Side-Channel Analysis of Kyber's Arithmetic Encodings: a Cautionary Note

No Author Given

No Institute Given

Abstract.

1 Introduction

It is not overstated to claim that the cryptography community is undergoing a major transition under the threat posed by quantum computing. Over the last few decades, building sufficiently large quantum computers has progressed from being physically impossible to being merely an engineer hurdle. As a result, it puts many of the public-key cryptosystems currently in use at risk, consequentially, compromises the confidentiality and integrity of digital communications. In order to prepare the information security systems to be able to resist quantum computing, in 2016 the USA National Institute of Standards and Technology (NIST), initialized a standardization process to promote quantum-resistant public-key cryptographic algorithms (also known as post-quantum cryptography, PQC).

After four rounds of evaluation, in 2022, NIST selected CRYSTALS-KYBER [ABD+17] as the winner of Public-key Encryption and Key-establishment Algorithms. KYBER belongs to lattice-based key encapsulation mechanism (KEM) family. Its hardness is based on module learning-with-errors problem (M-LWE) which is conjectured to be hard to solve even by sufficiently large quantum computers. Despite the concrete mathematical security of the algorithmic designs, KYBER has been shown to be vulnerable under implementation security perspective as more and more attacks from Side-channel Attack (SCA) family have been proven to be effective.

SCAs were first studied in the late 1990s by Kocher [Koc96], where it exploited the implementation execution time to significantly reduce the computational cost of the key-recovering attack. SCAs from then on have been spreading widely and various additional information such as power consumption, electromagnetic emanation, photoemission, . . . , are utilized to thwart the implementations' security.

Same as other lattice-based schemes, Kyber has been subjected to a variety of SCAs. These attacks are diverse in term of target procedure (e.g. Key Generation [PPM17], [PP19], [LZH+22], Encapsulation [ACLZ20], [SKL+20], [XPR+20], Decapsulation [NDGJ21], [MBM+22], [CKA+21], . . .), target operation within the procedure (e.g. Number Theoretic Transform (NTT), polynomial multiplication, Message Encode, KECCAK), attack vector (e.g. execution time, power consumption, Electromagnetic Emanation), etc. to name a few. Therefore, many studies have dedicated to improve side-channel resilience of Kyber's implementation by using countermeasures such as shuffling (e.g. [RPBC20], [ACLZ20]), or masking (e.g. [BDK+20], [OSPG18], [BC22]).

Masking on Kyber differs from masking for former symmetric or asymmetric key schemes. The computations happen in power-of-two and prime-order groups, thus it requires both Boolean and arithmetic masking at once as well as their two-way conversion. Both masking schemes have been carefully studied and yielded very concrete, provable security levels FiXme: cite Provable sec for Boolean masking and arithmetic masking.

Recently, arithmetic masking over prime-order group has been drawing attentions, thanks to its natural noise amplification effect [DFS16], [MMMS22]. However, some constraints over these formal results prevent a direct generalization for its effectiveness on Kyber, due to the special distribution of the secret. Therefore, in this note, we examine arithmetic masking particularly for Kyber, provide a atypical angle on its primary distinction from previously well-studied schemes, and so encourage further study on the subject.

We first give several notions and notations needed for the note in Section 2. Next, an analysis on simulated data is carried in Section 3. Finally, a quick evaluation proceeded on real measurements in Section 4 aims to verify the relevance of former results.

2 Background

Let us first lay out some notions regarding the arithmetic masking on Kyber.

2.1 Crystal-Kyber

Polynomial Arithmetic Kyber computations work on polynomial ring $R_q = \mathbb{Z}_q[X]/(X^n+1)$ where q is a prime that allow efficient Number Theory Transformation and n is the size of message space. A polynomial in such ring f has the form as:

$$f = f_0 + f_1 \cdot X^1 + \cdots + f_{n-1} \cdot X^{n-1} \tag{1}$$

where each coefficient f_i lies in $\mathbb{Z}_q$. The secret $\mathbf{s}$ in Kyber is a vector of k polynomials in R_q^k such that each polynomial $\mathbf{s}[i] \in R_q$.

Centered binomial distribution For the sake of simplicity we use upper cases to denote random variables and calligraphic letters to denote sets. A random variable X we denote $X \overset{\$}{\leftarrow} \mathcal{X}$ where X is uniform distributed over $\mathcal{X}$ and $X \leftarrow D$ when X is chosen according to the distribution D.

The noise (as well as secret) in Kyber is sampled from centered binomial distribution (CBD) β_η for $\eta \in \mathbb{N}$, $s \leftarrow \beta_\eta$ and is computed as:

$$(a_1, a_2, \ldots, a_\eta, b_1, b_2, \ldots, b_\eta) \overset{\$}{\leftarrow} \{0,1\}^{2\eta}$$
$$s = \sum_{i=1}^{\eta} (a_i - b_i)$$

The parameters set of Kyber for Round 3 of the NIST competition is summarized in below table: Since this note focus on encoding of polynomials we take n, q, η value as in the table but keep $k = 1$ for simplicity of the analysis.

Add spec table

Share Encoding Masking is a popular countermeasure against DPA that aims to randomize the intermediate values processed by the device, thus making the side-channel leakage independent of the sensitive values. The core idea is to probabilistically split the sensitive variables into shares, the computations process on shares and only combined at the end to get the correct output. Masking generally works for most cryptographic schemes, agnostic to the attack strategy, and guarantees a provable security.

In a d^{th}-order masked implementation, each intermediate variable is concealed by $d+1$ shares. The process of splitting a target value into $d+1$ *random* values is called share encoding. The encoding representation determines the relations among shares and its target value, thus determines how the computations on shares are combined in the end. We extend the definition in [PR13] in order to simplify the notations.

Definition 1 (d^{th}-order encoding). *Let $\mathcal{X}$ is a set in a group $(G, *)$ where $*$ is some group operation, let d be a positive integer. The d^{th}-order encoding of $x \in \mathcal{X}$ is a maps*

$$Enc^*_{d+1} : \mathcal{X} \to G^d :$$
$$x \mapsto (x_0, x_1, \ldots, x_d)$$

such that $(x_i)_{i=0}^{d-1} \overset{\$}{\leftarrow} G$ and $x = x_0 * x_2 * \cdots * x_d$

Difference masking schemes have different encoding representation, for example, in symmetric schemes where the working group is $\mathbb{Z}_{256}$, for ifrst-order Boolean masking $\text{Enc}_2^{\oplus}(X) = (X_1, X_2)$ where $X_1 \overset{\$}{\leftarrow} \mathbb{Z}_{256},\ X_2 = X \oplus X_1$ and for multiplicative masking, the encoding representation is in the form $\text{Enc}_2^{\otimes}(X) = (X_1, X_2)$ where $X_1 \overset{\$}{\leftarrow} \mathbb{Z}_{256},\ X_2 = X \otimes X_1$ or $X_2 = X \otimes X_1^{-1}$.

