *constraints* over the value of these specially structured ciphertexts such that their corresponding decryption outputs (oracle's responses) uniquely identify every possible candidate $\mathbf{s}_i$ for the target coefficient. More concretely, for every candidate $\mathbf{s}_i$, the oracle's responses for $M$ such chosen ciphertexts yield a binary sequence $\mathcal{S}_i = [b_0, b_1, b_2, \ldots, b_{M-1}]$. If these ciphertexts yield a unique sequence $\mathcal{S}_i \ \forall\, \mathbf{s}_i$, then the correct candidate can be identified based on the oracle's responses. The decapsulation device is triggered with these chosen ciphertexts and their corresponding EM side-channel observations are captured. Since information derived from these EM observations can serve as oracle's responses, the attacker can uniquely identify the targeted secret coefficient. This procedure can be repeated for other coefficients for full key recovery. In fact, we utilize the same approach albeit with minor modifications to target multiple LWE/LWR based schemes in this paper.

We first discuss our technique to recover a single coefficient of $\mathbf{s}$ and further extend it to the general case for recovery of any coefficient. Let us consider chosen ciphertexts where both $\mathbf{u}$ and $\mathbf{v}$ are constants with values $k_{\mathbf{u}}$ and $k_{\mathbf{v}}$ respectively (i.e) $\mathbf{u} = k_{\mathbf{u}} \cdot x^0$ and $\mathbf{v} = k_{\mathbf{v}} \cdot x^0$. Thus, the product $\mathbf{w} = (\mathbf{u} \times \mathbf{s}) \bmod \phi(x)$ is $\mathsf{Rotr}(k_{\mathbf{u}} \cdot \mathbf{s}, -1)$. Please note that the left rotation by one position is because degree$(\phi(x)) \neq$ degree$(\gamma(x))$ and they differ by one. The first $\mu$ coefficients of $\mathbf{w}$ are used to compute the corresponding bits $c''_i$ of the codeword $c''$ and they can be represented as in Eqn.4.

$$c''_i = \begin{cases} \mathsf{Round}_{p\to 2}((p/t) \cdot k_{\mathbf{v}} - k_{\mathbf{u}} \cdot \mathsf{s}[1], 1/2), & \text{if } i = 0 \\ \mathsf{Round}_{p\to 2}(-1 \cdot k_{\mathbf{u}} \cdot \mathsf{s}[i+1], 1/2), & \text{for } 1 \le i \le \mu - 1 \end{cases} \tag{4}$$

We can see that every bit $c''_i$ only depends on a single coefficient of the secret $\mathbf{s}$ and while the first bit depends upon both $k_{\mathbf{u}}$ and $k_{\mathbf{v}}$, all other bits of $c''$ only depend on $k_{\mathbf{u}}$. We now choose the value for ordered pairs $(k_{\mathbf{u}}, k_{\mathbf{v}})$ based on the constraints given in Eqn.5. Given an ordered pair $(k_{\mathbf{u}}, k_{\mathbf{v}})$,

$$c''_i = \begin{cases} \mathcal{D}(\mathbf{s}[1]), & \text{if } i = 0 \\ 0, & \text{for } 1 \le i \le \mu - 1 \end{cases} \tag{5}$$

where $\mathcal{D}$ is used to denote the function used to compute the codeword $c''$. First, we can search for values of $k_{\mathbf{u}}$ based on Eqn.4 such that, $c''_i = 0 \ \forall\, i \neq 0$ , irrespective of the value of the corresponding secret coefficient $\mathbf{s}[i+1]$. Thus for a chosen ($k_{\mathbf{u}}$, $k_{\mathbf{v}}$), the codeword $c''$ now only depends upon a single coefficient (i.e) $\mathsf{s}[1]$ and is only limited to two values 0 or 1. Subsequently, the attacker can search for several ordered pairs of ($k_{\mathbf{u}}$, $k_{\mathbf{v}}$) based on Eqn.4 such that the codeword $c''$ (0 or 1) uniquely identifies every candidate for $\mathbf{s}[1]$. Thus, given several values of $c''$ for such chosen ciphertexts, an attacker can easily identify $\mathbf{s}[1]$. As for the *pre-processing* phase, the attacker needs to collect traces of decoding procedures for codewords $c'' = 0$ and $c'' = 1$, without the knowledge of the secret key. The decrypted

**Table 1:** Unique distinguishability of every key candidate based on the validity of the codeword for the chosen values of $(k_{\mathbf{u}}, k_{\mathbf{v}})$ for our attack on the R5ND_1KEM_5d variant of IND-CCA secure Round5 KEM. **O** and **X** refer to valid $(c'' = 0)$ and invalid codewords $(c'' = 1)$ respectively.

| **Secret Coeff.** | $c'' = 0$ (**O**) $/c'' = 1$ (**X**) | |
|---|---|---|
| | $(k_{\mathbf{u}}, k_{\mathbf{v}})$ | |
| | (21,3) | (12,1) |
| -1 | **X** | **X** |
| 0 | **X** | **O** |
| 1 | **O** | **O** |

codeword can be made independent of the secret key $\mathbf{s}$ by simply choosing $k_{\mathbf{u}} = 0$ (i.e) $\mathbf{u} = 0$ and hence $\mathbf{u} \cdot \mathbf{s} = 0$. An appropriate value for $k_{\mathbf{v}}$ is chosen based on the value of the desired codeword. In fact, by choosing $\mathbf{u} = 0$, an attacker can decrypt to any chosen codeword by choosing an appropriate values for $\mathbf{v}$.

Tab.1, which we denote as the decision table $\mathbf{D}$, contains chosen values of $(k_{\mathbf{u}}, k_{\mathbf{v}})$ for our attack along with the corresponding unique sequence $\mathcal{S}$ for every secret candidate for the R5ND_1KEM_5d variant of the Round5 IND-CCA secure KEM. We can see that just *two* chosen ciphertexts are enough to identify the value of the secret coefficient. Since the validity of a given codeword can be determined from a single EM side-channel trace as shown in Sec.3.3, the attacker can retrieve the secret coefficient $\mathbf{s}[1]$ with just 2 traces. In order to retrieve the other coefficients, we leverage upon the property of multiplication modulo the cyclotomic polynomial $\phi(x) = x^{n+1} - 1$. We know that multiplication of a polynomial $\mathbf{s}$ by $x^i$ modulo $\phi(x)$ in the ring $\mathbb{Z}_q(x)/(\gamma(x))$ results in[5] $\mathsf{Rotr}(\mathbf{s}, i-1)$. Thus if $\mathbf{u} = k_{\mathbf{u}} \cdot x^p$, the first bit of the codeword $c''_0$ is now function of a different secret coefficient as shown in Eqn.6

$$c''_0 = \begin{cases} \mathsf{Round}_{p\to 2}(-1 \cdot k_{\mathbf{u}} \cdot \mathbf{s}[1-p], 1/2), & \text{if } 0 \leq p \leq 1 \\ \mathsf{Round}_{p\to 2}(-1 \cdot k_{\mathbf{u}} \cdot \mathbf{s}[n-p+2], 1/2), & \text{if } 2 \leq p \leq n-1 \end{cases} \tag{6}$$

Thus, the attacker can change the position of the non-zero coefficient of $\mathbf{u}$ from 0 to $n-1$ and retrieve the coefficients $(\mathbf{s}[1], \mathbf{s}[0], \mathbf{s}[n] = 0, \mathbf{s}[n-1], \ldots, \mathbf{s}[3])$ in the same order. A close observation of Eqn.6 reveals that the secret polynomial cannot be rotated enough to bring $\mathbf{s}[2]$ to the first position, thus it is not possible to retrieve $\mathbf{s}[2]$ using the same methodology, with the reason again being degree($\gamma(x)$) - degree($\phi(x)$) = 1. However, as we show later, such complications do not arise in the other schemes we cover in this paper. Nevertheless, the attacker only requires $2 \times (n-1)$ side-channel traces (recovery of $(n-1)$ coefficients) and a brute-force over $\mathbf{s}[2]$ for full key-recovery. This attack is applicable to all parameters of IND-CCA secure Round5 PKE and KEM that use ECC after decryption. Since our attack targets the output of the decryption, our attack applies to both IND-CCA secure PKE and KEM and this also applies to all schemes covered in this paper.

We summarize our attack methodology in form of an algorithm in Alg.3, while abstracting all the low-level information of the scheme under attack. The algorithm takes as inputs, the chosen ciphertext set $\mathsf{CT}_{\mathsf{attack}}$ with $r$ chosen ciphertexts ($r = 2$ for Round5) and the information gathered from the *pre-processing* stage $\mathcal{I} = (\mathcal{P}, m_{\mathrm{O},\mathcal{P}}, m_{\mathrm{X},\mathcal{P}})$ The function Classify classifies a given trace $t$ into either class $c = 0$ or $c = 1$ using $\mathcal{I}$. The function RowCompare compares the results of the assignment $(B)$ with the decision table $\mathbf{D}$ to retrieve the corresponding secret coefficient.

