Spilling-Cascade: an Optimal PKE Combiner for KEM Hybridization*

Céline Chevalier[1,2], Guirec Lebrun[1,3], and Ange Martinelli[3]

[1] DIENS, École normale supérieure, CNRS, PSL University, Inria, Paris, France
[2] CRED, Paris-Panthéon-Assas University
[3] ANSSI, Paris, France

Abstract. Hybrid Post-Quantum cryptography is a cautious approach that aims to guard against the threat posed by the quantum computer, through the simultaneous use of Post-Quantum (PQ) and classical (i.e. pre-quantum) cryptosystems, should the post-quantum schemes used prove insecure.
Regarding the hybridization of Key Encapsulation Mechanisms (KEMs), most recent studies focus on safely combining the symmetric keys output by a parallel execution of classical and Post-Quantum KEMs. While this architecture is straightforward, it appears to lack bandwidth optimization.
Hence, we propose a novel method for hybridizing several KEMs more effectively, by combining the underlying Public-Key Encryption schemes (PKEs) in an innovative variant of the cascade composition that we call "spilling-cascade", before turning the hybrid PKE into a KEM with a FO transformation. We prove that this architecture constitutes a robust combiner for encryption schemes up to IND-CPA security, which permits to eventually generate an IND-CCA-secure KEM.
In terms of performance, our spilling-cascade scheme has a better communication cost than the commonly used parallel combination, with a bandwidth gain of its ciphertext that ranges from 2.8% to 13 % compared to the latter, depending on the number and the characteristics of the PKEs that are combined. Moreover, we prove that for given PKEs to hybridize, the ciphertext communication cost of the spilling-cascade is optimal.

Keywords: PKE combiner · KEM hybridization · Cascade · Post-Quantum Cryptography · Hybrid Key Exchange

1 Introduction

Faced to the looming threat posed by the quantum computer to the classical public-key cryptography, the American National Institute of Standards and

* This work has been partially supported by the European Union - Next Generation EU through the "France Relance" program, and by the French government through the "France 2030" program, under the project RESQUE.

Technology (NIST) launched in 2017 a Post-Quantum Cryptography (PQC) competition aiming to select the best post-quantum Key Encapsulation Mechanism (KEM) and signature algorithms. KEMs, in particular, were chosen as basic bricks to ensure the public-key encryption functionality.

Informally, a KEM is a key transport mechanism between two users, a sender and a recipient, comprising key generation, encapsulation and decapsulation algorithms. During the encapsulation stage, the sender provides the KEM with a public key associated with the recipient and is given, as output, a symmetric key and a ciphertext. The decapsulation algorithm, fed by the recipient with their secret key and the exchanged ciphertext, outputs a symmetric key identical to the one given to the sender at the encapsulation (cf. Section 2.3 for a formal definition). In the PQ framework, many KEMs (e.g. [26], [1], [5], [6], [22], [24]) – and in particular the lattice-based ones – are built by applying an operation named "Fujisaki-Okamoto (FO) transformation" [12], [16], [18] to an OW-CPA[4] or IND-CPA secure PKE, which turns the whole scheme into an IND-CCA KEM.

NIST's PQC competition reached a turning point in summer 2022, when a KEM (Crystals Kyber [26]) and several signature schemes were selected for standardization. One may thus be tempted to use Kyber from now on in order to avoid the "harvest now – decrypt later" attacks that may occur with a quantum computer. However, due to the lack of maturity of post-quantum cryptography as a field of research, especially when it comes to the parametrization of real-life implementations of PQ primitives, it appears much safer to hybridize KEMs by simultaneously using classical and PQ algorithms, in order to benefit from the best security of both worlds, yet at the cost of some overhead.

A natural KEM hybridization in the PQ framework is the "double" one, where a classical key exchange scheme is combined with a single PQ KEM. However, when it comes to sensitive data whose secrecy must be ensured for at least few decades (and which are, consequently, particularly vulnerable to the aforementioned "harvest now – decrypt later" attacks), some users may want to ensure that they do not rely on only one PQ cryptosystem that could be shown insecure in a couple of years, in which case the hybridization would have been totally useless. A n-hybridization ($n > 2$) with one classical algorithm and two or more PQ KEMs solves this problem, but it suffers from important computational and bandwidth overheads that should be limited as much as possible.

There are two different ways to carry out a KEM hybridization: either with a straightforward combination of the selected KEMs or by the combination of the PKEs comprised in these KEMs.

[4] One-Way Chosen-Plaintext Attacks. This encryption security model, which aims to prevent the full recovery of a plaintext by an adversary having access to an encryption oracle, is weaker than models relying on "semantic security", where the adversary tries to distinguish the ciphertexts associated to chosen plaintexts, with the help of an encryption oracle (IND-CPA) and potentially an adaptive decryption oracle (IND-CCA).

KEM Combination. The first method – the combination of KEMs seen as black-boxes – is the most common. Its classical architecture is the parallel composition (cf. Section 2.4) where all component KEMs are running in parallel and output their own symmetric key and related ciphertext. Then, a dedicated primitive named "key combiner" (a.k.a. "core function" in [13]) mixes the symmetric keys with the ciphertexts to produce a combined key.

Using KEMs as core components is a natural approach, since these primitives were precisely targeted by the NIST for standardization in its PQC competition. However, the parallel composition of KEMs appears not to be the most effective way to generate a hybrid KEM, for the following reasons:

- Regarding the computation cost, combining n KEMs that are built with a FO transformation means carrying out this FO transformation n times as well, whereas we would like to reduce this number to a single operation.
- More importantly, the bandwidth of a key exchange using such a hybrid KEM (n public-keys and n ciphertexts) is not optimized, and this issue is even more pronounced with PQ algorithms that already have an important communication cost.

PKE Combination. The other – less studied – method is the combination of the PKEs within the KEMs that we want to hybridize, in order to produce a hybrid PKE that a single FO transformation will turn into an IND-CCA KEM. The main idea here is to save the computation cost of $(n-1)$ FO transformations, since this one must be applied only once, on the hybrid PKE. Furthermore, as this (intermediate) hybrid PKE only gets to be OW-CPA or IND-CPA secure, the key combiner can afford to be simpler – and thus computationally more sober – than in the IND-CCA security model. Indeed, it was proved by [13] that merely XORing the keys of component IND-CPA KEMs was sufficient to ensure the same security for the hybrid algorithm[5].

To the best of our knowledge, the only study of a hybrid KEM based on a PKE combination is [19], which presents a generic construction where PKEs are run in parallel composition before the resulting hybrid PKE undergoes a single FO transformation. However, the gains of this construction, compared to a KEM combination, are purely computational and in particular, the bandwidth is left unaltered.

Outside the scope of public-key cryptography, the combination of secret-key encryption primitives has been far more studied, and notably the cascade composition (cf. Section 2.4). The early works of [10] and [21] focus on cascades of block ciphers; the former proves the security of the hybrid scheme against message-recovery attacks, whereas the latter shows that a cascade of ciphers is at least as secure as the first cipher but that the security of the construction cannot be ensured, in the corresponding security model, by the second cipher only.