General masking scheme for KYBER

As mentioned earlier, the secret polynomial in Kyber can be express as a vector in $\mathbb{Z}_q$: $s = [s_0, s_1, \ldots, s_n],\ s_i \leftarrow \beta_\eta$, all available masking schemes for polynomial computation of Kyber FiXme: citations use additive arithmetic encoding, e.g.

$$\begin{aligned} Enc_2^{+}(s) &= (x_1, x_2) \\ x_1 &\xleftarrow{\$} \mathbb{Z}_q^n \\ x_2 &= s - x_1 \mod q \end{aligned}$$

For the rest of the note, we only study additive arithmetic encoding, thus, to simplify the notation we omit the operation specification of the encoding and keep d subscript to specify the number of shares.

Furthermore, since the coefficients of s are independent and the encoding acts on each coefficient independently, we only consider encoding for one coefficient (i.e. 1-D variable) instead of the full polynomial (i.e. 256-D variable).

Similar to multiplicative encoding in $\mathbb{Z}_{256}$, additive arithmetic encoding in $\mathbb{Z}_q$ can have different representation, e.g. for $\mathrm{Enc}_2(X) = (X_1, X_2)$ then there are two different ways to combine the two shares into protect variable X as:

$$\begin{aligned} X_2 &= (X - X_1) \mod q \qquad && \text{i.e.} X = (X1 + X2) \mod q \text{ or} \\ X_2 &= (X + X_1) \mod q \qquad && \text{i.e.} X = (X2 - X1) \mod q \end{aligned}$$

To distinguish different representations, as an abuse of notation, we denote:

$$\begin{aligned} &\mathrm{Enc}_d^{\mathrm{sum}}(X) = (X_1, X_2, \dots, X_d), \qquad && \mathrm{Enc}_d^{\mathrm{diff}}(X) = (X_1, X_2, \dots, X_d) \\ &X_d = X + \sum_{i=1}^{d-1} X_i \mod q \qquad && X_d = X - \sum_{i=1}^{d-1} X_i \mod q, \end{aligned}$$

where $X_i \xleftarrow{\$} \mathbb{Z}_q^n$, for $i \in \{1, \dots, d-1\}$.

SCA Metrics and Distinguishers

We next recall some SCAs' relevant notions.

Assumptions First, let X be the sensitive variable that is masked using the d-share encoding $\mathrm{Enc}_d(X) = (X_1, X_2, \dots, X_d)$, thus the device carries computations on $(X_1, X_2, \dots, X_d)$ and produces corresponding leakage vector $\boldsymbol{L}(X) = (L_1, L_2, \dots, L_d)$. , the leakage of each share can be written as $L(X_i) = \delta(X_i) + N_i$ where δ is a deterministic function of X_i and N_i denotes the random noise. The full leakage vector corresponding the process on X can be express as $\mathbf{L} = (L(X_1), L(X_2), \dots, L(X_d))$.

Independent Leakage and Independent Noise The independency of the leakage allows us to re-write the leakage corresponding to each variable X_i as $L_i = L(X_i) = \delta(X_i) + B$, where δ is a deterministic function of X_i and B denotes the random noise. This assumption infers that the leakage from the device depends only on the data being processed, and not on the difference of the data to another reference data.

Noise Independency indicates the random noise B in $L(X_i)$ is independent of the internal data X_i.

Gaussian Leakage and Gaussian Noise Gaussian leakage assumes that the distribution of the leakage L_i given the variable X_i follow is Gaussian, i.e. $(L_i|X_i = x) \leftarrow \mathcal{N}(\boldsymbol{m}_{i,x}, \boldsymbol{\Sigma}_{i,x})$, where $\boldsymbol{m}_{i,x}$ are expectation vectors and $\boldsymbol{\Sigma}_{i,x}$ are covariance matrices, both can be defined over high dimensional spaces.

Gaussian Noise assumes that the distribution of the random noise B follow a normal distribution with mean equals to zero and standard deviation σ, i.e. $B \leftarrow \mathcal{N}(0, \sigma^2)$, or in other word, $(L_i|X_i = x) \leftarrow \mathcal{N}(\delta(x), \sigma^2)$.

Signal-to-Noise Ratio Signal-to-Noise Ration (SNR) indicates the ratio between the signal and the noise component of a measurement and is widely used in electrical engineering and signal processing. In this note, we use SNR as a initial tool to detect relevant points that seems to carry useful information about target value in the measurements (i.e. Point of Interest (PoI)). SNR of variable X in leakage L is computed as

$$\mathrm{SNR} = \frac{Var_X[\mathbf{E}_L[L_x]]}{\mathbf{E}_X[Var_L[L_x]]}$$

Mutual Information and Perceived Information Information theoretic framework FiXme: IT citation is an usual tool to quantitatively analyze the worst-case security provided by a countermeasure. The mutual information (MI) measures the ‘amount of information’ obtained about one random variable (e.g. the targeting variable X) by observing the other random variable (e.g. the leakage L corresponding to the computation on X) and is computed by:

$$\mathrm{MI}(X; \mathbf{L}) = \sum_{x \in \mathcal{X}} p(x) \cdot \log_2 p(x) + \sum_{x \in \mathcal{X}} p(x) \int_{\boldsymbol{l} \in \mathcal{L}^d} f(\boldsymbol{l}|x) \cdot \log_2 p(x|\boldsymbol{l}). \tag{2}$$

The minuend is the self-entropy of X, $\mathrm{H}(X)$ and the subtrahend is the conditional entropy of X given L, $\mathrm{H}(X|L)$.

In Eq.2, $p(x)$ is the probability mass function (PMD) of X at the point x, $f(\boldsymbol{l}|x)$ the Probability Density Function (PDF) of the leakage for known value of $X = x$. and $p(x|\boldsymbol{l})$, followed the Bayes theorem, can be computed as

$$p(x|\boldsymbol{l}) = \frac{f(\boldsymbol{l}|x)p(x)}{\sum_{x' \in \mathcal{X}} f(\boldsymbol{l}|x')p(x')}. \tag{3}$$

The MI value between X and its leakage $\mathbf{L}$ directly links to the minimum number of measurements $N^\star$ that an adversary must obtained in order to recover specific value of X [DFS18], [CGPR19]:

$$N_a \geq \frac{c(\mathrm{sr}, |\mathcal{X}|)}{\mathrm{MI}(\mathbf{L}; X)},$$

where $c(\mathrm{sr}, |\mathcal{X}|)$ is a small constant depends on the success rate sr of the recovery attack, and size of the set $\mathcal{X}$.

Put MI by sampling here?

Precisely compute MI is not possible if $f(.|.)$, $p(.|.)$ are unknown priorly. Therefore, in practice, these quantities are estimated through a sampling process as $\hat{f}(.|.)$ and $\hat{p}(.|.)$. The perceived information (PI) allows to evaluate the quantity of such estimation. Generally state, PI quantifies the amount of information of X can be extracted from Y using an estimated model $\hat{p}$ of p. The PI is theoretically defined as:

$$PI(X;L) = \mathrm{H}(X) + \sum_{x\in\mathcal{X}} p(x) \int_{l\in\mathcal{L}} f(l|x) \cdot \log_2 \hat{p}(x|l)$$

and is practically computed as a sampling process as:

$$\widehat{PI}(X;L) = \mathrm{H}(X) + \sum_{x\in\mathcal{X}} p(x) \sum_{i=1}^{n_x} \frac{1}{n_x} \cdot \log_2 \hat{p}(x|l_{x,i}) \tag{4}$$

The sampling process that estimates $\widehat{PI}(X;L)$ needs to be carried on a separate set that used to estimate $\hat{p}$ to ensure $\widehat{PI}$ is unbiased. It has been shown in [BHM+19] that PI is upper bounded by MI and the equality holds if the model $\hat{p}$ is perfect.