[5]The rotation caused by $x^i$ is typically $i$ when degree($\gamma(x)$) = degree($\phi(x)$), but the factor of $i-1$ is due to the fact that degree($\gamma(x)$)-degree($\phi(x)$) = 1

**Algorithm 3:** SCA-Assisted Chosen Ciphertext Attack

```
Input  : Ciphertext Set: $\mathsf{CT}_{\mathsf{attack}} = \{ct_0, ct_1, \ldots, ct_{r-1}\}$, Pre-processing information:
         $\mathcal{I} = (\mathcal{P}, m_{\mathrm{O},\mathcal{P}}, m_{\mathrm{X},\mathcal{P}})$, Decision Table: $\mathbf{D}$
Output : Retrieved Secret: $\mathbf{s}' = \{\mathbf{s}'[0], \mathbf{s}'[1], \ldots, \mathbf{s}'[n-1]\}$

1 $B = \mathsf{Zeros}(r)$                                  /* Initialize $B$ */
2 for $i = 0$ to $n-1$ do
3     for $j = 0$ to $r-1$ do
4         $t_{i,j} \Longleftarrow \mathsf{Decaps}(ct_j)$            /* Trace Acquisition */
5         $B[j] = \mathsf{Classify}(t_{i,j}, \mathcal{I})$          /* Classify $c = 0$ or $c = 1$ */
6     end
7     $\mathbf{s}'[i] = \mathsf{RowCompare}(B, \mathbf{D})$         /* Compare with Decision Table */
8 end
```

## 3.5 Experimental Results

We implemented our attack on the optimized implementation of the R5ND_1KEM_5d variant of the IND-CCA secure Round5 KEM taken from the *pqm4* library [KRSS]. The attack requires about $2 \cdot 489 = 978$ traces (no. of coefficients $n = 490$) and brute-force of 3 for the coefficient $\mathbf{s}[2]$. Our attack works with an average success rate of about 99%. However, we were able to perform complete key recovery with high confidence by repeating the attack multiple times (a minimum of 3 times) and performing a simple majority voting of the retrieved coefficients. It actually only took about 95 seconds (including 10 seconds for *pre-processing*) to perform one complete iteration of the attack and 270 seconds (3 iterations) (i.e) 4.5 minutes for complete key recovery. This thus demonstrates the simplicity and effectiveness of our attack, which can be performed in a similar manner over the other parameter sets of Round5 PKE/KEM that utilize ECC.

## 4 Side-Channel Analysis of LAC

In this section, we adapt our chosen ciphertext attack to the IND-CCA secure LAC KEM which utilizes the heavy BCH error correcting code. While D'Anvers *et al.* [DTVV19] exploited the timing-side channel vulnerabilities in the non-constant time BCH decoding procedure, we exploit EM-side channel vulnerabilities over its constant-time variant. LAC is an RLWE based scheme operating in the polynomial ring $\mathbb{Z}_q[x]/(x^n + 1)$, with a byte level modulus $q$ and ternary secrets and errors with fixed hamming weight. We will first discuss the side-channel vulnerability of the BCH error correcting code followed by the practical demonstration of our chosen-ciphertext attack for full key-recovery.

### 4.1 Side-Channel Vulnerability of BCH

For brevity, we do not cover details of the BCH error correcting code, but refer the reader to [WR19] for in-depth details. We only focus on its decoding procedure, which contains the target operation for our attack. The decoding of BCH mainly works by computing a syndrome $S$ of the received codeword and utilizing $S$ to find error locations within the codeword. It is well known that the syndrome is zero for a valid codeword while it is non-zero even for erroneous codewords with single bit errors. We thus hypothesize that EM/power side-channel information from the syndrome computation might enable to distinguish between valid and invalid codewords. Similar to the XEf error-correcting code, an all zero codeword (i.e) $c = 0$ is a valid BCH codeword while $c = 1$ is an erroneous codeword with a single bit error. We attempt to utilize EM side-channel information from the syndrome computation of BCH ECC to distinguish between $c = 0$ and $c = 1$.

We implemented the constant-time reference implementation of the IND-CCA secure LAC128 on the same DUT and utilized the same setup for EM trace acquisition as described in Sec.3.2. We utilize the Welch's $t$-test based reduced template approach to classify a given trace either into class $c = 0$ or $c = 1$. Please refer to Fig.3 for the Welch's $t$-test

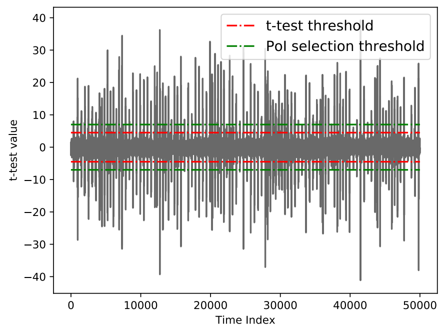


**Figure 3:** TVLA results for LAC (LAC128) between decoding of valid ($c = 0$) and invalid ($c = 1$) codeword

results between two trace sets $\mathcal{T}_{\mathrm{O}}$ and $\mathcal{T}_{\mathrm{X}}$ (with 50 traces each) corresponding to the syndrome computation for the codewords $c = 0$ and $c = 1$ respectively. A number of high peaks (well above $\pm 4.5$) indicate a clear difference between the two measurements and we choose a threshold of $\pm 12$ to construct reduced templates for each class using the PoI set $\mathcal{P}$. In the following section, we will demonstrate a chosen-ciphertext attack over IND-CCA secure LAC instantiating the identified leakage as an efficient *plaintext checking* oracle.

## 4.2 Chosen-Ciphertext Attack on IND-CCA secure LAC

Refer to Alg.4 for the decryption of LAC PKE. The ciphertext of LAC consists of two polynomials $\mathbf{c}_1, \mathbf{c}_2 \in \mathbb{Z}_q(x)/(x^n+1)$ with degree $n-1$ (same as the secret polynomial $\mathbf{s}$) and $\ell_v - 1$ respectively, with $n > \ell_v$ for all parameter sets of LAC. Firstly, the product of $\mathbf{c}_1$ and $\mathbf{s}$ in the ring $\mathbb{Z}_q[x]/(x^n+1)$ is truncated to its lowermost $\ell_v$ coefficients. The truncated product is then subtracted from $\mathbf{c}_2$ resulting in $\mathbf{x}$. In case of the LAC128/LAC192 variant, each bit $c''_i$ of the codeword is only dependent on the corresponding coefficient $\mathbf{x}[i]$ for $i \in \{0, \ell_v - 1\}$. But in case of LAC256, redundancy is introduced to lower error rates by utilizing two coefficients of $\mathbf{x}$ to compute one bit of the codeword (i.e) $\mathbf{x}[i]$ and $\mathbf{x}[i + \ell_v/2]$ are together used to compute the bit $c''_i$. This is referred to as the D2 encoding/decoding [LLZ+18], which is a light error correcting procedure implemented on top of the BCH error correcting code for the LAC256 variant. The codeword $c''$ is decoded to the message $m''$ by Ecc_Dec, the decoding of BCH.

**Algorithm 4:** LAC CPA.PKE scheme

```
Procedure LAC.Decrypt($ct = (\mathbf{c}_1, \mathbf{c}_2), sk = \mathbf{s}$)
    $\mathbf{w} = \mathsf{Truncate}_{\ell_v}(\mathbf{c}_1 \cdot \mathbf{s})$;
    $\mathbf{x} = \mathbf{c}_2 - \mathbf{w}$;
    $c'' = 0$;
    if LAC256 then
        for $i = 0$ to $\ell_v/2 - 1$ do
            $t = 2Q - (\mathbf{x}[i] + \mathbf{x}[i + \ell_v/2])$;
            if $t < (Q/2)$ then
                $c''_i = 1$;
            end
        end
    end
    else
        for $i = 0$ to $\ell_v - 1$ do
            if $\mathbf{x}[i] \geq (Q/4)$ and $\mathbf{x}[i] < (3Q/4)$ then
                $c''_i = 1$;
            end
        end
    end
```

### 4.2.1 Attack Methodology

Our attack methodology for LAC is very similar to that used for Round5. Due to the use of the additional D2 encoding/decoding in LAC256, our attack approach towards LAC256 is slightly different from that over the LAC128/LAC192 variant.

#### Attack on LAC128 and LAC192

We choose $\mathbf{c}_1 = k_{\mathbf{c}_1}$ and $\mathbf{c}_2 = k_{\mathbf{c}_2}$ respectively. Thus, any given bit $c''_i$ of the decrypted codeword $c''$ can be represented as in Eqn.7.

$$c''_i = \begin{cases} \mathcal{D}(k_{\mathbf{c}_2} - k_{\mathbf{c}_1} \cdot \mathbf{s}[0]), & \text{if } i = 0 \\ \mathcal{D}(-1 \cdot k_{\mathbf{c}_1} \cdot \mathbf{s}[i]), & \text{for } 1 \le i \le \ell_v - 1 \end{cases} \tag{7}$$

where $\mathcal{D}$ is used to denote the function used to compute the codeword $c''$. Similar to our attack on Round5, the attacker should subsequently choose enough number of ordered pairs $(k_{\mathbf{c}_1}, k_{\mathbf{c}_2})$ such that the corresponding ciphertexts result in (1) $c''_i = 0 \,\forall\, i \neq 0$ and (2) uniquely identification of every possible candidate for $\mathbf{s}[0]$ based on the corresponding values obtained for $c''_0$ or $c''$. Thus, as shown in Tab.2 in Appendix C, an attacker only needs 2 chosen ciphertext queries to identify the correct candidate for $\mathbf{s}[0]$ based on $c''$. Since he/she can easily distinguish between $c'' = 0$ and $c'' = 1$ with just a single EM/power trace as shown in Sec.4.1, just 2 traces are required to recover $\mathbf{s}[0]$.