[5] This result can be extended to the case of encryption schemes.

At a higher level, [15] proves that a cascade of encryption schemes yields IND-RCCA security[6] if at least one of its components achieves that security. In a similar study, [30] introduces the notion of "multi-encryption" and shows that the cascade scheme can reach IND-gCCA security[6] in their slightly different multi-encryption security model.

Our Contributions and Outline of this Paper. We present in Section 3 of this paper a new way to combine encryption schemes, designed to keep the computational gains of the PKE parallel combination of [19] while offering a lighter bandwidth. This construction, that we call "spilling-cascade", is a mix between parallel and cascade combinations. We prove in Section 3.3 that our spilling-cascade scheme is an IND-CPA secure robust combiner for encryption schemes (in the sense of Definition 5), which is a sufficient condition to build an IND-CCA KEM from the hybrid PKE, *via* the appropriate FO transformation (cf. Section 2.3). We also study in Section 3.5 the functionalities offered by such an architecture, mainly the aforementioned KEM hybridization but also a lighter Authenticated Key Exchange (AKE) protocol that can be used, for instance, to implement the recent KEM-TLS proposal [27].

We then instantiate in Section 4 two types of spilling-cascade hybrid KEMs:

- a first one combining a classical encryption scheme (ElGamal) and a PQ PKE (Crystals Kyber);
- a second one, where another PQ PKE (either NTRU-HRSS [20], its variant npNTRU from [25] or BAT [11]) is added to the former double hybridization.

We also refer to some possible improvements that are further detailed in the full version, including one, named "Integrated Diffie-Hellman (IDH) KEM", in which the first classical PKE in the cascade chain is replaced by a Diffie-Hellman key agreement scheme.

Finally, we prove in Section 5 that the spilling-cascade architecture has an *optimal* ciphertext communication cost when it comes to PKE combination – under some conditions –, and we compare in practice the communication efficiencies of the instantiations from Section 4.

2 Preliminaries

In the whole paper, we include both symmetric and public-key cryptosystems under the generic term of "encryption scheme", our spilling-cascade combiner being proved secure for both primitives. However, for practical purpose, we use the notations of public-key cryptography in most instances and figures, as we use PKEs to build KEMs. This does not narrow the range of our study.

[6] IND-RCCA [4] and IND-gCCA [2] are relaxed versions of the IND-CCA security notion, designed to exclude trivial security failures in the IND-CCA model that come from "benign malleability".

2.1 Notations and Terminology

In all the document, sampling an element s from a space $\mathcal{S}$ with uniform distribution is represented by $s \leftarrow\!\!\$\ \mathcal{S}$. The output of a probabilistic algorithm is represented by $\leftarrow$ and that of a deterministic algorithm is given by $:=$.

$[\cdot]$ denotes optional values or parameters. $\lfloor\ \rceil$ and $\lceil\ \rceil$ respectively denote the rounding and ceiling values of a decimal number.

Regarding bit-strings: $\cdot||\cdot$ is used for the concatenation operation. $|\cdot|$ represents the bit-length of a string; when applied on an arbitrary element, it denotes the bit-length of the string encoding that element. Moreover, we use the notation $a \sqsubseteq b$ to mean that the bit-string representing an arbitrary element a is included in the bit-string encoding another element b (and conversely for $a \sqsupseteq b$).

In the context of lattice-based cryptography, vectors are written in lower case bold characters, matrices in upper case characters and scalars in lower case roman characters.

2.2 Public-Key Encryption Scheme (PKE)

Definition 1 (Public-Key Encryption Scheme).
A probabilistic public-key encryption scheme, defined over finite message and ciphertext spaces $\mathcal{M}$ and $\mathcal{C}$, is a tuple of polynomial-time algorithms (KeyGen, Enc, Dec) *such that:*

- **KeyGen** *takes as input the security parameter λ and probabilistically outputs a couple of public and private keys that lie in public and private key spaces $\mathcal{PK}$ and $\mathcal{SK}$:* $\boxed{(pk, sk) \leftarrow \text{KeyGen}(1^\lambda)}$.
- **Enc** *takes as input a public-key $pk \in \mathcal{PK}$ and a plaintext m. It maps the plaintext into the the message space $\mathcal{M}$ – notably by adding a padding p if necessary – and probabilistically yields a ciphertext $c \in \mathcal{C}$. This algorithm becomes deterministic if its internal randomness r is considered as part of the parameters:* $\boxed{c \leftarrow \text{Enc}(pk, m) \quad \Leftrightarrow \quad c := \text{Enc}(pk, m; r)}$.
- **Dec** *takes as input a secret-key $sk \in \mathcal{SK}$ and a ciphertext $c \in \mathcal{C}$ and deterministically outputs a plaintext $m \in \mathcal{M}$:* $\boxed{m := \text{Dec}(sk, c)}$

Worst-Case Correctness. If the encryption scheme is *perfectly* correct, we have: $\forall m \in \mathcal{M}, \forall (pk, sk) \leftarrow \text{KeyGen}(1^\lambda),\ \text{Dec}(sk, \text{Enc}(pk, m) = m$.

However, many PKEs, especially in the post-quantum framework, do not have this property and it appears useful to statistically bound their correctness error. Specifically, the Fujisaki-Okamoto transformation (cf. below) considers that correctness error in its worst-case.

Definition 2 (Worst-Case Correctness (from [18])). *The worst-case correctness of a public-key encryption scheme* PKE*, with a message space $\mathcal{M}$ and a randomness space $\mathcal{R}$, is defined as:*

$$\delta^{wc} := \mathop{\mathbb{E}}_{(pk,sk)\leftarrow \text{KeyGen}(1^\lambda)} \left(\max_{m \in \mathcal{M}} \left\{ \Pr_{r \in \mathcal{R}} \left[\text{Dec}\big(sk, \text{Enc}(pk, m; r)\big) \neq m \right] \right\} \right) \tag{1}$$

γ-Spreadness. This property, issued from [12], lower-bounds the min-entropy of the ciphertexts output by an encryption scheme over its internal randomness, for every given message and key-pair from the message and keys spaces.