It also has been pointed out in [BDMS22] that PI provides a mean to compare different model $\hat{p}$s via their *profiling complexity* and *online attack complexity*. Loosely speaking, profiling complexity is the number of samples n needed for an estimation $\hat{p}$ to reach a positive value and the complexity of the best online attack can be performed with a model is the asymptotic PI value that can possibly reach by that model (i.e. n is sufficiently high for the model to be optimal FiXme: wording) .

We now detail several notations that involves the process of estimating the model $\hat{p}(x|\boldsymbol{l})$.

Multivariate Gaussian Template Multivariate Gaussian Template (MGT) aims to model the distribution $\hat{f}(\boldsymbol{l}|x)$ using the Gaussian Leakage assumption, i.e.

$$\hat{f}(\boldsymbol{l}_i|x_i) = \frac{1}{\sqrt{(2\pi)^k \det \bar{\Sigma}_{i,x}}} \exp\left(-\frac{1}{2}(l_i - \bar{\boldsymbol{m}}_{i,x})^T \bar{\Sigma}_{i,x}^{-1}(l_i - \bar{\boldsymbol{m}}_{i,x})\right), \tag{5}$$

where $\bar{\boldsymbol{m}}_{i,x}$ and $\bar{\Sigma}_{i,x}$ are estimated means and covariance matrices resulted from profiling data set. During the profiling phase, a template (i.e. mean and covariance matrix) has to be build for each value from the set of all possible values. $\hat{p}(x|\boldsymbol{l})$ then can be estimated follows Eq.**??**.

Linear Discriminant Analysis Fisher's Linear Discriminant Analysis (LDA) is usually used in SCA as a pre-process technique, aims to reduce the dimension of the leakage traces. LDA is known to be optimal in terms of minimizing the Bayes error for binary classification under normality and homoscedasticity assumptions [GBHG17]. LDA projects the original data to a subspace of lower dimension, the projection directions $\mathbf{w}$ is the solution of the maximization problem of the objective: $\frac{\mathbf{w}^T\mathbf{S}_B\mathbf{w}}{\mathbf{w}^T\mathbf{S}_W\mathbf{w}}$, where $\mathbf{S}_B, \mathbf{S}_W$ are respectively between-class scatter and within-class scatter matrices and is computed, respectively as:

$$\boldsymbol{S}_B = \sum_{c=1}^{n_c} N(\bar{\mu}_c - \bar{\mu})(\bar{\mu}_c - \bar{\mu})^T,$$

$$\boldsymbol{S}_W = \sum_{c=1}^{n_c}\sum_{i=1}^{N}(\boldsymbol{l}_i^c - \bar{\boldsymbol{\mu}}_c)(\boldsymbol{l}_i^c - \bar{\boldsymbol{\mu}}_c)^T,$$

where $\bar{\mu}_c = \frac{1}{N_c}\sum_{i=1}^{N_c}\boldsymbol{l}_i^c$ is the mean of the traces corresponding to variable of class c, and, $\boldsymbol{\mu} = \frac{1}{N}\sum_{c=1}^{n_c}\bar{\boldsymbol{\mu}}_c N_c$ is the total mean of all traces. Finding $\boldsymbol{w}$ is usually reduced to the problem of finding the eigenvectors of the matrix $\boldsymbol{S}_W^{-1}\boldsymbol{S}_B$.

After LDA, the original data (e.g. $\boldsymbol{L}$) is transformed to lower dimension space (e.g. $\boldsymbol{Lw}$) since the number of classes is usually smaller than the original dimension, however, it has been shown in FiXme: LDA in SCA citations that only few of the eigenvectors that corresponds to highest eigenvalues are sufficient. The dimension after projection also affects the performance of LDA.

MGT after LDA

Multi-Layer Perceptron Multi-Layer Perceptron (MLP) acts as a efficient tool for supervised classification problem FiXme: few citations and extensively used in SCA.An MLP is composed of several different trainable layers (i.e.). In its simplest form, MLP usually has a few pair of *linear layer* directly follows by a *non-linear activation function*, sequentially stacked up, and then finally outputs a discriminative model using a *sofmax* layer.

During training phase, the parameters of the layers are gradually changed in the direction that optimize an objective. We set the objective to be minimizing the Negative Logarithm Likelihood Loss, since it was show in [MDP19] to be relevant to profiled SCA.

Using NLL loss as the objective, we can also use the validation loss of the network to estimate the PI value of the training model p, thus, we apply Early Stopping technique, where the criteria is the minimal validation loss (in other word, maximal validation PI).

MLP can be used in our analysis as a general (no specific assumption requires) and straightforward (input the leakage traces directly) model, therefore, it is used to estimate model on sensitive variable as well as on different shares of the target one.

Soft-Analytical Side-channel Attack

Soft-Analytical Side-channel Attack (SASCA) was first introduced in [VCGS14] and has been extensively used in recent studies ([BHM+19], [BS21], [MMMS22], ...). The Belief-Propagation (BP) algorithm at the core of the method allows many time samples (corresponding to many intermediate variables) in the leakages to be exploited simultaneously while keeping the computational cost be reasonable enough. FiXme: wording

We also apply SASCA in this note to extend our study to widen the scenario, similarly in [MMMS22] and [BS21], with some changes to fit our problem.

Recall that the PDF $f(\boldsymbol{l}|x)$ in d-share encoding implementation can be rewritten as the convolution of each share leakage component. Precisely, if $\mathrm{Enc}_d(S) = (X_1, \ldots, X_d)$ then

$$f(\boldsymbol{l}|s) = \sum_{\substack{x_1,\\ \ldots,\\ x_{d-1} \in \mathbb{Z}_Q}} f(\boldsymbol{l}|s, x_1, \ldots, x_{d-1}) \cdot p(x_1, \ldots, x_{d-1}), \tag{6}$$

where $X_i \xleftarrow{\$} \mathbb{Z}_Q$, independently, for $i \in 1, \ldots, d-1$. If we consider independent leakage assumption, we have

$$f(\boldsymbol{l}|x) = \sum_{\substack{x_1,\\ \ldots,\\ x_{d-1} \in \mathbb{Z}_Q}} \prod_{i=1}^{d} f(l_i|x_i) \cdot \prod_{i=1}^{d-1} p(x_i). \tag{7}$$

Plug this expression into Eq.3 we have

$$p\left(s|\boldsymbol{l}\right) = \sum_{\substack{x_1,\\ \ldots,\\ x_{d-1} \in \mathbb{Z}_Q}} \prod_{i=1}^{d} p\left(x_i|l_i\right) \frac{p(s)}{p(x_d)}, \tag{8}$$

where $x_d = s \pm \sum_{i=1}^{d} x_i$.