In order to retrieve the other coefficients, we leverage upon the property of polynomial multiplication modulo the anti-cyclic cyclotomic polynomial $\phi(x) = x^n + 1$. We know that multiplication of a polynomial $\mathbf{s}$ by $x^i$ modulo $\phi(x)$ results in $\mathsf{AntiRotr}(\mathbf{s}, i)$. Thus, if we choose $\mathbf{c}_1 = k_{\mathbf{c}_1} \cdot x^p$, then the first bit $c''_i$ of the codeword $c''$ as a function of $p$ is

$$c''_0 = \begin{cases} k_{\mathbf{c}_2} - k_{\mathbf{c}_1} \cdot \mathbf{s}[0], & \text{if } p = 0 \\ k_{\mathbf{c}_2} - k_{\mathbf{c}_1} \cdot (-\mathbf{s}[n-p]), & \text{for } 1 \le p \le n-1 \end{cases} \tag{8}$$

Thus, the attacker can simply change the position of the non-zero coefficient of $\mathbf{c}_1$ from 0 to $n-1$ and use the same values for $(k_{\mathbf{c}_1}, k_{\mathbf{c}_2})$ to recover the coefficients $\mathbf{s}[0], -\mathbf{s}[n-1], -\mathbf{s}[n-2], \ldots, -\mathbf{s}[2], -\mathbf{s}[1]$ in the same order and polarity. We acknowledge that the chosen ciphertexts used by D'Anvers et al. [DTVV19] in their attack over IND-CCA secure LAC can also be used to mount a very similar attack in the EM side-channel setting.

#### Attack on LAC256

Attacking LAC256 is a bit more challenging due to the utilization of the additional D2 encoding/decoding before application of the BCH error correcting code. As can be seen from Line 7 in Alg.4, the D2 encoding ensures that each bit $c''_i$ of the codeword depends on two coefficients $\mathbf{x}[i]$ and $\mathbf{x}[i + \ell_v/2]$. Thus, if $\mathbf{c}_1 = k_{\mathbf{c}_1}$, then $c''_i$ depends on two secret coefficients $\mathbf{s}[i]$ and $\mathbf{s}[i + \ell_v/2]$. Thus, we propose to utilize a modified $\mathbf{c}_2$ which has two non-zero coefficients at positions 0 and $\ell_v/2$ (i.e) $\mathbf{c}_2 = k'_{\mathbf{c}_2} + k''_{\mathbf{c}_2} \cdot x^{\ell_v/2}$. Thus, any given bit $c''_i$ of the resulting codeword can be represented as follows:

$$c''_i = \begin{cases} \mathcal{V}(k'_{\mathbf{c}_2} - k_{\mathbf{c}_1} \cdot \mathbf{s}[0], k''_{\mathbf{c}_2} - k_{\mathbf{c}_1} \cdot \mathbf{s}[\ell_v/2]), & \text{if } i = 0 \\ \mathcal{V}(-1 \cdot k_{\mathbf{c}_1} \cdot \mathbf{s}[i], -1 \cdot k_{\mathbf{c}_1} \cdot \mathbf{s}[i + \ell_v/2]), & \text{for } 1 \le i \le \ell_v - 1 \end{cases} \tag{9}$$

Now, the attacker should choose values for the tuple $(k_{\mathbf{c}_1}, k'_{\mathbf{c}_2}, k''_{\mathbf{c}_2})$ that satisfy the constraints in Eqn.10 such that for any such chosen tuple $(k_{\mathbf{c}_1}, k'_{\mathbf{c}_2}, k''_{\mathbf{c}_2})$,

$$c''_i = \begin{cases} \mathcal{V}'(\mathbf{s}[0], \mathbf{s}[\ell_v/2]), & \text{if } i = 0 \\ 0, & \text{for } 1 \le i \le \ell_v - 1 \end{cases} \tag{10}$$

Here again, the possible values for the codeword is restricted to $c'' = 0$ and $c'' = 1$, but the value of the codeword now depends on two secret coefficients $\mathbf{s}[0]$ and $\mathbf{s}[\ell_v/2]$. Thus, as shown in Tab.3 in Appendix D, an attacker only needs five chosen ciphertexts and thus five EM traces to uniquely identify the value of the secret coefficient pair $(\mathbf{s}[0], \mathbf{s}[\ell_v/2])$.

For the retrieval of the other coefficient pairs, we again use the same technique used in the case of LAC128/LAC192 and choose $\mathbf{c}_1 = k_{\mathbf{c}_1} \cdot x^p$. Thus, the first bit of the codeword $c''$ as function of the position $p$ of the non-zero coefficient of $\mathbf{c}_1$ is given by Eqn.11. For the parameters of LAC256, we calculate that the attacker simply repeats the attack for about $(n - \ell_v/2)$ times to recover the complete secret polynomial $\mathbf{s}$.

$$c''_0 = \begin{cases} \mathcal{V}(k'_{\mathbf{c}_2} - k_{\mathbf{c}_1} \cdot \mathbf{s}[0], k''_{\mathbf{c}_2} - k_{\mathbf{c}_1} \cdot \mathbf{s}[\ell_v/2]), & \text{if } p = 0 \\ \mathcal{V}(k'_{\mathbf{c}_2} - k_{\mathbf{c}_1} \cdot (-\mathbf{s}[n-p]), k''_{\mathbf{c}_2} - k_{\mathbf{c}_1} \cdot \mathbf{s}[\ell_v/2 - p]), & \text{for } 1 \le p \le \ell_v/2 \\ \mathcal{V}(k'_{\mathbf{c}_2} - k_{\mathbf{c}_1} \cdot (-\mathbf{s}[n-p]), k''_{\mathbf{c}_2} - k_{\mathbf{c}_1} \cdot (-\mathbf{s}[n-p+\ell_v/2])), & \text{for } \ell_v/2 + 1 \le p < n \end{cases} \tag{11}$$

### 4.3 Experimental Results

We implemented our attack on the constant-time implementation of the LAC128 variant of the IND-CCA secure LAC KEM, taken from the *pqm4* library [KRSS]. The attack requires about $2 \cdot 512 = 1024$ traces (no. of coefficients $n = 512$) and works with an average success rate of ≈97%. A minimum of 3 repetitions of the attack and a subsequent majority voting resulted in complete key-recovery. Implementation of our complete attack (including trace-acquisition) for one iteration took about 525 seconds (8.75 minutes) and the complete key was recovered in 1490 seconds (3 repetitions in ≈ 25 minutes).

## 5 Side-Channel Analysis of the FO transform

We have shown that an attacker can efficiently utilize EM-side channel information from execution of error correcting procedures to instantiate an efficient *plaintext checking* oracle, leading to chosen ciphertext attacks over IND-CCA secure Round5 and LAC KEM. We also identify a similar vulnerability in the Fujisaki-Okamoto (FO) transform in schemes that do not utilize ECC, that enables an attacker to gain information about decrypted messages leading to similar chosen ciphertext attacks to perform full key recovery.

### 5.1 Side-Channel vulnerability in the FO transform

We refer to the KEM.Decaps procedure of Alg.1 for the generic framework of the IND-CCA secure decapsulation. When Ecc_Used = 0, the operations after PKE.Decrypt depend upon the decrypted message $m'$. If any of these operations exhibit any differential behaviour based on the value of the decrypted message $m'$, we hypothesize that this behaviour can also be observed over the EM side-channel. Similar to our attacks on Round5 and LAC, we would like to distinguish between the two cases (1) $m' = 0$ ($\mathbf{Z}$) and (2) $m' = 1$ ($\mathbf{O}$) through the EM side-channel. We target the hash operation $\mathcal{G}$ operating over $m'$ (Line 7 of KEM.Decaps in Alg.1). In particular, we observe side-channel traces from operations (somewhere) towards the end of the hash computation so that the diffusion of the modified bit induces enough differential behaviour for easy distinguishability.

We ran the optimized implementation of Kyber (KYBER512 variant) over the same DUT and used the same experimental setup described in Sec.3.2. We collect two sets of 50 replicated measurements denoted as $\mathcal{T}_\text{Z}$ and $\mathcal{T}_\text{O}$, corresponding to decrypted messages $m = 0$ and $m = 1$ respectively. As stated earlier in Sec.3.4.3, it is possible to create ciphertexts that decrypt to a chosen message without the knowledge of the secret key. Referring to the $t$-test plot between $\mathcal{T}_\text{Z}$ and $\mathcal{T}_\text{O}$ in Fig.4(a), we can clearly see many peaks

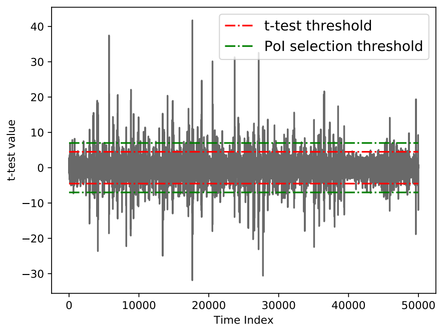


**Figure 4:** TVLA results for Kyber (KYBER512) between $\mathcal{G}(m) = 0$ and $\mathcal{G}(m) = 1$

(greater than the $\pm 4.5$), thus indicating a significant difference between the two sets of measurements at these peaks and we choose a threshold of $\pm 7$ for PoI selection and construct reduced templates for each class. Thus, an attacker can classify a single EM trace from the FO transform into class $m = 0$ and $m = 1$ to realize an efficient *plaintext checking oracle*. This leakage comes directly from the FO transform and hence applies to multiple lattice-based KEMs that utilize the same to achieve IND-CCA security.