Definition 3 (γ-Spreadness of a PKE (from [12])). *A probabilistic public-key encryption scheme with message space $\mathcal{M}$, ciphertext space $\mathcal{C}$ and randomness space $\mathcal{R}$ is said to be γ-spread iff:*

$$\forall (pk, sk) \leftarrow \mathrm{KeyGen}(1^\lambda), \forall m \in \mathcal{M}, \quad \max_{c \in \mathcal{C}} \left\{ \Pr_{r \in \mathcal{R}} [\mathrm{Enc}(pk, m; r) = c] \right\} \leq 2^{-\gamma} \tag{2}$$

IND-CPA Security. The security game corresponding to the IND-CPA security of public-key encryption schemes, with two experiments $\mathbf{Exp}^{\mathrm{IND-CPA-b}}$ ($b \in \{0, 1\}$), is detailed in Figure 1. In this model, the advantage of any Probabilistic Polynomial-Time (PPT) adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ (where $\mathcal{A}$ wins the security experiment $\mathbf{Exp}^{\mathrm{IND-CPA-b}}$ if its guessing bit $\hat{b}$ equals b) is:

$$\mathrm{adv}_{\mathcal{E}}^{\mathrm{IND-CPA}}(\mathcal{A}) = \left| \Pr[1 \leftarrow \mathcal{A}]_{\mathcal{E}}^{\mathrm{IND-CPA-1}} - \Pr[1 \leftarrow \mathcal{A}]_{\mathcal{E}}^{\mathrm{IND-CPA-0}} \right| \tag{3}$$

Experiment $\mathbf{Exp}^{\mathrm{IND-CPA-b}}_{\mathrm{PKE}}$

1 : $(pk, sk) \leftarrow \mathrm{KeyGen}(1^\lambda)$
2 : $(m^0, m^1, st) \leftarrow \mathcal{A}_1(pk)$
3 : $c^* \leftarrow \mathrm{Enc}(pk, m^b)$
4 : $\hat{b} \leftarrow \mathcal{A}_2(pk, c^*, m^0, m^1, st)$
5 : **return** $\hat{b}$

Experiment $\mathbf{Exp}^{\mathrm{IND-CCA-b}}_{\mathrm{KEM}}$

1 : $(pk, sk) \leftarrow \mathrm{KeyGen}(1^\lambda)$
2 : $(k_0^*, c^*) \leftarrow \mathrm{Encaps}(pk)$
3 : $k_1^* \leftarrow\!\!\$\ \mathcal{K}$
4 : $\hat{b} \leftarrow \mathcal{A}^{\mathcal{O}^{Decaps}}(pk, c^*, k_b^*)$
5 : **return** $\hat{b}$

Decapsulation oracle $\mathcal{O}^{Decaps}(c \neq c^*)$

1 : $k := Decaps(sk, c)$
2 : **return** k

Fig. 1: IND-CPA security game for a public-key encryption scheme (left) and IND-CCA security game for a KEM (right).

2.3 Key Encapsulation Mechanism (KEM)

Definition 4 (Key Encapsulation Mechanism). *A KEM is a tuple of algorithms (KeyGen, Encaps, Decaps) defined as follows:*

- **KeyGen** *takes as input the security parameter λ and probabilistically outputs a couple of public and private keys:* $\boxed{(pk, sk) \leftarrow \mathrm{KeyGen}(1^\lambda)}$.
- **Encaps** *takes as input a public-key $pk \in \mathcal{PK}$ and probabilistically yields a symmetric key and an associated ciphertext c:* $\boxed{(k, c) \leftarrow \mathrm{Encaps}(pk)}$.
- **Decaps** *takes as input a secret-key $sk \in \mathcal{SK}$ and a ciphertext $c \in \mathcal{C}$ and deterministically outputs a symmetric key k:* $\boxed{k := \mathrm{Decaps}(sk, c)}$ *or (for an explicit rejection) a symbol $\perp$ stating that the encapsulation is invalid.*

We say that the KEM is correct if:
$\forall (pk, sk) \leftarrow \text{KeyGen}(1^\lambda),\ \big(\text{Decaps}(sk, c), c\big) \leftarrow \text{Encaps}(pk)$.

Security. The security generally expected for a KEM – and that was explicitly stated by the NIST for its PQC competition, is the IND-CCA(2) security. It relies on the indistinguishability of the output symmetric key, while the adversary has access to a decapsulation oracle (cf. Figure 1 for the related security game).

FO Transformation. A common construction of an IND-CCA KEM in the (Quantum) Random Oracle Model ((Q)ROM) consists in applying, to an IND-CPA secure PKE, an operation called "Fujisaki-Okamoto (FO) transformation" [12], [28]. [16] later proposes a modular approach of the original transformation, with several subroutines that must be selected according to the features of the underlying PKE. However, the security bounds given by [16] appear non-tight in the QROM. In this framework, [25], later improved by [18], uses the concept of "Disjoint Simulatability" (DS) to implement an alternate modular FO transformations with tighter security bounds in the QROM.

Composition. The FO transformation of [18] comprises two subroutines that transform any IND-CPA PKE into an IND-CCA scheme. In particular, the property of DS, that is naturally satisfied by many code-based and lattice-based PKEs, can be achieved by any PKE – through the "puncturing" (Punc) operation – at a negligible cost.

- "Punc" transformation: it turns any (probabilistic) IND-CPA secure PKE PKE_0 into another scheme PKE_1 with the Disjoint Simulatability (DS) property. To do so, this operation removes an arbitrary message $\hat{m}$ from the message space of the original scheme: $\boxed{\mathcal{M}_1 := \mathcal{M}_0 \backslash \{\hat{m}\}}$ and uses that message $\hat{m}$ to sample fake encryptions.
- $\text{FO}^{\not\perp}_{m[,c]} = \text{U}^{\not\perp}_{m[,c]} \circ T$ operation, with $\text{U}^{\not\perp}_{m[,c]}$ and T as defined in [16]:
  - T turns the IND-CPA secure and Disjoint Simulatable PKE PKE_1 into a deterministic OW-PCVA[7] scheme by deriving its internal randomness r from the input plaintext m instead of randomly drawing it: $\boxed{r := G(m[, pk])}$[8], with G a cryptographic hash function modeled as a quantum random oracle. In the decryption stage, the validity of the decrypted plaintext also generally needs to be checked *via* a re-encryption.
  - $\text{U}^{\not\perp}_{m[,c]}$ derives the symmetric key from the random plaintext that was encapsulated (instead of straightly using this message as the key):

[7] The One-Way Plaintext-Checking and Validity Attacks (OW-PCVA) security notion is the OW-CPA security where the adversary has an additional access to both a Plaintext Checking Oracle (which states if a couple (plaintext, ciphertext) is related) and a Ciphertext Validity Oracle (asserting the validity of a proposed ciphertext).

[8] The inclusion in T operation of the recipient's public key, or part of it, corresponds to a multi-user variant [8] that is used, for instance, by ML-KEM.

$\boxed{k := H(m[,c])}$, with H a cryptographic hash function also modeled as a quantum random oracle.

In $\text{FO}^{\not\perp}_{m[,c]}$, any decapsulation error of the FO-built KEM is implicitly rejected, with a random key z that is output instead of the symmetric key.