When $p(x_i|li)$ are known (e.g. results from a model or through knowing $f(l_i|x_i)$ in simulation), we instantiate $r_{ii}^0 = f_i(x_i)$ where $f_i(x_i) = p(x_i|l_i)$ and $r_{ss}^0 = p(s)/Q$, run BP for 2 steps, we can read out the message at note S as:

$$Z_s(s) = \frac{p(s)}{Q} \cdot \sum_{\substack{x_i \in \mathbb{Z}_Q\\ \sum x_i = s}} \prod_{i=1}^{d} p\left(x_i|l_i\right), \tag{9}$$

and obtain expected value.

3 Simulated analysis

We first try to investigate the difference between representations on simulated data. Same reason as several similar studies [BDMS22], [MMMS22] on the related subject, this allows us to inspect them in wider range of scenario (from very low noise to high noise levels, from small number of shares to sufficiently high ones).

Recall that we only focus on masking the prime-field arithmetic operations, and especially on the sensitive variables that lie in small range of the field (i.e. results from CBD sampling). Let the sensitive variable is S, then the sensitive set $\mathcal{S}$ equals to $\{0, 1, 2, -1, -2\}$ corresponding to the probability $P_{\mathcal{S}} = [0.375, 0.25, 0.0625, 0.25, 0.0625]$.

The encoding under investigations are $\mathrm{Enc}_d^{\mathrm{sum}}$ and $\mathrm{Enc}_d^{\mathrm{diff}}$ where the shares X_i are drawn at random from $\mathbb{Z}_Q$ for $i \in \{1, \ldots, d-1\}$, where Q is 3329 as proposed by KYBER's authors. The last share X_d is computed differently for each representation, $X_d^{\mathrm{sum}} = S + \sum_{i=1}^{d-1} X_i$ while $X_d^{\mathrm{sum}} = S - \sum_{i=1}^{d-1} X_i$.

The leakage vector corresponding to the processing of S is $\boldsymbol{L}$ is simulated as the concatenation of the leakage corresponding to the individual shares i.e $\boldsymbol{L} = [L_1, \ldots, L_d]$. We consider Hamming Weight model, i.e. $\delta(X_i) = \mathrm{HW}(X_i)$, since it is the most used in theoretical analysis [SVCO+10] and was proven to be relevant to real-life setups for CMOS devices FiXme: citations. We also assume Gaussian Noise, i.e. $L_i = \mathrm{HW}(X_i) + b_i$ where $b_i \leftarrow \mathcal{N}(0, \sigma^2)$.

To recap, the leakage corresponding to sensitive variable S, which is encoded as $(X_1, X_2, \ldots, X_d)$, is a vector $\boldsymbol{L} = [\mathrm{HW}(X_1) + b_1, \mathrm{HW}(X_2) + b_2, \ldots, \mathrm{HW}(X_d) + b_d]$, where $b_i \leftarrow \mathcal{N}(0, \sigma^2)$. Precisely, the probability of the share leakage give its value is in the form:

$$f(l_i|x_i) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{\mathrm{HW}(x_i)}{\sigma}\right)^2}. \tag{10}$$

Accordingly, SNR can be computed as a function of the noise σ, note that $\mathbb{Z}_Q \subset \mathbb{Z}_{2^{11}}$:

$$\mathrm{SNR} \approx \frac{2.76}{\sigma}$$

Ultimately, we use d (i.e. the number of shares) and σ (i.e. noise level of the implementation) as the parameters of the analysis since they determines the security level of masking countermeasure FiXme: citations.

3.1 2-share leakage distributions

Theoretically, second-order DPA is possible if the joint probability distributions are different for different sensitive values s (i.e. key-dependent). This has been clearly shown to be relevant in the context of an 8-bit Boolean masking, where each Hamming Weight value of 9 possible values produces one distinct leakage distribution (Figures 11, 12, 13 in [SVCO+10]).

Since the computations now moved from binary field to prime field, encoding operation moves from exclusive-or (XOR) to arithmetic ones, the distinctiveness of these distributions becomes less obvious. For example, the distributions for $\mathrm{Enc}_d^{\mathrm{sum}}$ with uniform S over $\mathbb{Z}_5$ is shown in Fig.1, we use histogram to estimate such distributions. For the same $\mathrm{HW} = 1$, $s = 1, 2, 4$ have three distinct shapes.

Intuitively, XOR operation acts independently for each bit, leads to a certain FiXme: wording relations among operands' HWs (e.g. $\mathrm{HW}(X \oplus Y) = \mathrm{HW}(X) + \mathrm{HW}(Y) - 2 \cdot \mathrm{HW}(X \times Y)$), hence, produces well-separate distribution shapes for each value. The arithmetic operations (e.g. addition, subtraction, modulo) let results from lower-order bits affect higher-order bits through the carry (or borrow) bits, leads to more unclear connection.In addition, because it is frequently applied at the end, the binary representation of the moduli has a significant impact on the variety of the distribution shapes.

Add more info on dist of diff prime in A?

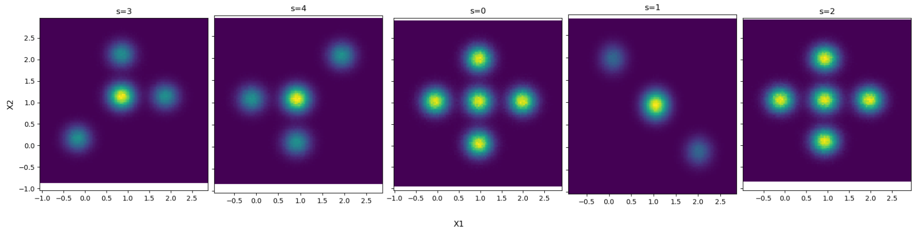


Fig. 1: 2-share leakage distributions over $\mathbb{Z}_5$, uniform sensitive variable.

If S is uniform over $\mathbb{Z}_Q$ there should not be much different between representations, for example, in Fig.6, 7, we can see even they look different in shapes but generally they have the same number of shapes.

However, when we take the small support of S into account, the dissimilarity of these distributions generated by different encoding representation emerges clearly. In Fig.2, there certainly is a significant different between two representations. It is reasonable to question the gap of the amount of information about sensitive variable S given these two sets of distributions.