## 5.2 Attack methodology

We demonstrate our attack on Kyber, an IND-CCA secure module lattice-based KEMs. The reason for the choice of module lattice-based schemes is that a module lattice can be transformed into a standard or an ideal lattice. Thus, our attack also be adapted to schemes based on problems on standard (Frodo) and ideal (NewHope) lattice problems.

Refer to Alg. 5 for the description of the decryption procedure in Kyber, an MLWE-based KEM. The ciphertext consists of $\mathbf{u} \in R_q^k$ and $\mathbf{v} \in R_q$ and the secret module $\mathbf{s} \in R_q^k$ with $R_q = \mathbb{Z}_q(x)/(x^n + 1)$. We will demonstrate our attack to retrieve one coefficient ($\mathbf{s}_0[0]$) and then extend it for complete recovery of the secret module. We choose $\mathbf{u}_0[0] = k_\mathbf{u}$ and $\mathbf{v}[0] = k_\mathbf{v}$ while all the other coefficients of $\mathbf{u}$ and $\mathbf{v}$ are set to zero. Thus, each bit of the decrypted message can be represented as in Eqn.12.

$$m'_j = \begin{cases} \mathsf{Poly\_to\_Msg}(k_\mathbf{v} - k_\mathbf{u} \cdot \mathbf{s}_0[0]), & \text{if } j = 0 \\ \mathsf{Poly\_to\_Msg}(-1 \cdot k_\mathbf{u} \cdot \mathbf{s}_0[j]), & \text{for } 1 \le j \le n-1 \end{cases} \quad (12)$$

We can see that the value of the decrypted message $m'$ only depends on one polynomial $\mathbf{s}_0$. Now, we further reduce the dependence of $m'$ to a single secret coefficient by choosing those values for $(k_\mathbf{u}, k_\mathbf{v})$ based on the constraints in Eqn.13.

$$m'_i = \begin{cases} \mathcal{D}(\mathbf{s}_0[0]), & \text{if } i = 0 \\ 0, & \text{for } 1 \le i \le n-1 \end{cases} \quad (13)$$

We thus ensure that for the selected ordered pairs $(k_\mathbf{u}, k_\mathbf{v})$, the value of the decrypted message ($m' = 0$ or $m' = 1$) solely depends on $\mathbf{s}_0[0]$. Thus, an attacker can collect enough number of such ciphertexts to uniquely identify the value of $\mathbf{s}_0[0]$ based on the value of the decrypted message $m'$. Refer to Tab.1 for the chosen values for $(k_\mathbf{u}, k_\mathbf{v})$ for our attack on the Kyber512 variant of the Kyber IND-CCA secure KEM. Since an attacker can classify a given trace as belonging to either $m' = 0$ or $m' = 1$, the secret coefficient $\mathbf{s}_0[0]$ can be retrieved with just 5 side-channel traces.

Similar to our attack on LAC, the attacker can sweep the position of the non-zero coefficient of $\mathbf{u}_0$ from 0 to $n-1$ (i.e) $\mathbf{u}_0[0], \ldots, \mathbf{u}_0[n-1]$, to recover the coefficients

**Algorithm 5:** Decryption in module lattice-based KEM

```
1 Procedure KEM.Decrypt($ct, sk$)
2     $(\mathbf{u}, \mathbf{v}) \in (R_q^k \times R_q) =$ DecodeCT($ct$);
3     $\mathbf{s} \in R_q^k =$ DecodeSK($sk$);
4     $\mathbf{w} \in R_q = \mathbf{u} \times \mathbf{s}$;
5     $m' =$ Poly_to_Msg($\mathbf{v} - \mathbf{w}$);
```

$\mathbf{s}_0[0], -\mathbf{s}_0[n-1], -\mathbf{s}_0[n-2], \ldots, -\mathbf{s}_0[2], -\mathbf{s}_0[1]$ in the same order and polarity. Once the attacker recovers polynomial $\mathbf{s}_0$, he/she can also recover the remaining $k-1$ polynomials of $\mathbf{s}$ by correspondingly changing the non-zero polynomial of $\mathbf{u}$ from $\mathbf{u}_1$ to $\mathbf{u}_{k-1}$. More concretely, if $\mathbf{u}[p] = k_{\mathbf{u}} \cdot x^q$, then the first bit of the message $m'_0$ is given as

$$m'_0 = \begin{cases} \text{Poly\_to\_Msg}(k_{\mathbf{v}} - k_{\mathbf{u}} \cdot (\mathbf{s}_p[0])), & \text{if } q = 0 \\ \text{Poly\_to\_Msg}(k_{\mathbf{v}} - k_{\mathbf{u}} \cdot (-\mathbf{s}_p[n-q])), & \text{for } 1 \leq q \leq n-1 \end{cases} \tag{14}$$

Thus, we can see that the attacker can retrieve all the $n \cdot k$ coefficients of the secret module $\mathbf{s}$ (using $5 \cdot n \cdot k$ traces) by simply repeating the attack for different values of the ordered pair $(p, q)$ with $p \in \{0, k-1\}$ and $q \in \{0, n-1\}$.

## 5.3 Experimental Results

We implemented our attack on the optimized implementation of the KYBER512 variant of the IND-CCA secure Kyber KEM from the *pqm4* library (Refer to Sec.3.2 for details on DUT and experimental setup). We were able to retrieve the complete secret in $5 \cdot 256 \cdot 2 = 2560$ traces ($n = 256, k = 2$) with an average success rate of about 99% and about 3 repetitions of the attack for full key-recovery. One complete iteration of the attack (including trace acquisition) required about 230 seconds. Thus, complete recovery of the secret key could be performed in about 650 seconds (3 iterations in 10.83 minutes), thus demonstrating the ease and effectiveness of our attack. The same attack can be adapted easily to the other parameter sets of the IND-CCA secure Kyber KEM.

## 5.4 Adapting our Attack Methodology to Other Schemes

In this section, we will briefly discuss about the adaptation of our attack methodology to other lattice-based schemes. Since Saber also deals with problems in module-lattices, it is structurally similar to Kyber and hence our attack methodology over Kyber directly applies to Saber as well. Moreover, our attack methodology for variants of Round5 that use ECC (described in Sec.3.4.3) also directly applies to the variants that do not use ECC. The only key difference is that side-channel information is retrieved from the computation of the hash function over the decrypted message in the latter case. Thus we do not discuss about the adaptability of our attacks on Saber and Round5, but focus on other structurally different schemes such as NewHope (RLWE) and Frodo (Standard-LWE).

### 5.4.1 Attacking NewHope

NewHope is based on the RLWE problem and operates in a very similar polynomial ring $\mathbb{Z}_q(x)/(x^n + 1)$ as LAC albeit with a different $n$ and $q$. Thus, we choose to adapt our attack methodology over LAC to work for NewHope. An important point to note is that both variants of NewHope utilize redundancy in the message encoding procedure to gain robustness against errors. While the NewHope512 variant encodes a single bit of the message in 2 coefficients similar to LAC256 (redundancy factor $Rf = 2$), the NewHope1024 variant encodes a single bit in four coefficients ($Rf = 4$). Thus, recovery of coefficients in case of NewHope512 is done in pairs (similar to LAC256), while it is done in quadruples for NewHope1024. Moreover, secrets are binomially distributed in $[-8, 8]$ for both variants and thus for NewHope512 with $Rf = 2$, the total key candidates for each pair is $17^2 = 289$.

In case of NewHope1024, the total key candidates for each quadruple is $17^4 = 83521$. Thus, the number of key candidates are significantly larger compared to that for Kyber (5), Round5 (3) and LAC (3). Both the span of the secret coefficients and the redundancy in encoding together have a significant impact on the attack complexity, tremendously increasing the number of ciphertexts required for unique identification. Thus, we adopt a two stage approach towards generating the chosen ciphertexts for our attack on both NewHope512 and NewHope1024.

In the first stage, we generate a set of $G$ chosen ciphertexts based on the same approach for earlier schemes and the aim is to obtain unique bit sequences $\mathcal{S}$ for as many secret candidate tuples as possible. Upon repeated experiments, we found that for a reasonably small $G$, there are still many disjoint clusters of candidates say $L$ (i.e) $(K_0, K_1, \dots, K_{L-1})$, each with varying number of candidates $(n_1, n_2, \dots, n_{N-1})$ that are not uniquely distinguishable. In the second stage, we consider a *pairwise* classification approach to uniquely identify the correct candidate within every cluster. For example, within the cluster $K_0$, we consider every possible pair of candidates and generate a ciphertext for each of them that distinguishes the candidates within a pair. It is well known that unique identification of a particular candidate within a group is possible if there exists a pairwise distinguisher for every possible pair of candidates [KU02]. This yields a total of $\binom{n_0}{2}$ ciphertexts for equally many number of pairs within the cluster $K_0$ and the same can be repeated for all clusters yielding a total of $\sum_{i=0}^{L-1}\binom{n_i}{2}$ additionally precomputed ciphertexts.