Hence, the modular FO transformation from [18] can be summarized as follows – with straight and dashed arrows respectively denoting tight and non-tight reductions in the QROM:

$$\underset{\text{PKE}_0}{\overset{\text{IND-CPA}}{}} \xrightarrow{\text{Punc}} \underset{\text{PKE}_1}{\overset{\text{IND-CPA+DS}}{}} \dashrightarrow^{\text{U}^{\not\perp}_{m[,c]} \circ \text{T}} \underset{\text{KEM}}{\overset{\text{IND-CCA}}{}} \tag{4}$$

Security Bounds. Let us consider an IND-CPA secure PKE PKE_0, γ-spread and with a worst-case correctness δ^{wc} (cf. Section 2.2). According to Theorem 3.6 from [18], if we apply a FO transformation to that PKE, then for any adversary $\mathcal{A}$ against the IND-CCA security of the output KEM, there exist quantum adversaries $\mathcal{B}_1$ and $\mathcal{B}_2$ against the IND-CPA security of PKE_0, with running times similar to that of $\mathcal{A}$ and s.t.:

$$\begin{aligned}\text{Adv}^{\text{IND-CCA}}_{\text{KEM}}(\mathcal{A}) \leq (8 \cdot (3 \cdot q_G + 2 \cdot q_H + q_D + 6)^2 + 1) \cdot \delta^{wc} + \text{Adv}^{\text{IND-CPA}}_{\text{PKE}_0}(\mathcal{B}_1) \\ + 2 \cdot \sqrt{(q_G + q_H) \cdot \text{Adv}^{\text{IND-CPA}}_{\text{PKE}_0}(\mathcal{B}_2) + \frac{4(q_G + q_H)^2}{|\mathcal{M}| - 1}} + 2^{-\gamma}\end{aligned} \tag{5}$$

q_D denotes the number of classical decapsulation queries of the adversary and q_G and q_H the number of queries to the random oracles G and H.

2.4 Combination of KEMs and Encryption Schemes

Robust Combiner. The combination of several primitives of the same type can be used to guaranty a certain property when some of the primitives that are combined – but we do not necessarily know which ones – fulfill themselves this property. The algorithm that mixes together these primitives is called a combiner, for which [15] has defined the notion of "robustness". We adapt hereunder this concept of robust combiner to the case of encryption schemes.

Definition 5 (Robust Combiner for Encryption Schemes). *Let s_{sec} be a predicate, associated with a security level SEC[9], that takes as input an encryption scheme $\mathcal{E}$ and outputs 1 if $\mathcal{E}$ respects that security level, and 0 otherwise.*

A combiner for encryption schemes is an algorithm that takes as input n arbitrary encryption schemes $(\mathcal{E}_i)_{i \in [\![1,n]\!]}$ and that outputs a single encryption scheme $\mathcal{E}_{hy}$ – called hybrid. We say that this combiner is SEC robust if the output hybrid scheme yields SEC security, provided that at least one of its component schemes is itself SEC secure and regardless of any other assumptions about these component schemes:

$$\forall n \in \mathbb{N}^* \backslash \{1\}, \forall \mathcal{E}_{i \in [\![1,n]\!]}, \quad \sum_{i=1}^{n} s_{sec}(\mathcal{E}_i) \geq 1 \Rightarrow \; s_{sec}(\mathcal{E}_{hy}) = 1 \tag{6}$$

We now present beneath the two main modes of combination of KEMs and encryption schemes : the parallel and cascade (sequential) compositions.

[9] Such as OW-CPA, IND-CPA, IND-CCA...

Parallel Composition. The common way to hybridize KEMs or encryption schemes is called the "parallel composition" (cf. [13] for KEM combination). This method consists in running in parallel each one of the n component primitives that we want to hybridize, in order to generate a combined public-key $pk := (pk_1, \ldots, pk_n)$, a combined ciphertext $c := (c_1, \ldots, c_n)$ and, in the case of KEMs, a combined key $k_c := \mathrm{Comb}(k_1, \ldots, k_n, c)$, with Comb a key combiner.

Cascade Composition. The combination of n encryption schemes $(\mathcal{E}_i := (\mathrm{KeyGen}_i, \mathrm{Enc}_i, \mathrm{Dec}_i))_{i \in [\![1,n]\!]}$ is called a cascade (a.k.a. sequential) composition if every ciphertext output by the first $n-1$ encryption schemes is given as input to the following encryption scheme:

$$\mathrm{Enc}_{casc}((pk_1, \ldots, pk_n), m) := \mathrm{Enc}_n(pk_n, \mathrm{Enc}_{n-1}(pk_{n-1}, \ldots, \mathrm{Enc}_1(pk_1, m))) \quad (7)$$

3 Spilling-Cascade Hybridization of Encryption Schemes

3.1 Description

Because the ciphertexts output by most post-quantum PKEs are much larger than their input plaintexts, it is not possible to combine these schemes in a regular cascade. We consequently conceive, for the purpose of PQ KEM hybridization, the spilling-cascade combination where only a part of the intermediate ciphertexts is encapsulated as in a regular cascade. The remaining part of these intermediate ciphertexts is joined to the last ciphertext c_n of the chain, in order to produce the global ciphertext of the hybrid scheme (cf. Figure 2 & Figure 3).

This architecture aims to keep most of the advantages brought by the regular cascade hybridization, compared to the more common parallel composition, when the regular cascade combination of encryption schemes appears impossible. It indeed performs a hybridization more effective than the parallel one in terms of bandwidth and can still be proved a robust combiner – in the sense of Definition 5 – under certain assumptions, as detailed beneath.

Definition 6 (Spilling-Cascade Combination of Encryption Schemes). *The combination of n encryption schemes $(\mathcal{E}_i := (\mathrm{KeyGen}_i, \mathrm{Enc}_i, \mathrm{Dec}_i))_{i \in [\![1,n]\!]}$ is called a $\mathcal{S}$-spilling-cascade composition if there exists a non-empty subset of the $n-1$ first indices, $\mathcal{S} \subseteq \{1, \ldots, n-1\}$, such that:*

1. *For every encryption scheme $\mathcal{E}_j$, $j \in \mathcal{S}$, in this subset – called a "spilling primitive" –, the generated ciphertext is of the form $c_j = \{u_j, v_j\}$ s.t.:*
   - *u_j is made public as part of the global ciphertext, along with the output c_n of the last scheme of the chain:* $\boxed{c_{sc} := \big((u_j)_{j \in \mathcal{S}}, c_n\big)}$;
   - *v_j, that has a bit-length corresponding to the input size of the following encryption algorithm Enc_{j+1} ($|v_j| = |m_{j+1}|$), is given as input to that scheme:* $\boxed{c_{j+1} \leftarrow \mathrm{Enc}_{j+1}(pk_{j+1}, v_j)}$.