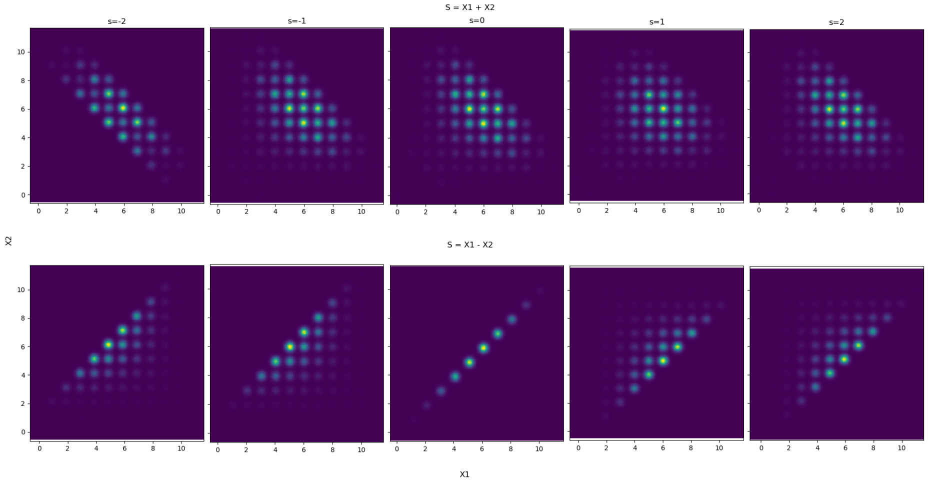
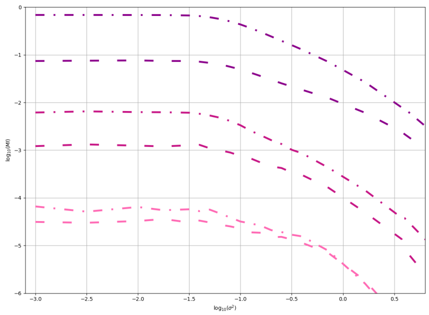


Fig. 2: 2-share leakage distributions over $\mathbb{Z}_Q$.

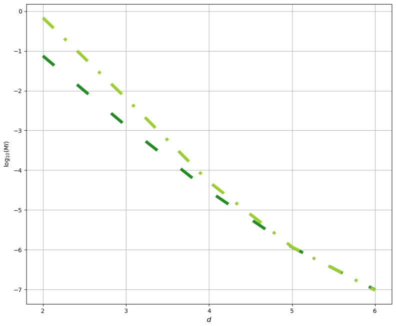
3.2 Quantify observations

We next evaluate concretely the information exposed from the leakages of different encoding representations by its MI with the sensitive variable S. With

$f(li|xi)$ described in Eq.10, we first compute $p(x_i|l_i)$ followed Bayes theorem as in Eq.3, we, then, obtain $p(s|\boldsymbol{l})$ from SASCA. We let d runs from 2 to 6 and σ^2 runs from 10^{-3} to 10^2, the results are shown in Fig.3a.



(a) MI on simulated data d from 2 to 4 shares, dashed line, dot-dashed lines correspond to Enc^{sum}, Enc^{diff} respectively.



(b) MI on noiseless leakages w.r.t. d, dashed line, dot-dashed lines correspond to Enc^{sum}, Enc^{diff} respectively..

There are several noticeable observations from this result:

- First, we confirm once again noise amplification effect of arithmetic masking in prime-order groups under HW leakage assumption that theoretical hinted in [DFS16] and experimentally shown in [MMMS22]. Even when S lies in small set we still have exponential increment of security with respect to number of shares, this trend is stable when noise increases . Additionally, this effect both holds in different encoding representations.
- Next, as anticipated, there is a significant gap between two representations, where leakage from Enc^{diff} exposes more information of the secret than Enc^{sum}. This gap is a factor of ten for $d = 2$ and seems to decreases when the number of shares increases.

We would like to stress the notable gap between two representations. There would be no difference to mask S as Enc^{diff} or Enc^{sum} if S is uniformly distributed over $\mathbb{Z}_Q$. However, when S only take few specific values, operations on shares matter. For example, in 2-share case, $s = 0$ is exposed totally given $\text{HW}(X_1)$ and $\text{HW}(X_2)$ in Enc^{sum} but there is no such value of s in $\mathcal{S}$ has the same issue if Enc^{diff} is used(this could be viewed as an homomorphic attack" as suggested in [DFS16]).

Show this in A?

Additionally, the difference between two ways of encoding diminishes when number of shares d increases. We confirm this trend with smaller prime-order group $\mathbb{Z}_{23}$(Fig.8). We adopt the reduction to random walks as [DFS16] then the length of operation repetition seems to be relevant. For example, in considering $d = 4$, we have $S^{\text{diff}} = X_1 + X_2 + X_3 + X_4$ in Enc^{diff} and $S^{\text{sum}} = -X_1 - X_2 - X_3 + X_4$ in Enc^{sum} then intuitively, $Z_1 + Z_2 + Z_3 + Z_4$ is close to uniform than

Better fig

$-Z_1 - Z_2 - Z_3 + Z_4$. By trying two other representations:

$$S^1 = -X_1 + X_2 + X_3 + X_4$$
$$S^2 = -X_1 - X_2 + X_3 + X4,$$

and compute MI corresponds to each one in the same manner, we obtain results depicted in Fig.4.

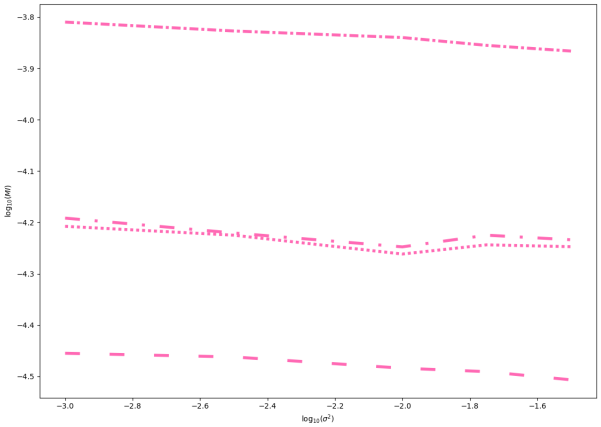


Fig. 4: MI for 4 different representations for 4-share encoding. densely dashdotted, loosely dashdotted, dotted, dashed are S^2, S^{sum}, S^1 and S^{diff} respectively

Better fig

It somehow infers the impact of operation repetition as suspected. Where this length is equal and is 3 in S^{sum} and S^1, we gain the same amount of information about S given $\boldsymbol{L}$. We have the least in S^{diff} where this length is the longest (i.e., 4) and have the most information in S^2 where this length is 2.

We emphasize that this interpretation is merely experimental and lacks of concrete theoretical proof. However, we are certain that this trend can be explained in formal manner.

4 Real-world analysis

We next put the observations in simulation to test with real measurements.

We note that several available polynomial masking for KYBER [BC22], [HKL+22], [DHP+21] and all polynomial masking schemes for KYBER (especially and for PQC in general) such as [OSPG18] FiXme: more citations use $\mathrm{Enc}^{\mathrm{diff}}$ for their representation. This is positive sign as it has been shown in simulation that it allows exponential security with respect to the number of shares

d and Enc^{diff} provide better security compare to Enc^{sum}. However, depending on the implementation, the leakage can actually be in form of Enc^{sum}. Normally to mask a coefficient of the secret s, the computations follow:

1. Copy s to the last (or the first) share $x_d \leftarrow s$.
2. Generate a random number r in $\mathbb{Z}_Q$.
3. Assigning the shares and accumulating the them can be done in two ways:
   - Assign $x_i \leftarrow r$ and accumulate on the last share $x_d = x_d - r$, or
   - Assign $x_i \leftarrow Q - r$ and $x_d = x_d + r$.