Let us assume a perfectly correct *plaintext checking* oracle. In the attack phase, the attacker also follows a two stage approach. Firstly, he/she queries the device with the $G$ initially chosen ciphertexts. It is possible that the secret candidate be retrieved directly from the first stage if the obtained binary sequence $\mathcal{S}$ from the oracle corresponds to a uniquely identifiable candidate. Otherwise, the sequence corresponds to a smaller set of candidates belonging to one of these clusters (lets assume $K_0$). Subsequently, unique identification of the correct candidate within the cluster $K_0$ can be done in a binary fashion in $\lceil log_2(n_0)\rceil$ stages using just $(n_0 - 1)$ ciphertexts (similar to a knock-out tournament involving $2^{n_0}$ players). The same process can be repeated to retrieve all the tuples of the secret key. There is a fine trade-off observed between the number of queries ($G$) used for the first stage and the number of queries required in the second stage denoted as $G'$, required to uniquely retrieve the coefficients of the secret key.

We performed concrete attack simulations over the reference implementations of NewHope512 and NewHope1024 submitted to round 2 of the NIST process. Our attacks yield a success rate of 100% success rate on both the variants in the presence of a perfectly correct *plaintext checking* oracle. Our chosen ciphertexts are generated randomly and hence the results might vary. However, we provide empirical numbers for the minimum query complexity obtained from several simulation runs. For NewHope512, we chose $G = 20$ and assuming a central binomial distribution for the coefficients, we estimated an average of $G' = 7.13$ extra queries for every secret coefficient, which amounts to a total of $27.13 \times 256 = 6945.28$ queries on average to recover the complete secret key. Our repeated attack simulations also confirmed the estimated query complexity for NewHope512. For NewHope1024, we chose $G = 100$ and obtained an estimated average of $G' = 4.1$ extra queries for every secret coefficient. This amounts to a total of 26624 queries for NewHope1024, which was also confirmed by our repeated attack simulations. However in the EM side-channel setting, one can observe multiple traces (minimum of $3$ *or* $5$) for replications of each query and decide each oracle response based on a majority vote.

Compared to prior works to attack NewHope [QCD19, BGRR19], we freely choose values for both ciphertext components $\mathbf{u}$ and $\mathbf{v}$ (with reference to Alg.5), while both the prior works [QCD19, BGRR19] fix the value of $\mathbf{u}$ and only analyze the oracle's responses for different chosen values for $\mathbf{v}$, thus naturally suffering from lesser degrees of freedom.

For comparison, Bauer *et al.* [BGRR19] report a success rate of 95% for full key recovery on NewHope1024 using 16,700 queries on average while brute forcing among $\approx 6 \times 10^7$ possibilities for the remaining coefficients in each secret key. Qin *et al.* [QCD19] however report a success rate of 96.9% for full key recovery using $\approx 870k$ queries on average.

### 5.4.2 Attacking Frodo

Frodo is based on the standard LWE problem and involves computation over elements in matrix/vector rings, unlike RLWE/MLWE-based schemes which mainly involve polynomial arithmetic. However, Frodo is also based on the same framework of the LPR encryption scheme and thus our attack can also be adapted to attack Frodo PKE/KEM. Please refer to Appendix F for a detailed explanation of our attack methodology on Frodo. We show that our attack can be performed in a similar manner albeit requiring to perform multiple *pre-processing* steps for full key-recovery. This is mainly due to the fact that multiplication in matrix and vector rings do not exhibit any property similar to cyclic or anti-cyclic rotations which we exploited for our attacks on RLWE/MLWE-based lattice schemes.

## 6 Countermeasures

Developing countermeasures for different side-channel attacks is a continuous research. We briefly talk about the countermeasures separately for both our attacks. Firstly, our attack targeting the constant-time ECC can be protected through incorporation of side-channel protected implementations of ECC procedures. One can consider masking the ECC procedures. However, efficient masking schemes for the ECC used in lattice-based cryptography are yet to be invented. It will be an interesting but a challenging research topic to design masking schemes for complicated strong ECC such as BCH.

Secondly, our attack exploiting the vulnerability of the FO transform in schemes that do not use ECC requires a generic countermeasure provided the generic nature of the attack. Here again, masking the decryption/decapsulation operation can definitely protect against our attack. There have been several works on lattice-based PKE/KEMs [RRVV15, RdCR+16] and in particular, Oder *et al.* [OSPG18] describe masking techniques for protecting the FO transform operations in the CCA setting. However, masking complete decryption/decapsulation has been shown to be costly in terms of performance and thus efficient masking strategies for CCA-secure LWE/LWR schemes is an interesting research direction that warrants immediate attention by the cryptographic research community.

## 7 Conclusion

We have demonstrated generic EM side-channel assisted chosen ciphertext attacks applicable to over *six* IND-CCA secure LWE/LWR-based PKE/KEMs currently competing in the second round of the NIST standardization process. We identified EM-side channel vulnerabilities within error correcting procedures used in some of these schemes and also in operations within the FO transformation, that leak information about the output of the decryption algorithm, which subsequently lead to full key-recovery. We perform experimental validation of our attack on implementations from the open-source *pqm4* library, implemented on the ARM Cortex-M4F microcontroller. All the demonstrated attacks are directly performed on the target device and could perform key-recovery in *minutes.* We thus stress the need for attention from the cryptographic community towards side-channel resistant implementations of error correcting codes along with efficient masking strategies to protect CCA-secure LWE/LWR-based PKE and KEMs. We intend to investigate susceptibility of NTRU-based schemes to similar side-channel attacks as future work.

## Acknowledgments

This research is supported in parts by the National Research Foundation, Singapore, under its National Cybersecurity Research & Development Programme / Cyber-Hardware Forensic & Assurance Evaluation R&D Programme (Award: NRF2018NCR-NCR009-0001). The work was done during the visit of second author to NTU. The visit and the work was partially funded by BMW Asia Pte Ltd (Award no. M4062657.B60.500000).

## References

[ADPS16] Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. Post-Quantum Key Exchange-A New Hope. In *USENIX Security Symposium*, pages 327–343, 2016.

[BBF+19] Hayo Baan, Sauvik Bhattacharya, Scott Fluhrer, Oscar Garcia-Morchon, Thijs Laarhoven, Ronald Rietman, Markku-Juhani O. Saarinen, Ludo Tolhuizen, and Zhenfei Zhang. Round5: Compact and fast post-quantum public-key encryption. Cryptology ePrint Archive, Report 2019/090, 2019. https://eprint.iacr.org/2019/090.

[BCD+16] Joppe Bos, Craig Costello, Léo Ducas, Ilya Mironov, Michael Naehrig, Valeria Nikolaenko, Ananth Raghunathan, and Douglas Stebila. Frodo: Take off the ring! practical, quantum-secure key exchange from LWE. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1006–1018. ACM, 2016.

[BCNS15] Joppe W Bos, Craig Costello, Michael Naehrig, and Douglas Stebila. Post-quantum key exchange for the TLS protocol from the ring learning with errors problem. In *Security and Privacy (SP), 2015 IEEE Symposium on*. IEEE, 2015.

[BDHD+19] Ciprian Băetu, F Betül Durak, Loïs Huguenin-Dumittan, Abdullah Talayhan, and Serge Vaudenay. Misuse attacks on post-quantum cryptosystems. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 747–776. Springer, 2019.

[BDK+17] Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé. Crystals – kyber: a cca-secure module-lattice-based kem. Cryptology ePrint Archive, Report 2017/634, 2017. https://eprint.iacr.org/2017/634.

[BDK+18] Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé. Crystals-kyber: a cca-secure module-lattice-based kem. In *2018 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 353–367. IEEE, 2018.

[BGRR19] Aurélie Bauer, Henri Gilbert, Guénaël Renault, and Mélissa Rossi. Assessment of the key-reuse resilience of newhope. In *Cryptographers' Track at the RSA Conference*, pages 272–292. Springer, 2019.

[BKS19] Leon Botros, Matthias J. Kannwischer, and Peter Schwabe. Memory-efficient high-speed implementation of kyber on cortex-m4. In *Progress in Cryptology - AFRICACRYPT 2019 - 11th International Conference on Cryptology in Africa, Rabat, Morocco, July 9-11, 2019, Proceedings*, pages 209–228, 2019.

[Ble98] Daniel Bleichenbacher. Chosen ciphertext attacks against protocols based on the rsa encryption standard pkcs# 1. In *Annual International Cryptology Conference*, pages 1–12. Springer, 1998.

[BP18] Leon Groot Bruinderink and Peter Pessl. Differential fault attacks on deterministic lattice signatures. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2018(3), 2018. https://eprint.iacr.org/2018/355.pdf.

[BPR12] Abhishek Banerjee, Chris Peikert, and Alon Rosen. Pseudorandom functions and lattices. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 719–737. Springer, 2012.

[Cac04] Christian Cachin. *Advances in Cryptology-EUROCRYPT 2004: International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004. Proceedings*, volume 3027. Springer Science & Business Media, 2004.

[DAS$^+$17] Jintai Ding, Saed Alsayigh, RV Saraswathy, Scott Fluhrer, and Xiaodong Lin. Leakage of signal function with reused keys in rlwe key exchange. In *2017 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, 2017.

[DCQ19] Jintai Ding, Chi Cheng, and Yue Qin. A simple key reuse attack on lwe and ring lwe encryption schemes as key encapsulation mechanisms (kems). *IACR Cryptology ePrint Archive*, 2019:271, 2019.