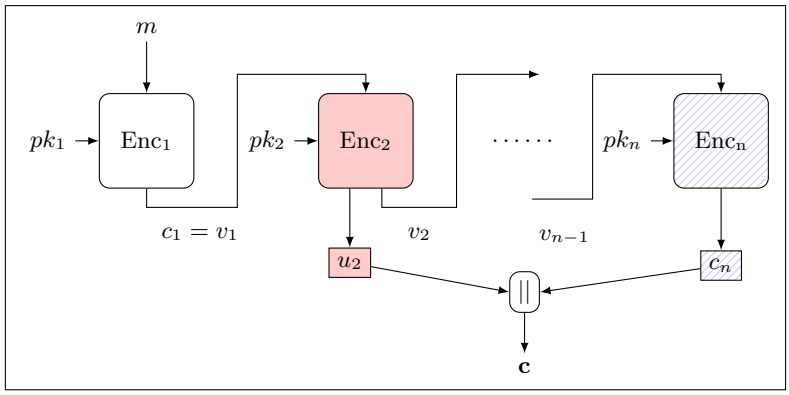

Fig. 2: Architecture of a {2}-spilling-cascade. In this instance, the encryption scheme $\mathcal{E}_2$ publicly displays a part u_2 of its ciphertext, whereas the other encryption schemes are combined in a regular (i.e. non spilling) cascade.

2. *Any encryption scheme $\mathcal{E}_i$, $i \notin \mathcal{S}$, outside the subset $\mathcal{S}$ is composed in regular cascade and thus feeds the next primitive* Enc_{i+1} *with its full ciphertext*[10] $c_i = v_i$ *s.t.* $|c_i| \leq |m_{i+1}|$ *:* $\boxed{c_{i+1} \leftarrow \mathrm{Enc}_{i+1}(pk_{i+1}, c_i)}$.

In this paper, we focus on a hybridization where all upstream primitives are spilling, that we call a fully spilling-cascade, as stated below.

Definition 7 (Fully Spilling-Cascade). *A fully spilling-cascade composition of n encryption schemes is a $\{1, \ldots, n-1\}$-spilling-cascade combination of these schemes, i.e. a spilling-cascade where* ***all*** *the $n-1$ first primitives $(\mathcal{E}_i)_{i \in [\![1, n-1]\!]}$ are spilling a part of their ciphertexts in the global ciphertext.*

3.2 Partitioned-Ciphertext Encryption Schemes

The security offered by the spilling-cascade architecture is based on a property of encryption schemes, that we name "partitioned-ciphertext". It states that the ciphertexts yielded by such encryption schemes can be decomposed into two parts, c_r and c_m, such that c_r does not depend on the encrypted plaintext but rather (generally) on the randomness of the probabilistic encryption, whereas c_m comprises all the information related to the plaintext.

As it is proved hereunder, the security of a $\mathcal{S}$-spilling-cascade composition can be ensured if and only if the spilling primitives $(\mathcal{E}_j)_{j \in \mathcal{S}}$ of the chain have this partitioned-ciphertext property *and* if their c_m element is included in the encapsulated part v of their ciphertext: $\forall j \in \mathcal{S},\ c_{m_j} \sqsubseteq v_j$ (which, in turn, implies that $c_{r_j} \sqsupseteq u_j$).

[10] In the case where $|c_i| < |m_{i+1}|$, a padding p_i of size $|m_{i+1}| - |c_i|$ is added in order to map c_i to the message space $\mathcal{M}_{i+1}$ of the scheme $\mathcal{E}_{i+1}$, according to Definition 1 and as in the regular cascade construction of Section 2.4. This padding is determined by the specifications of the associated encryption scheme.

Spilling-cascade $\mathbf{Enc_{sc}}(pk, m, \mathcal{S})$

$(pk_1, \ldots, pk_n) := \text{Parse}(pk)$
$c := \varnothing$
$v_0 := m$
for $i \in [\![1, n-1]\!]$ **do** :
  $c_i \leftarrow \text{Enc}_i(pk_i, v_{i-1})$
  if $i \in \mathcal{S}$ **then** :
    $c_i := (u_i, v_i)$
  else :
    $c_i := v_i$
  $c := (c, u_i)$
$c_n \leftarrow \text{Enc}_n(pk_n, v_{n-1})$
$c := (c, c_n)$
return c

Spilling-cascade $\mathbf{Dec_{sc}}(sk, c, \mathcal{S})$

$(sk_1, \ldots, sk_n) := \text{Parse}(sk)$
$((u_i)_{i \in \mathcal{S}}, c_n) := \text{Parse}(c)$
$v_{n-1} := \text{Dec}_n(sk_n, c_n)$
for $i \in [\![n-1, 1]\!]$ **do** :
  if $i \in \mathcal{S}$ **then** :
    $c_i := (u_i, v_i)$
  else :
    $c_i := v_i$
  $v_{i-1} := \text{Dec}_i(sk_i, c_i)$
$m := v_0$
return m

Fig. 3: Encryption and decryption algorithms of a $\mathcal{S}$-spilling-cascade encryption combiner. $\mathcal{S} \subseteq \{1, \ldots, n-1\}$ denotes the set of spilling primitives in the cascade chain. The key generation, identical to that of a parallel combination [19], is not detailed.

Definition 8 (Partitioned-Ciphertext Encryption Scheme). *An encryption scheme $\mathcal{E} = (\text{KeyGen}, \text{Enc}, \text{Dec})$ is said to have a "partitioned ciphertext" if any ciphertext c output by that scheme can be decomposed into two non-empty parts c_r and c_m*[11] *such that:*

1. *The bit-lengths of c_r and c_m are constant over the message space $\mathcal{M}$;*
2. *c_r does not depend on the plaintext m.*

More formally: $\forall pk \in \mathcal{PK},\ \forall r \in \mathcal{R},\ \forall (m, m') \in \mathcal{M}^2$, $(c_r, c_m) := \text{Enc}(pk, m; r)$ and $(c'_r, c'_m) := \text{Enc}(pk, m'; r)$ s.t.

1. $|c_r| = |c'_r|$ and $|c_m| = |c'_m|$
2. $c_r = c'_r$.

A common instance of a partitioned-ciphertext classical encryption scheme is ElGamal, that yields a ciphertext of the form $c_{eg} := (c_r = g^r, c_m = m.X^r)$, with r a random coin and m the plaintext to encrypt.

Regarding PQ algorithms, the PKEs of three of the four KEMs that are most likely to be standardized in the future[12] have this partitioned-ciphertext property: BIKE, HQC and Kyber – which has a ciphertext $c_{ky} := (c_r = A^T.\mathbf{r} + \mathbf{e_1}, c_m = \mathbf{t}^T.\mathbf{r} + e_2 + \left\lfloor \frac{q}{2} \right\rceil .m)$, with $\mathbf{r}$ and m as for ElGamal.

We prove in Appendix A.1 the partitioned-ciphertext property of ElGamal and Kyber, that we precisely use in our instantiations of Section 4.

[11] As in Definition 6, the relative order of c_r and c_m in the ciphertext c is not determined in the general case.

[12] These promising algorithms are the current winner of round 3 of NIST's PQC competition (Crystals Kyber, standardized as "ML-KEM") as well as the remaining three candidates of the fourth round, studied as an alternative solution to Kyber (BIKE, ClassicMcEliece & HQC).