Both ways in step 3 eventually leads to Enc^{diff} representation but notice if we have that leakage corresponding to r then the second ways of computing share gives us the leakage corresponding to Enc^{sum}.

We then use the same implementation such as [BC22] but use leakage on x_i and x_d to produce data for Enc^{diff} and use leakage on r and x_d to produce data for Enc^{sum}.

We run our implementations on Arm Cortex-M4 STM32F415 and acquire two million traces for each implementation. We first compute SNR on the data to make sure there is no first-order information of S in the traces and to select relevant PoIs for the next step[1]. In order to compare between implementations, we again, base on IT metric, and PI to be more specific.

We proceed our evaluation in several steps:

1. For each share, we select n_{PoI} points in the traces that have maximum SNR correspond to that share. We then have truncated leakage $\boldsymbol{L}'_i$ for each share. These shares' leakages are then concatenated to be reduced traces $\boldsymbol{L}'$ or kept separately depends on the distinguisher in use.
2. We then use $\boldsymbol{L}'$ or $\boldsymbol{L}'_i$ to model $p(s|\boldsymbol{l}')$ directly or $p(x_i|\boldsymbol{l}'_i)$. In the latter case, we combine $p(x_i|\boldsymbol{l}'_i)$ to obtain $p(s|\boldsymbol{l}')$ eventually.
3. PI is estimated using Eq.4 on the model given by the previous step.

We use different distiguishers to model the unknown $p(\cdot|\cdot)$ to inspect the leakages in different assumptions. The distiguishers are:

Plain MLP We feed MLP the reduced traces $\boldsymbol{L}'$ and use it to directly model $p_(s|\boldsymbol{l})$ and PI is estimated based on MLP's output. The model $p_(s|\boldsymbol{l})$ is estimated without any assumption about the leakage. We make sure that the selected points do not have maximal SNR value w.r.t. the secret to avoid any first-order information of the secret.

MLP x SASCA We keep truncated traces for each share $\boldsymbol{L}'_i$ separately. Then we use MLP to model $p(x_i|\boldsymbol{l}_i)$ for each share. The results then are fed to SASCA to obtain $p(s|\boldsymbol{l})$. We restrict the information of the secret is only from the leakage of its shares, however, there is no assumption on the shares' leakage.

[1] SNR on traces are shown in Fig.9

LDA x SASCA Similar process as MLPxSASCA, we used truncated traces $\boldsymbol{L}'_i$ to build the model for $p(x_i|\boldsymbol{l}_i)$ separately using LDA. In LDA, the truncated traces $\boldsymbol{L}'_i$ are projected to subspace of dimension n_{LDA}, then GMT are applied as described in Sec.2.1 to obtain $p(x_i|\boldsymbol{l}_i)$. Finally, $p(s|\boldsymbol{l})$ is estimated using SASCA. In this setting, shares' leakages are assumed to be multivariate Gaussian distributed and satisfy homoscedasticity.

Optimal n_{PoI} , optimal n_{LDA}

Complete this fig

We compare convergence rates among distiguishers by compute PI correspondingly to increasing number of profiling traces N_p. The results is shown in Fig.5.

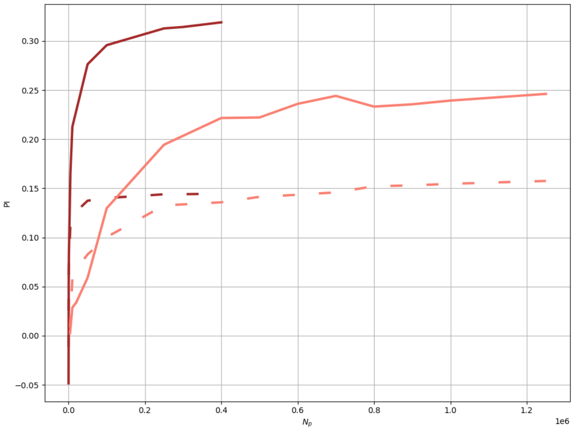


Fig. 5: PI of different distinguishers. Brown and salmon curves correspond to Plain MLP and MLPxSASCA, solid and dashed curves correspond to Enc$^{\text{diff}}$ and Enc$^{\text{sum}}$ respectively.

5 Conclusion

A Supplementary

References

ABD$^+$17. Roberto Maria Avanzi, Joppe W. Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé. Crystals-kyber algorithm specifications and supporting documentation. 2017.

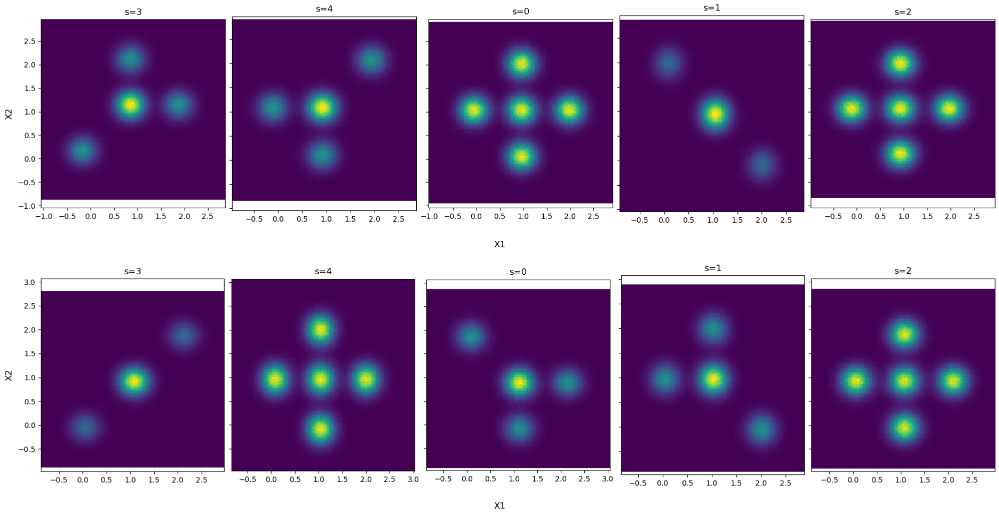


Fig. 6: Leakages distribution for uniform secret over $\mathbb{Z}_5$. Top and bottom figure is corresponding to $\mathrm{Enc}^{\mathrm{sum}}$ and $\mathrm{Enc}^{\mathrm{diff}}$ respectively

ACLZ20. Dorian Amiet, Andreas Curiger, Lukas Leuenberger, and Paul Zbinden. *Defeating NewHope with a Single Trace*, pages 189–205. 04 2020.

BC22. Olivier Bronchain and Gaëtan Cassiers. Bitslicing arithmetic/boolean masking conversions for fun and profit with application to lattice-based kems. Cryptology ePrint Archive, Paper 2022/158, 2022. https://eprint.iacr.org/2022/158.