[DFR18] Jintai Ding, Scott Fluhrer, and Saraswathy Rv. Complete attack on rlwe key exchange with reused keys, without signal leakage. In *Australasian Conference on Information Security and Privacy*, pages 467–486. Springer, 2018.

[DKRV18] Jan-Pieter D'Anvers, Angshuman Karmakar, Sujoy Sinha Roy, and Frederik Vercauteren. Saber: Module-LWR based key exchange, CPA-secure encryption and CCA-secure KEM. In *International Conference on Cryptology in Africa*, pages 282–305. Springer, 2018.

[DTVV19] Jan-Pieter D'Anvers, Marcel Tiepelt, Frederik Vercauteren, and Ingrid Verbauwhede. Timing attacks on error correcting codes in post-quantum secure schemes. *IACR Cryptology ePrint Archive*, 2019:292, 2019.

[DVV18] Jan-Pieter D'Anvers, Frederik Vercauteren, and Ingrid Verbauwhede. On the impact of decryption failures on the security of LWE/LWR based schemes. *IACR Cryptology ePrint Archive*, 2018:1089, 2018.

[DXL12] Jintai Ding, Xiang Xie, and Xiaodong Lin. A simple provably secure key exchange scheme based on the learning with errors problem. *IACR Cryptology EPrint Archive*, 2012:688, 2012.

[Flu16] Scott R Fluhrer. Cryptanalysis of ring-LWE based key exchange with key share reuse. *IACR Cryptology ePrint Archive*, 2016:85, 2016.

[FO99] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In *Annual International Cryptology Conference*, pages 537–554. Springer, 1999.

[FPS18] Tim Fritzmann, Thomas Pöppelmann, and Johanna Sepulveda. Analysis of error-correcting codes for lattice-based key exchange. In *International Conference on Selected Areas in Cryptography*, pages 369–390. Springer, 2018.

[GGJR$^+$11] Benjamin Jun Gilbert Goodwill, Josh Jaffe, Pankaj Rohatgi, et al. A testing methodology for side-channel resistance validation. In *NIST non-invasive attack testing workshop*, volume 7, pages 115–136, 2011.

[GLRP06] Benedikt Gierlichs, Kerstin Lemke-Rust, and Christof Paar. Templates vs. stochastic methods. In *International Workshop on Cryptographic Hardware and Embedded Systems*, pages 15–29. Springer, 2006.

[HGS99] Chris Hall, Ian Goldberg, and Bruce Schneier. Reaction attacks against several public-key cryptosystem. In *International Conference on Information and Communications Security*, pages 2–12. Springer, 1999.

[HMO$^+$16] James Howe, Ciara Moore, Máire O'Neill, Francesco Regazzoni, Tim Güneysu, and Kevin Beeden. Lattice-based encryption over standard lattices in hardware. In *Proceedings of the 53rd Annual Design Automation Conference*, page 162. ACM, 2016.

[HP10] W Cary Huffman and Vera Pless. *Fundamentals of error-correcting codes*. Cambridge university press, 2010.

[KMRV18] Angshuman Karmakar, Jose M. Bermudo Mera, Sujoy Sinha Roy, and Ingrid Verbauwhede. Saber on ARM CCA-secure module lattice-based key encapsulation on ARM. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2018(3):243–266, 2018.

[KRSS] Matthias J. Kannwischer, Joost Rijneveld, Peter Schwabe, and Ko Stoffelen. PQM4: Post-quantum crypto library for the ARM Cortex-M4. https://github.com/mupq/pqm4.

[KU02] Boonserm Kijsirikul and Nitiwut Ussivakul. Multiclass support vector machines using adaptive directed acyclic graph. In *Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No. 02CH37290)*, volume 1, pages 980–985. IEEE, 2002.

[LLZ$^+$18] Xianhui Lu, Yamin Liu, Zhenfei Zhang, Dingding Jia, Haiyang Xue, Jingnan He, Bao Li, Kunpeng Wang, Zhe Liu, and Hao Yang. LAC: Practical ring-LWE based public-key encryption with byte-level modulus. *IACR Cryptology ePrint Archive*, 2018:1009, 2018.

[LPR10] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 1–23. Springer, 2010.

[LZZ18] Chao Liu, Zhongxiang Zheng, and Guangnan Zou. Key reuse attack on newhope key exchange protocol. In *International Conference on Information Security and Cryptology*, pages 163–176. Springer, 2018.

[MU10] Alfred Menezes and Berkant Ustaoglu. On reusing ephemeral keys in diffie-hellman key agreement protocols. *IJACT*, 2(2):154–158, 2010.

[NIS16] NIST. Submission requirements and evaluation criteria for the post-quantum cryptography standardization process. https://csrc.nist.gov/csrc/media/projects/post-quantum-cryptography/documents/call-for-proposals-final-dec-2016.pdf, 2016.

[OPG14] Tobias Oder, Thomas Pöppelmann, and Tim Güneysu. Beyond ECDSA and RSA: Lattice-based digital signatures on constrained devices. In *Proceedings of the 51st Annual Design Automation Conference*, pages 1–6. ACM, 2014.

[OSPG18] Tobias Oder, Tobias Schneider, Thomas Pöppelmann, and Tim Güneysu. Practical CCA2-secure and masked ring-LWE implementation. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2018(1):142–174, 2018.

[PDG14] Thomas Pöppelmann, Léo Ducas, and Tim Güneysu. Enhanced lattice-based signatures on reconfigurable hardware. In *International Workshop on Cryptographic Hardware and Embedded Systems*, pages 353–370. Springer, 2014.

[Pei14] Chris Peikert. Lattice cryptography for the internet. In *international workshop on post-quantum cryptography*, pages 197–219. Springer, 2014.

[Pes16] Peter Pessl. Analyzing the shuffling side-channel countermeasure for lattice-based signatures. In *International Conference on Cryptology in India*, pages 153–170. Springer, 2016.

[POG15] Thomas Pöppelmann, Tobias Oder, and Tim Güneysu. Speed records for ideal lattice-based cryptography on avr. *IACR Cryptology ePrint Archive*, 2015:382, 2015.

[PPM17] Robert Primas, Peter Pessl, and Stefan Mangard. Single-trace side-channel attacks on masked lattice-based encryption. In Wieland Fischer and Naofumi Homma, editors, *Cryptographic Hardware and Embedded Systems – CHES 2017*, pages 513–533, Cham, 2017. Springer International Publishing.

[QCD19] Yue Qin, Chi Cheng, and Jintai Ding. A complete and optimized key mismatch attack on nist candidate newhope. *IACR ePrint Archive*, page 435, 2019.

[Rat05] Dominic Rath. Openocd-open on-chip debugger, 2005.

[RdCR$^+$16] Oscar Reparaz, Ruan de Clercq, Sujoy Sinha Roy, Frederik Vercauteren, and Ingrid Verbauwhede. Additively homomorphic ring-LWE masking. In *Post-Quantum Cryptography - 7th International Workshop, PQCrypto 2016, Fukuoka, Japan, February 24-26, 2016, Proceedings*, pages 233–244, 2016.

[Reg09] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM (JACM)*, 56(6):34, 2009.

[RJJ$^+$18] Prasanna Ravi, Bernhard Jungk, Dirmanto Jap, Zakaria Najm, and Shivam Bhasin. Feature selection methods for non-profiled side-channel attacks on ecc. In *2018 IEEE 23rd International Conference on Digital Signal Processing (DSP)*, pages 1–5. IEEE, 2018.

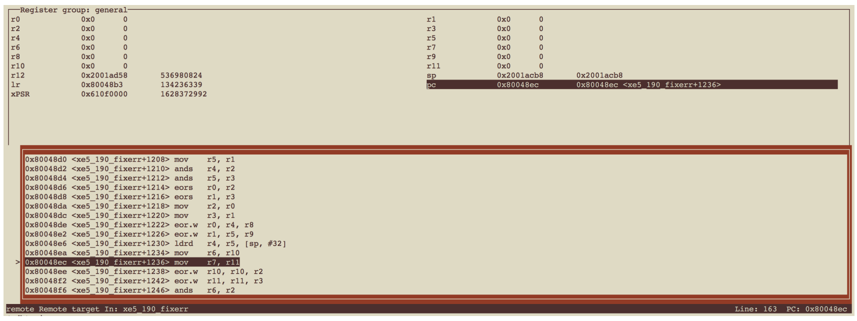
[RRVV15] Oscar Reparaz, Sujoy Sinha Roy, Frederik Vercauteren, and Ingrid Verbauwhede. A masked ring-LWE implementation. In *Cryptographic Hardware and Embedded Systems - CHES 2015 - 17th International Workshop, Saint-Malo, France, September 13-16, 2015, Proceedings*, pages 683–702, 2015.

[Sze17] A Szepieniec. Ramstake. Technical report, Technical report, National Institute of Standards and Technology, 2017.

[WR19] Matthew Walters and Sujoy Sinha Roy. Constant-time BCH error-correcting code. *IACR Cryptology ePrint Archive*, 2019:155, 2019.