3.3 IND-CPA Security of Spilling-Cascade

Theorem 1. *The combination of n encryption schemes* $(\mathcal{E}_i = (\mathrm{KeyGen}_i, \mathrm{Enc}_i, \mathrm{Dec}_i))_{i\in[\![1,n]\!]}$ *in a* $\mathcal{S}$*-spilling-cascade mode constitutes a robust IND-CPA secure combiner for encryption schemes, in the sense of Definition 5* [13]*, if and only if all spilling primitives are partitioned-ciphertext algorithms whose message-dependent part of the ciphertext* (c_m) *is encapsulated by the next encryption scheme in the chain:* $\forall j \in \mathcal{S},\ c_{m_j} \sqsubseteq v_j$.

Nota : Breaking the security of a spilling primitive – from the $n-1$ upstream schemes – in a spilling-cascade chain, due to a mathematical flaw or a sufficient computational power of the adversary, does not question at all the partitioned-ciphertext property of this encryption scheme. All information regarding the plaintext remains contained in the message-dependent part c_m of its ciphertext, which is encapsulated by the next encryption scheme in the spilling-cascade chain and thus never made public.

Proof Sketch. The conditions expected in Theorem 1 from any spilling primitive $\mathcal{E}_{s\in[\![1,n-1]\!]}$ in a spilling-cascade chain are the following:
- Condition 1: that spilling primitive $\mathcal{E}_s$ has a partitioned-ciphertext;
- Condition 2: the message-related part of its ciphertext is encapsulated by the following encryption scheme: $c_{m_s} \sqsubseteq v_s$.

Our proof consists of two statements, that correspond respectively to the "sufficient" and "necessary" conditions of the IND-CPA robustness of the spilling-cascade combiner:

1. If, in a spilling-cascade combination of n encryption schemes, any spilling primitive respects both conditions 1 and 2, then that spilling-cascade is an IND-CPA robust combiner for encryption schemes.
2. Conversely, if the spilling-cascade is an IND-CPA robust combiner, then all spilling primitives from the spilling-cascade chain respect both conditions 1 and 2.

Proof of Statement 1. Let us consider n encryption schemes $(\mathcal{E}_i)_{i\in[\![1,n]\!]}$ such that all $n-1$ upstream primitives $(\mathcal{E}_i)_{i\in[\![1,n-1]\!]}$ either are non-spilling or respect both condition 1 and condition 2. We additionally consider that only a single encryption scheme in the chain, with index $j \in [\![1,n]\!]$, is IND-CPA secure.

For any encryption scheme that yields a ciphertext of the form $(u,v) \leftarrow \mathrm{Enc}(pk,m)$ from a plaintext $m \in \mathcal{M}$, we note π_u and π_v the functions that respectively output u and v from that ciphertext: $u := \pi_u(\mathrm{Enc}(pk,m))$ and $v := \pi_v(\mathrm{Enc}(pk,m))$.

Without loss of generality, the ciphertext output by the spilling-cascade chain related to a plaintext $m \in \mathcal{M}$ can be written as:

[13] In other terms, the hybrid scheme is IND-CPA secure if at least anyone of its component primitives is IND-CPA secure as well.

$$\boxed{c_{sc} := \left((u_i)_{i\in[\![1,n-1]\!]}, c_n\right)} \tag{8}$$
with $v_0 := m$ and $\forall i \in [\![1,n]\!]\,,\ c_i := (u_i, v_i) \leftarrow \mathrm{Enc}_i(pk_i, v_{i-1})$ s.t. $\forall i \notin \mathcal{S}, u_i = \varnothing$.

Furthermore, we can express the ciphertext c_i of any encryption scheme located after the IND-CPA secure encryption scheme $\mathcal{E}_j$ in the spilling-cascade chain ($i > j$), in function of the encapsulated part v_j of that secure primitive:

$$\forall i \in [\![j+1,n]\!]\,, c_i \leftarrow \mathrm{Enc}_i(pk_i, \pi_v(\mathrm{Enc}_{i-1}(pk_{i-1}, \ldots, \pi_v(\mathrm{Enc}_{j+1}(pk_{j+1}, v_j))))) \tag{9}$$

Game 0. This is the experiment $\mathbf{Exp}^{\mathrm{IND-CPA}-0}_{spill-casc}$ of the spilling-cascade security game (cf. Figure 1). The ciphertext produced by the hybrid scheme from a plaintext m^0 is:

$$\begin{aligned}
&c_{sc} := \left((u_i^0)_1^{j-1}, u_j^0, (u_i^0)_{j+1}^{n-1}, c_n^0\right)\\
&\text{with}: \forall i \in [\![1,j-1]\!]\,, c_i^0 := (u_i^0, v_i^0) \leftarrow \mathrm{Enc}_i(pk_i, v_{i-1}^0) \quad \text{and} \quad v_0^0 := m^0\\
&\qquad (u_j^0, v_j^0) \leftarrow \mathrm{Enc}_j(pk_j, v_{j-1}^0)\\
&\qquad \forall i \in [\![j+1,n-1]\!]\,, u_i^0 := \pi_u(\mathrm{Enc}_i(pk_i, \ldots, \pi_v(\mathrm{Enc}_{j+1}(pk_{j+1}, v_j^0))))\\
&\qquad c_n^0 \leftarrow \mathrm{Enc}_n(pk_n, \ldots, \pi_v(\mathrm{Enc}_{j+1}(pk_{j+1}, v_j^0)))
\end{aligned} \tag{10}$$

Game 1. We replace in this game the term v_{j-1}^0 by the encapsulated part $\tilde{v}_{j-1}$ of the encryption of a random message $\tilde{m}_{j-1}$ from the message space of $\mathcal{E}_{j-1}$: $\tilde{v}_{j-1} := \pi_v(\mathrm{Enc}_{j-1}(pk_{j-1}, \tilde{m}_{j-1}))$, with $\tilde{m}_{j-1} \leftarrow\!\!\$\ \mathcal{M}_{j-1}$.

Thus, we have (with the boxes representing the values changed compared to the previous game):

$$\begin{aligned}
&c_{sc} := \left((u_i^0)_1^{j-1}, \boxed{\tilde{u}_j}, \boxed{(\tilde{u}_i)_{j+1}^{n-1}}, \boxed{\tilde{c}_n}\right)\\
&\text{with}: \forall i \in [\![1,j-1]\!]\,, c_i^0 := (u_i^0, v_i^0) \leftarrow \mathrm{Enc}_i(pk_i, v_{i-1}) \text{ and } v_0^0 := m^0\\
&\qquad (\boxed{\tilde{u}_j}, \boxed{\tilde{v}_j}) \leftarrow \mathrm{Enc}_j(pk_j, \boxed{\tilde{v}_{j-1}})\\
&\qquad \forall i \in [\![j+1,n-1]\!]\,, \boxed{\tilde{u}_i} := \pi_u(\mathrm{Enc}_i(pk_i, \ldots, \pi_v(Enc_{j+1}(pk_{j+1}, \boxed{\tilde{v}_j}))))\\
&\qquad \boxed{\tilde{c}_n} \leftarrow \mathrm{Enc}_n(pk_n, \ldots, \pi_v(\mathrm{Enc}_{j+1}(pk_{j+1}, \boxed{\tilde{v}_j})))
\end{aligned} \tag{11}$$

The advantage of any attacker $\mathcal{A}$ to distinguish $(\tilde{u}_j, \tilde{v}_j)$ from (u_j^0, v_j^0), i.e. to distinguish Game 0 from Game 1, is bounded by the IND-CPA security of the scheme $\mathcal{E}_j$: $\epsilon_1 \leq adv^{\mathrm{IND-CPA}}_{\mathcal{E}_j}$.