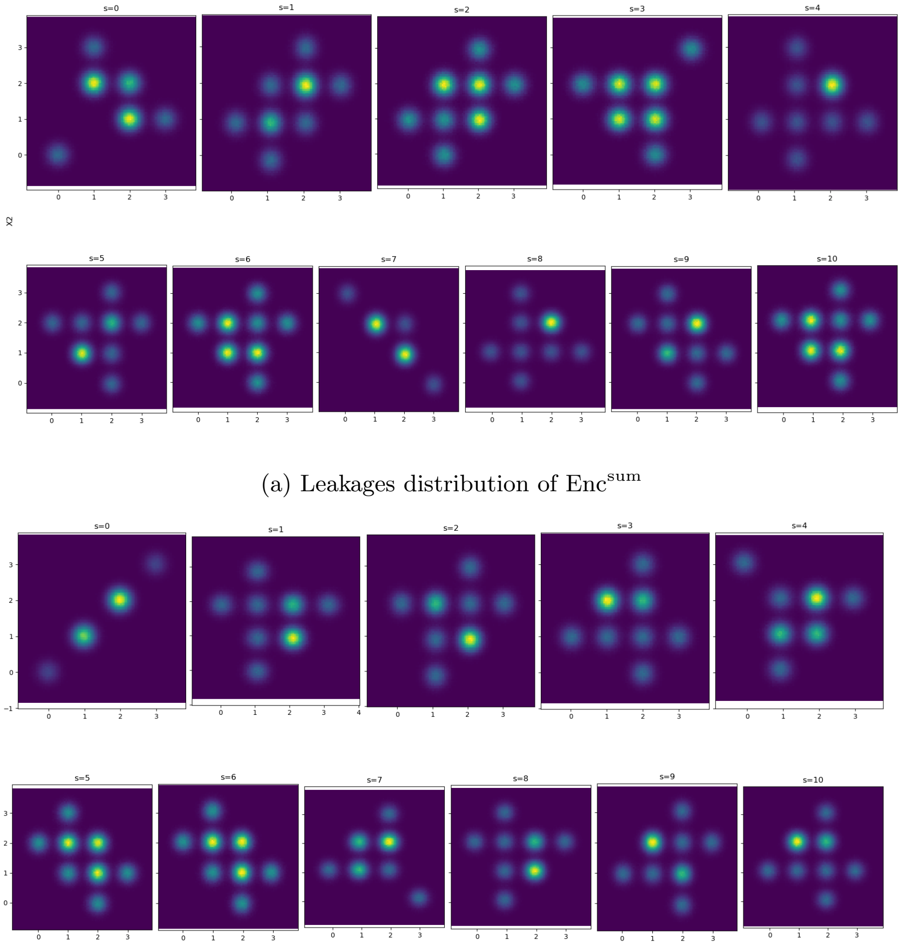
BDK+20. Michiel Van Beirendonck, Jan-Pieter D'Anvers, Angshuman Karmakar, Josep Balasch, and Ingrid Verbauwhede. A side-channel resistant implementation of saber. Cryptology ePrint Archive, Paper 2020/733, 2020. https://eprint.iacr.org/2020/733.

BDMS22. Olivier Bronchain, François Durvaux, Loïc Masure, and François-Xavier Standaert. Efficient profiled side-channel analysis of masked implementations, extended. *IEEE Transactions on Information Forensics and Security*, 17:574–584, 2022.

BHM+19. Olivier Bronchain, Julien M. Hendrickx, Clément Massart, Alex Olshevsky, and François-Xavier Standaert. Leakage certification revisited: Bounding model errors in side-channel security evaluations. Cryptology ePrint Archive, Paper 2019/132, 2019. https://eprint.iacr.org/2019/132.

BS21. Olivier Bronchain and François-Xavier Standaert. Breaking masked implementations with many shares on 32-bit software platforms: or when the security order does not matter. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2021(3):202–234, Jul. 2021.

CGPR19. Eloi Chérisey, Sylvain Guilley, Pablo Piantanida, and Olivier Rioul. Best information is most successful: Mutual information and success rate in side-channel analysis. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 02 2019.



(a) Leakages distribution of $\mathrm{Enc}^{\mathrm{sum}}$

(b) Leakages distribution of $\mathrm{Enc}^{\mathrm{diff}}$

Fig. 7: Leakages distribution for uniform secret over $\mathbb{Z}_{11}$, Fig.7a , Fig.7b corresponds to $\mathrm{Enc}^{\mathrm{sum}}$, $\mathrm{Enc}^{\mathrm{diff}}$ respectively.

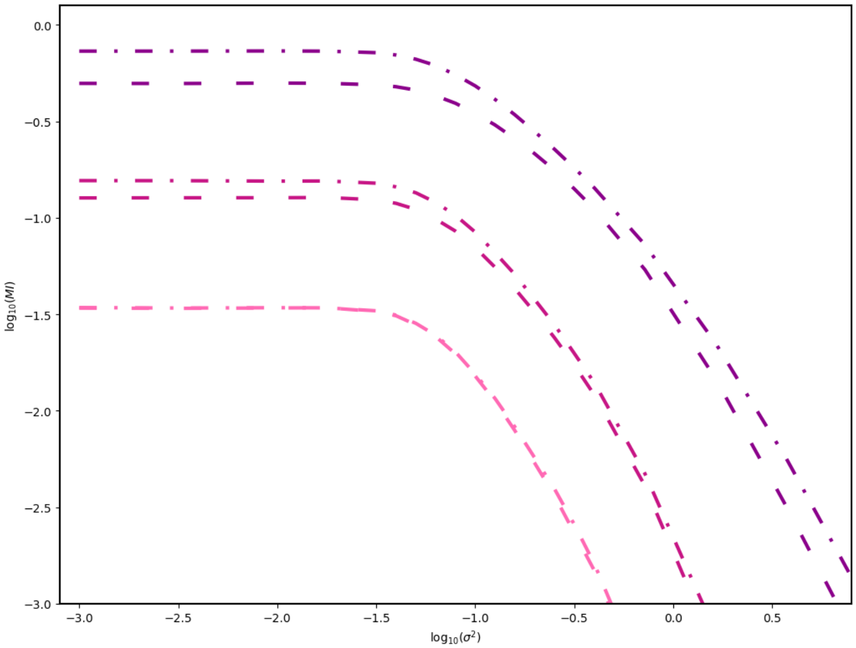


Fig. 8: SNR on traces, top figures correspond to $\mathrm{Enc}^{\mathrm{sum}}$, bottom figures correspond to $\mathrm{Enc}^{\mathrm{diff}}$, left figures are SNR on shares where blue and orange line correspond to the first and second share respectively, right figures are SNR on secret.