## A Visualization of Activity in Registers

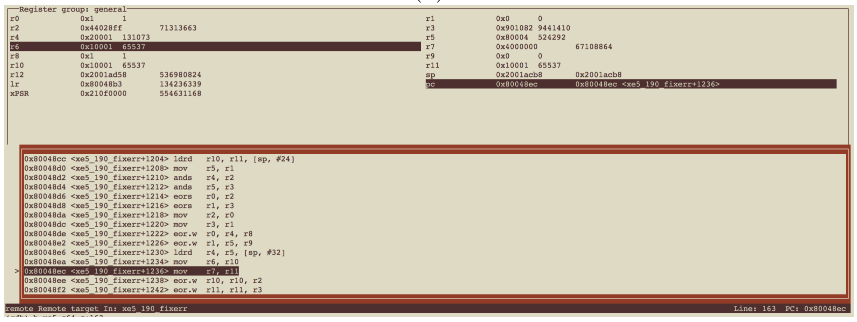
Please refer to Fig.5 for the screenshots from the Openocd Open On-Chip debugger tool [Rat05] showing values of the internal general purpose registers during operation of the targeted majority logic within the decoding procedure of XEf error correcting code of the Round5 PKE scheme. It is important to note that the values of the registers for the different cases are captured at the same point of time in the program (Refer to the program counter in the register set (highlighted in brown) in each figure). While Fig.5(a) corresponds to the decoding of a valid codeword (i.e) $c = 0$ and Fig.5(b) corresponds to the decoding of a invalid codeword $c = 1$. We can see that all the operating registers are zeros in case of the valid codeword (a), but is not the case with the invalid codewords (b). Since there is a significant difference between the data being operated on, based on the validity of the codeword, we are able to easily distinguish the same using the power/EM side-channel information.



```
Register group: general
r0             0x0      0                                  r1             0x0      0
r2             0x0      0                                  r3             0x0      0
r4             0x0      0                                  r5             0x0      0
r6             0x0      0                                  r7             0x0      0
r8             0x0      0                                  r9             0x0      0
r10            0x0      0                                  r11            0x0      0
r12            0x2001ad58        536980824                 sp             0x2001acb8        0x2001acb8
lr             0x80048b3         134236339                 pc             0x80048ec         0x80048ec <xe5_190_fixerr+1236>
xPSR           0x610f0000        1628372992

   0x80048d0 <xe5_190_fixerr+1208> mov     r5, r1
   0x80048d2 <xe5_190_fixerr+1210> ands    r4, r2
   0x80048d4 <xe5_190_fixerr+1212> ands    r5, r3
   0x80048d6 <xe5_190_fixerr+1214> eors    r0, r2
   0x80048d8 <xe5_190_fixerr+1216> eors    r1, r3
   0x80048da <xe5_190_fixerr+1218> mov     r2, r0
   0x80048dc <xe5_190_fixerr+1220> mov     r3, r1
   0x80048de <xe5_190_fixerr+1222> eor.w   r0, r4, r8
   0x80048e2 <xe5_190_fixerr+1226> eor.w   r1, r5, r9
   0x80048e6 <xe5_190_fixerr+1230> ldrd    r4, r5, [sp, #32]
   0x80048ea <xe5_190_fixerr+1234> mov     r6, r10
 > 0x80048ec <xe5_190_fixerr+1236> mov     r7, r11
   0x80048ee <xe5_190_fixerr+1238> eor.w   r10, r10, r2
   0x80048f2 <xe5_190_fixerr+1242> eor.w   r11, r11, r3
   0x80048f6 <xe5_190_fixerr+1246> ands    r6, r2
remote Remote target In: xe5_190_fixerr                                                            Line: 163   PC: 0x80048ec
```

(a)



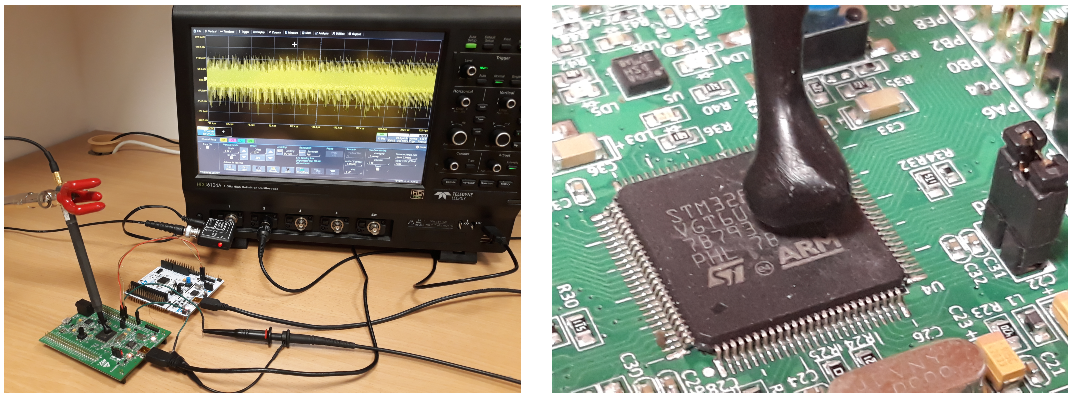
```
Register group: general
r0             0x1      1                                  r1             0x0      0
r2             0x44028ff         71313663                  r3             0x901082 9441410
r4             0x20001  131073                             r5             0x80004  524292
r6             0x10001  65537                              r7             0x4000000         67108864
r8             0x1      1                                  r9             0x0      0
r10            0x10001  65537                              r11            0x10001  65537
r12            0x2001ad58        536980824                 sp             0x2001acb8        0x2001acb8
lr             0x80048b3         134236339                 pc             0x80048ec         0x80048ec <xe5_190_fixerr+1236>
xPSR           0x210f0000        554631168

   0x80048cc <xe5_190_fixerr+1204> ldrd    r10, r11, [sp, #24]
   0x80048d0 <xe5_190_fixerr+1208> mov     r5, r1
   0x80048d2 <xe5_190_fixerr+1210> ands    r4, r2
   0x80048d4 <xe5_190_fixerr+1212> ands    r5, r3
   0x80048d6 <xe5_190_fixerr+1214> eors    r0, r2
   0x80048d8 <xe5_190_fixerr+1216> eors    r1, r3
   0x80048da <xe5_190_fixerr+1218> mov     r2, r0
   0x80048dc <xe5_190_fixerr+1220> mov     r3, r1
   0x80048de <xe5_190_fixerr+1222> eor.w   r0, r4, r8
   0x80048e2 <xe5_190_fixerr+1226> eor.w   r1, r5, r9
   0x80048e6 <xe5_190_fixerr+1230> ldrd    r4, r5, [sp, #32]
   0x80048ea <xe5_190_fixerr+1234> mov     r6, r10
 > 0x80048ec <xe5_190_fixerr+1236> mov     r7, r11
   0x80048ee <xe5_190_fixerr+1238> eor.w   r10, r10, r2
   0x80048f2 <xe5_190_fixerr+1242> eor.w   r11, r11, r3
remote Remote target In: xe5_190_fixerr                                                            Line: 163   PC: 0x80048ec
```

(b)

**Figure 5:** Visualization of internal registers during the targeted majority logic (a) Decoding of valid codeword $c = 0$ (b) Decoding of invalid codeword $c = 1$

## B Experimental Setup



(a) (b)

**Figure 6:** (a) Experimental Setup (b) Zoomed-in view of EM-probe over the DUT

## C Chosen Ciphertexts for Attack on LAC128

**Table 2:** Validity of the codeword as a function of the chosen values for the ordered pair $(k_{\mathbf{c}_1}, k_{\mathbf{c}_2})$ and the targeted secret coefficient, used for our attack on the LAC128 variant of LAC CCA-secure KEM. **O** and **X** refer to valid ($c'' = 0$) and invalid codewords ($c'' = 1$) respectively.

| **Secret Coeff.** | $c'' = 0$ (**O**) $/c'' = 1$ (**X**) | |
|---|---|---|
| | $(k_{\mathbf{c}_1}, k_{\mathbf{c}_2})$ | |
| | (5,192) | (14,48) |
| -1 | **O** | **X** |
| 0 | **O** | **X** |
| 1 | **X** | **O** |

## D Chosen Ciphertexts for Attack on LAC256

**Table 3:** Validity of the codeword as a function of the chosen values for the ordered tuple $(k_{\mathbf{c}_1}, k'_{\mathbf{c}_2}, k''_{\mathbf{c}_2})$ and the targeted secret coefficients, used for our attack on the LAC256 variant of LAC CCA-secure KEM. **O** and **X** refer to valid ($c''=0$) and invalid codewords ($c''=1$) respectively.

| **Secret Coeff. Pair** | $c''=0$ (**O**)/$c''=1$ (**X**) | | | | |
|---|---|---|---|---|---|
| | $(k_{\mathbf{c}_1}, k'_{\mathbf{c}_2}, k''_{\mathbf{c}_2})$ | | | | |
| | (207,64,240) | (207,64,32) | (207,64,64) | (207,64,112) | (207,64,208) |
| (-1,-1) | **O** | **O** | **O** | **O** | **O** |
| (-1,0) | **O** | **O** | **O** | **X** | **O** |
| (-1,1) | **O** | **O** | **X** | **O** | **O** |
| (0,-1) | **O** | **O** | **O** | **X** | **X** |
| (0,0) | **O** | **O** | **X** | **X** | **O** |
| (0,1) | **O** | **X** | **X** | **X** | **O** |
| (1,-1) | **X** | **O** | **X** | **X** | **X** |
| (1,0) | **O** | **X** | **X** | **X** | **X** |
| (1,1) | **X** | **X** | **X** | **X** | **O** |