Proof. From any adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ trying to distinguish Games 0 and 1, we can build an adversary $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ against the CPA-indistinguishability of the encryption scheme $\mathcal{E}_j$, as described in Figure 4:

- $\mathcal{B}_1$ gets as input the public-key of $\mathcal{E}_j$, generates all other public keys from the spilling-cascade chain and computes the hybrid ciphertext associated with a random message m^0 up to the $(j-1)$-th ciphertext c_{j-1}^0. The adversary also randomly draws a message $\tilde{m}_{j-1} \leftarrow\!\!\$\ \mathcal{M}_{j-1}$ and encrypts it under $\mathcal{E}_{j-1}$ to get $\tilde{c}_{j-1}$. $\mathcal{B}_1$ then outputs v_{j-1}^0 and $\tilde{v}_{j-1}$.

– $\mathcal{B}_2$ runs on the challenge $(c_j^*, v_{j-1}^0,\ \tilde{v}_{j-1})$. Its computes the ciphertexts of the encryption schemes $\mathcal{E}_{j+1}$ to $\mathcal{E}_n$, all issued from $v_j^* := \pi_v(c_j^*)$. The hybrid ciphertext generated this way is given to the adversary $\mathcal{A}_2$ that outputs a guessing bit $\hat{b}$ that is also output by $\mathcal{B}_2$.

Consequently, $\mathcal{A}$'s advantage is bounded as follows:
$\epsilon_1 = adv^{G0-G1}(\mathcal{A}) \leq adv_{\mathcal{E}_j}^{\text{IND-CPA}}(\mathcal{B})$.

Adversary $\mathcal{B}_1(pk_j)$

$(pk_i, sk_i) \leftarrow \text{KeyGen}_i(1^\lambda)$
$v_0^0 := m^0 \leftarrow\!\$\ \mathcal{M}_1$
for $i \in [\![1, j-1]\!]$:
  $c_i^0 := (u_i^0, v_i^0) \leftarrow \text{Enc}_i(pk_i, v_{i-1}^0)$
$\tilde{m}_{j-1} \leftarrow\!\$\ \mathcal{M}_{j-1}$
$\tilde{v}_{j-1} := \pi_v\left(\text{Enc}_{j-1}(pk_{j-1}, \tilde{m}_{j-1})\right)$
$st \leftarrow \mathcal{A}_1\left((c_i^0)_1^{j-1}, (pk_i)_1^n\right)$
$st' := \left(st, (u_i^0)_1^{j-1}, (pk_i)_{i\in[\![1,n]\!]\setminus\{j\}}\right)$
return $v_{j-1}^0, \tilde{v}_{j-1}, st'$

Adversary $\mathcal{B}_2(pk_j, c_j^*, v_{j-1}^0, \tilde{v}_{j-1}, st')$

$(u_j^*, v_j^*) := \left(\pi_u(c_j^*), \pi_v(c_j^*)\right)$
for $i \in [\![j+1, n]\!]$:
  $c_i^* := (u_i^*, v_i^*) \leftarrow \text{Enc}_i(pk_i, v_{i-1}^*)$
$c^* := \left((u_i^0)_1^{j-1}, u_j^*, (u_i^*)_{j+1}^{n-1}, c_n^*\right)$
$\hat{b} \leftarrow \mathcal{A}_2\left(pk_j, c^*, v_{j-1}^0, \tilde{v}_{j-1}, st'\right)$
return $\hat{b}$

Fig. 4: Adversary $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ against the CPA-indistinguishability of the j-th encryption scheme $\mathcal{E}_j$ of a spilling-cascade, based on an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ against the indistinguishability of games G0 and G1.

Game 2. $\forall i \in [\![1, j-1]\!]$, we replace u_i^0 by u_i^1, which modifies the hybrid ciphertext as follows:

$$\begin{aligned} c_{sc} &:= \left(\boxed{(u_i^1)_1^{j-1}}, \tilde{u}_j, (\tilde{u}_i)_{j+1}^{n-1}, \tilde{c}_n\right) \\ &\text{with} : \forall i \in [\![1, j-1]\!], \boxed{u_i^1} := \pi_u(\text{Enc}_i(pk_i, \boxed{v_{i-1}^1})) \quad \text{and} \quad \boxed{v_0^1} := m^1 \end{aligned} \tag{12}$$

None of the u_i^0 and u_i^1 ($i \in [\![1, j-1]\!]$) are related in any way to the ciphertexts $\tilde{c}_k := (\tilde{u}_k, \tilde{v}_k) = \text{Enc}_k(pk_k, \tilde{v}_{k-1}),\ k \in [\![j, n]\!]$, since the latter *all* originate from the *random* element $\tilde{m}_{j-1}$ (through $\tilde{v}_{j-1}$).

Moreover, $\forall i \in [\![1, j-1]\!]$, u_i^0 and u_i^1 follow the exact same distribution independent of their associated plaintexts m_i^0 and m_i^1,

– either because of the "partitioned-ciphertext" property of the associated encryption scheme $\mathcal{E}_i$ and the fact that $c_{m_i} \sqsubseteq v_i$, which implies by definition that no information related to the plaintexts m_i^0 and m_i^1 is contained in u_i^0 and u_i^1,
– or because the scheme $\mathcal{E}_i$ is not spilling, and in that case $u_i^0 = u_i^1 = \varnothing$.

Consequently, the advantage of any attacker in distinguishing Game 2 from Game 1 is null: $\epsilon_2 = 0$.

Game 3. We replace in this game the term $\tilde{v}_{j-1}$ by v^1_{j-1}, which implies the following changes:

$$\begin{aligned} & c_{sc} := \left((u^1_i)^{j-1}_1, \boxed{u^1_j}, \boxed{(u^1_i)^{n-1}_{j+1}}, \boxed{c^1_n}\right) \\ & \text{with} : (\boxed{u^1_j}, \boxed{v^1_j}) \leftarrow \text{Enc}_j(pk_j, \boxed{v^1_{j-1}}) \\ & \qquad \forall i \in [\![j+1, n-1]\!], \boxed{u^1_i} := \pi_u(\text{Enc}_i(pk_i, \ldots, \pi_v(\text{Enc}_{j+1}(pk_{j+1}, \boxed{v^1_j})))) \\ & \qquad \boxed{c^1_n} \leftarrow \text{Enc}_n(pk_n, \ldots, \pi_v(\text{Enc}_{j+1}(pk_{j+1}, \boxed{v^1_j}))) \end{aligned} \tag{13}$$

Similarly to the passage from Game 0 to Game 1, and using the same type of adversary than in Figure 4, we can bound the advantage of any attacker trying to distinguish Game 3 from Game 2 by : $\epsilon_3 \leq adv^{\text{IND-CPA}}_{\mathcal{E}_j}$.