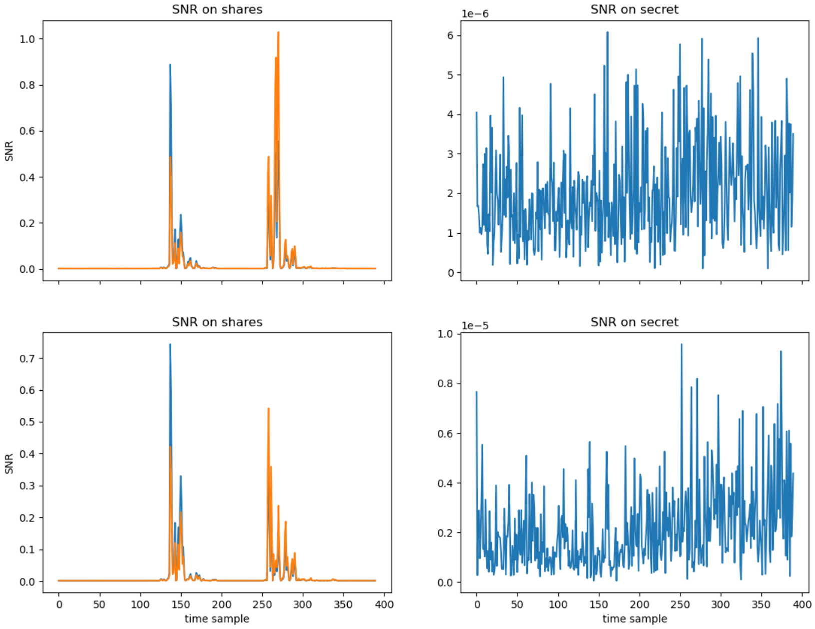


Fig. 9: SNR on traces, top figures correspond to Enc^{sum}, bottom figures correspond to Enc^{diff}, left figures are SNR on shares where blue and orange line correspond to the first and second share respectively, right figures are SNR on secret.

CKA$^+$21. Zhaohui Chen, Emre Karabulut, Aydin Aysu, Yuan Ma, and Jiwu Jing. An efficient non-profiled side-channel attack on the crystals-dilithium post-quantum signature. In *2021 IEEE 39th International Conference on Computer Design (ICCD)*, pages 583–590, 2021.

DFS16. Stefan Dziembowski, Sebastian Faust, and Maciej Skórski. Optimal amplification of noisy leakages. In Eyal Kushilevitz and Tal Malkin, editors, *Theory of Cryptography*, pages 291–318, Berlin, Heidelberg, 2016. Springer Berlin Heidelberg.

DFS18. Alexandre Duc, Sebastian Faust, and François-Xavier Standaert. Making masking security proofs concrete (or how to evaluate the security of any leaking device), extended version. *Journal of Cryptology*, 32, 01 2018.

DHP$^+$21. Jan-Pieter D'Anvers, Daniel Heinz, Peter Pessl, Michiel van Beirendonck, and Ingrid Verbauwhede. Higher-order masked ciphertext comparison for lattice-based cryptography. Cryptology ePrint Archive, Paper 2021/1422, 2021. https://eprint.iacr.org/2021/1422.

GBHG17. Kojo Sarfo Gyamfi, James Brusey, Andrew Hunt, and Elena Gaura. Linear classifier design under heteroscedasticity in linear discriminant analysis. *Expert Systems with Applications*, 79:44–52, 2017.

HKL$^+$22. Daniel Heinz, Matthias J. Kannwischer, Georg Land, Thomas Pöppelmann, Peter Schwabe, and Daan Sprenkels. First-order masked kyber on arm cortex-m4. Cryptology ePrint Archive, Paper 2022/058, 2022. https://eprint.iacr.org/2022/058.

Koc96. Paul C. Kocher. Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems. In *Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings*, volume 1109 of *Lecture Notes in Computer Science*, pages 104–113. Springer, 1996.

LZH$^+$22. Yanbin Li, Jiajie Zhu, Yuxin Huang, Zhe Liu, and Ming Tang. Single-trace side-channel attacks on the toom-cook: The case study of saber. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2022(4):285–310, Aug. 2022.

MBM$^+$22. Catinca Mujdei, Arthur Beckers, Jose Maria Bermudo Mera, Angshuman Karmakar, Lennert Wouters, and Ingrid Verbauwhede. Side-channel analysis of lattice-based post-quantum cryptography: Exploiting polynomial multiplication. Cryptology ePrint Archive, Paper 2022/474, 2022. https://eprint.iacr.org/2022/474.

MDP19. Loïc Masure, Cécile Dumas, and Emmanuel Prouff. A comprehensive study of deep learning for side-channel analysis. Cryptology ePrint Archive, Paper 2019/439, 2019. https://eprint.iacr.org/2019/439.

MMMS22. Loïc Masure, Pierrick Méaux, Thorben Moos, and François-Xavier Standaert. Effective and efficient masking with low noise using small-mersenne-prime ciphers. Cryptology ePrint Archive, Paper 2022/863, 2022. https://eprint.iacr.org/2022/863.

NDGJ21. Kalle Ngo, Elena Dubrova, Qian Guo, and Thomas Johansson. A side-channel attack on a masked ind-cca secure saber kem implementation. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2021(4):676–707, Aug. 2021.

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, Feb. 2018.

PP19. Peter Pessl and Robert Primas. More practical single-trace attacks on the number theoretic transform. Cryptology ePrint Archive, Paper 2019/795, 2019. https://eprint.iacr.org/2019/795.

PPM17. Robert Primas, Peter Pessl, and Stefan Mangard. Single-trace side-channel attacks on masked lattice-based encryption. Cryptology ePrint Archive, Paper 2017/594, 2017. https://eprint.iacr.org/2017/594.

PR13. Emmanuel Prouff and Matthieu Rivain. Masking against side-channel attacks: A formal security proof. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology – EUROCRYPT 2013*, pages 142–159, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

RPBC20. Prasanna Ravi, Romain Poussier, Shivam Bhasin, and Anupam Chattopadhyay. On configurable sca countermeasures against single trace attacks for the ntt - a performance evaluation study over kyber and dilithium on the arm cortex-m4. Cryptology ePrint Archive, Paper 2020/1038, 2020. https://eprint.iacr.org/2020/1038.

SKL$^+$20. Bo-Yeon Sim, Jihoon Kwon, Joohee Lee, Il-Ju Kim, Tae-Ho Lee, Jaeseung Han, Hyojin Yoon, Jihoon Cho, and Dong-Guk Han. Single-trace attacks on message encoding in lattice-based kems. *IEEE Access*, 8:183175–183191, 2020.

SVCO$^+$10. François-Xavier Standaert, Nicolas Veyrat-Charvillon, Elisabeth Oswald, Benedikt Gierlichs, Marcel Medwed, Markus Kasper, and Stefan Mangard. The world is not enough: Another look on second-order dpa. In Masayuki Abe, editor, *Advances in Cryptology - ASIACRYPT 2010*, pages 112–129, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

VCGS14. Nicolas Veyrat-Charvillon, Benoît Gérard, and François-Xavier Standaert. Soft analytical side-channel attacks. 12 2014.

XPR$^+$20. Zhuang Xu, Owen Pemberton, Sujoy Sinha Roy, David Oswald, Wang Yao, and Zhiming Zheng. Magnifying side-channel leakage of lattice-based cryptosystems with chosen ciphertexts: The case study of kyber. Cryptology ePrint Archive, Paper 2020/912, 2020. https://eprint.iacr.org/2020/912.