## E Chosen Ciphertexts for Attack on Kyber512

**Table 4:** Value of the decrypted message $m'$ as a function of the chosen values for the ordered pair $(k_{\mathbf{u}}, k_{\mathbf{v}})$ and the targeted secret coefficient, used for our attack on the KYBER512 variant of Kyber CCA-secure KEM. While **O** refers to the case of $m'=0$, **X** refers to $m'=1$.

| **Secret Coeff.** | $m')=0$ (**O**)/$m'=1$ (**X**) | | | | |
|---|---|---|---|---|---|
| | $(k_{\mathbf{u}}, k_{\mathbf{v}})$ | | | | |
| | $(210,1)$ | $(210,7)$ | $(101,2)$ | $(100,6)$ | $(415,3)$ |
| -2 | **X** | **O** | **X** | **O** | **X** |
| -1 | **O** | **O** | **X** | **O** | **X** |
| 0 | **O** | **O** | **O** | **O** | **X** |
| 1 | **O** | **O** | **O** | **X** | **O** |
| 2 | **O** | **X** | **O** | **X** | **O** |

## F Attacking FO transform in Frodo KEM

Refer to Alg.6 for the decryption procedure of the Frodo PKE scheme. Since Frodo is based on the standard LWE problem, all computations within the scheme are performed over matrices/vectors with elements in the ring $\mathbb{Z}_q$, with the ciphertext matrices $\mathbf{U} \in \mathbb{Z}_q^{\bar{m}\times\bar{n}}$, $\mathbf{V} \in \mathbb{Z}_q^{n\times\bar{n}}$ and the secret matrix $\mathbf{S} \in \mathbb{Z}_q^{n\times\bar{n}}$. We will demonstrate our attack to retrieve one particular element ($\mathbf{S}[0][0]$) and then extend it to recover all the elements in the same column ($\mathbf{S}[0]$) and further extend it to the whole secret matrix $\mathbf{S}$. We assume that the attacker can successfully classify any given EM side-channel trace into either $m'=0$ or $m'=1$ (Refer to Sec.5.1).

**Algorithm 6:** Decryption operation in Frodo KEM

```
1 Procedure KEM.Decrypt($ct = (\mathbf{U}, \mathbf{V}) \in (\mathbb{Z}_q^{\bar{m}\times n}, \mathbb{Z}_q^{\bar{m}\times\bar{n}}), sk = \mathbf{S} \in \mathbb{Z}_q^{n\times\bar{n}}$)
2     $\mathbf{M} = \mathbf{V} - \mathbf{U} \times \mathbf{S}$;
3     $m' =$ Frodo.Decode($\mathbf{M}$);
```

We choose $\mathbf{U}[0][0] = k_{\mathbf{U}}$ while all the other elements of $\mathbf{U}$ are set to 0. Similarly we also choose $\mathbf{V}[0][0] = k_{\mathbf{V}}$ while all of its other elements are set to 0. Please note that the Frodo.Decode procedure (Line 2 of Alg.6) encodes every element of $\mathbf{M}$ to $B$ bits of the message $m'$. Thus, we utilize the notation $m'_{i,j}$ to denote $B$ bits of $m'$ starting from positions $(i \cdot \bar{n} + j) \cdot B$ to $(i \cdot \bar{n} + (j+1)) \cdot B - 1$. Thus, the resulting decrypted message $m'$ can be represented as in Eqn.15.

$$m'_{i,j} = \begin{cases} \text{Frodo.Decode}(k_{\mathbf{V}} - k_{\mathbf{U}} \cdot \mathbf{S}[0][0]), & \text{if } i = 0, j = 0 \\ \text{Frodo.Decode}(-1 \cdot k_{\mathbf{U}} \cdot \mathbf{S}[0][j]), & \text{if } i = 0, j \neq 0 \\ 0, & \text{otherwise} \end{cases} \tag{15}$$

Now, the attacker chooses those values for the ordered pair $(k_{\mathbf{U}}, k_{\mathbf{V}})$ based on the constraints in Eqn.16, such that given $(k_{\mathbf{U}}, k_{\mathbf{V}})$,

$$m'_{i,j} = \begin{cases} \mathcal{D}(\mathbf{S}[0][0]), & \text{if } i = 0, j = 0 \\ 0, & \text{otherwise} \end{cases} \tag{16}$$

We can thus see that the value of the decrypted message is solely dependent on the value of the secret coefficient $\mathbf{S}[0][0]$. Moreover, the ciphertexts can also be chosen to ensure that the decrypted message only takes two values $m' = 0$ or $m' = 1$. The attacker can collect enough number of such ciphertexts to uniquely identify the value of $\mathbf{S}[0][0]$ based on the value of the correspondingly decrypted messages. We also know that an attacker can easily classify a given trace into either $m' = 0$ or $m' = 1$. Thus, if the attacker requires $r$ ciphertexts to uniquely identify the value of $\mathbf{S}[0][0]$, then he/she would require $r$ traces to retrieve $\mathbf{S}[0][0]$.

The other elements of the first column of $\mathbf{S}$ (i.e) $\mathbf{S}[1][0], \ldots, \mathbf{S}[n-1][0]$ can be retrieved by simply changing the position of the non-zero element of $\mathbf{U}$ from $\mathbf{U}[0][1], \ldots, \mathbf{U}[0][n-1]$. More concretely, in order to recover $\mathbf{S}[p][0]$, we choose $\mathbf{U}[0][p] = k_{\mathbf{U}}$ with all other elements of $\mathbf{U}$ set to 0, while the value of $\mathbf{V}$ is unchanged (i.e) $\mathbf{V}[0][0] = k_{\mathbf{V}}$ and all its other elements set to zero. Thus, the resulting decrypted message can be represented as

$$m'_{i,j} = \begin{cases} \text{Frodo.Decode}(k_{\mathbf{V}} - k_{\mathbf{U}} \cdot \mathbf{S}[p][0]), & \text{if } i = 0, j = 0 \\ \text{Frodo.Decode}(-1 \cdot k_{\mathbf{U}} \cdot \mathbf{S}[p][j]), & \text{if } i = 0, j \neq 0 \\ 0, & \text{otherwise} \end{cases} \tag{17}$$

For the same previously chosen ordered pairs $(k_{\mathbf{U}}, k_{\mathbf{V}})$, the value of the resulting decrypted message will solely dependent on $\mathbf{S}[p][0]$. Thus, the attacker can similarly repeat the attack $n$ times to recover all the elements in the first column of $\mathbf{S}$ (i.e) $\mathbf{S}[0]$. However, there is a small caveat in repeating the attack to recover the other columns of $\mathbf{S}$. Let us consider our attack to retrieve the first coefficient of the $f^{th}$ column ($\mathbf{S}[0][f]$). We choose $\mathbf{U}[0][0] = k_{\mathbf{U}}$ and $\mathbf{V}[0][f] = k_{\mathbf{V}}$ while all the other elements of both $\mathbf{U}$ and $\mathbf{V}$ are set to zero. Now, the decrypted message is represented as in Eqn.18.

$$m'_{i,j} = \begin{cases} \text{Frodo.Decode}(k_{\mathbf{V}} - k_{\mathbf{U}} \cdot \mathbf{S}[0][f]), & \text{if } i = 0, j = f \\ \text{Frodo.Decode}(-1 \cdot k_{\mathbf{U}} \cdot \mathbf{S}[0][j]), & \text{if } i = 0, j \neq f \\ 0, & \text{otherwise} \end{cases} \tag{18}$$

For the same chosen ordered pairs $(k_{\mathbf{U}}, k_{\mathbf{V}})$, the decrypted message solely depends upon $\mathbf{S}[0][f]$ and can only take two possible values, $m' = 0$ and $m' = 1 \ll (Bf)$. We can thus see that we can no-longer restrict the location of the non-zero bit to the least significant position of $m'$. While we were able to restrict the location of the non-zero bit to the least significant position by leveraging the property of polynomial multiplication modulo cyclic (Round5) and anti-cyclic (LAC, Kyber, NewHope and Saber) cyclotomic polynomials, the same cannot done for Frodo as it operates over matrices and vectors. This means that the attacker should now distinguish between $m' = 0$ and $m' = 1 \ll (Bf)$. using side-channel information.

Thus, the attacker has to perform the *pre-processing* step to find new differentiating features (feature set $\mathcal{P}$) and a corresponding decision threshold $\overline{\mathbf{u}}$ that will enable to differentiate between $m' = 0$ and $m' = 1 \ll (Bf)$. The same has to be repeated for all $\bar{n}$ columns of $\mathbf{S}$ ($\bar{n} = 8$ for all parameter sets of Frodo). Thus, the attacker will require $\bar{n}$ feature sets ($\mathcal{P}_i$ for $i \in \{0, \bar{n} - 1\}$) and the corresponding $\bar{n}$ decision thresholds ($\overline{\mathbf{u}}_{\mathbf{i}}$ for $i \in \{0, \bar{n} - 1\}$) to recover all $\bar{n}$ columns of the secret matrix $\mathbf{S}$. Thus, the attacker can thus use the feature set $\mathcal{P}_f$ and decision threshold $\overline{\mathbf{u}}_{\mathbf{f}}$ to recover all the elements of the $f^{\text{th}}$ column of $\mathbf{S}$ and repeat the same for all $\bar{n}$ columns of $\mathbf{S}$ for full key-recovery.