This game corresponds to the experiment $\mathbf{Exp}^{\text{IND-CPA-1}}_{spill-casc}$ of the spilling-cascade security game. Consequently, the security bound of the hybrid encryption scheme generated by a spilling-cascade combination of n primitives, with at least an IND-CPA secure component encryption scheme $\mathcal{E}_j$, is bounded by $adv^{\text{IND-CPA}}_{\mathcal{E}_{sc}} \leq 2 \cdot adv^{\text{IND-CPA}}_{\mathcal{E}_j}$, which completes the proof of statement 1.

Proof of Statement 2. Let us define the following logical proposals:

- **Robust**($\mathcal{SC}$): the spilling-cascade $\mathcal{SC}$ is an IND-CPA robust combiner;
- **IND-CPA**($\mathcal{E}$): the encryption scheme $\mathcal{E}$ is IND-CPA secure;
- **Cond1**($\mathcal{E}$): the encryption scheme $\mathcal{E}$ has a partitioned-ciphertext, therefore it respects the condition 1 expressed in the proof sketch of Section 3.3;
- **Cond2**($\mathcal{E}$): the encryption scheme $\mathcal{E}$ respects the condition 2 from that proof sketch by having the message-related part of its ciphertext encapsulated by the following encryption scheme: $c_{m_s} \sqsubseteq v_s$.

With $\mathcal{S}$ the spilling primitives in a spilling-cascade chain, statement 2 can be expressed as:

$$\mathbf{Robust}(\mathcal{SC}) \Rightarrow \ \forall \mathcal{E}_s \in \mathcal{S}, \mathbf{Cond1}(\mathcal{E}_s) \wedge \mathbf{Cond2}(\mathcal{E}_s) \tag{14}$$

$$\Leftrightarrow \ \exists \mathcal{E}_s \in \mathcal{S} : \neg\mathbf{Cond1}(\mathcal{E}_s) \vee \neg\mathbf{Cond2}(\mathcal{E}_s) \Rightarrow \ \neg\mathbf{Robust}(\mathcal{SC}) \tag{15}$$

As per Definition 5, a combiner for encryption schemes is said to be IND-CPA robust if for every possible set $\mathcal{C}_n = (\mathcal{E}_i)^n_1$ of n component encryption schemes – and for all values of $n \geq 2$ –, the IND-CPA security of any one of these schemes implies that the output hybrid encryption scheme $\mathcal{E}_{hy}$ is itself IND-CPA secure:

$$\mathbf{Robust} \Leftrightarrow (\forall n \geq 2, \forall \mathcal{C}_n, \ \forall \mathcal{E}_j \in \mathcal{C}_n, \mathbf{IND\text{-}CPA}(\mathcal{E}_j) \Rightarrow \ \mathbf{IND\text{-}CPA}(\mathcal{E}_{hy})) \tag{16}$$

$$\Rightarrow \neg\mathbf{Robust} \Leftrightarrow (\exists n \geq 2, \exists \mathcal{C}_n, \ \exists \mathcal{E}_j \in \mathcal{C}_n : \mathbf{IND\text{-}CPA}(\mathcal{E}_j) \wedge \neg\mathbf{IND\text{-}CPA}(\mathcal{E}_{hy})) \tag{17}$$

Combining Equation (15) and Equation (17), statement 2 is rephrased as:

$$\exists \mathcal{E}_s \in \mathcal{S} : \neg\mathbf{Cond1}(\mathcal{E}_s) \vee \neg\mathbf{Cond2}(\mathcal{E}_s) \Rightarrow \ \exists n \geq 2, \exists \mathcal{C}_n = (\mathcal{E}_i)^n_i, \exists \mathcal{E}_j \in \mathcal{C}_n : \mathbf{IND\text{-}CPA}(\mathcal{E}_j) \wedge \neg\mathbf{IND\text{-}CPA}(\mathcal{E}_{hy}) \tag{18}$$

Therefore, to prove Equation (18), we assume that its left-hand proposal is true and we need to find an instance of a set $\mathcal{C}_n$ of component encryption schemes, for a given number n of encryption schemes, that validates the right-hand proposal. We study separately the two cases $\exists \mathcal{E}_s \in \mathcal{S} : \neg\mathbf{Cond1}(\mathcal{E}_s)$ and $\exists \mathcal{E}_s \in \mathcal{S} : \neg\mathbf{Cond2}(\mathcal{E}_s)$.

*Spilling Primitive with Non-Partitioned-Ciphertext ($\neg$**Cond1**).* Let us consider a spilling-cascade combination of encryption schemes in which there is (at least) one spilling primitive with non-partitioned ciphertext. Then, one can always find at least one set of encryption schemes $\mathcal{C}_n$ in which one of the primitives is IND-CPA secure but not the output hybrid scheme.

For instance, we can set the number of component encryption schemes to $n = 2$, for simplicity's sake, and therefore combine together two schemes such that:

- The first scheme $\mathcal{E}_1$ is a spilling primitive wrapped around a (random) non-partitioned-ciphertext encryption scheme $\mathcal{E}'_1$, that only differs from the latter by adding, at the beginning of its ciphertext, the first bit b_{m_1} of its plaintext: $c_1 := (b_{m_1}, c'_1)$. Straightforwardly, such a scheme is also a non-partitioned-ciphertext scheme. We also choose that $\mathcal{E}_1$ includes that bit in the part of its ciphertext that is made public: $u_1 := (b_{m_1}, \bar{u}_1)$.
- The second scheme in the spilling-cascade chain is IND-CPA secure.

In such a case, any PPT adversary can win with probability 1 the security experiment on the CPA-indistinguishability of the spilling-cascade, by choosing two challenge plaintexts m^0 and m^1 starting respectively with 0 and 1, and by checking, in the challenge ciphertext $c^* := (u_1^*, c_2^*) = (b_{m_1}^*, \bar{u}_1^*, c_2^*)$ output by the spilling-cascade, to which plaintext corresponds the leaked bit $b_{m_1}^*$.

*Spilling Primitive with $c_m \not\sqsubseteq v$ ($\neg$**Cond2**).* In this case, the spilling-cascade combiner comprises a partitioned-ciphertext spilling primitive that does not entirely encapsulate the message-related part c_m of its ciphertext into the following encryption scheme. In the most conservative hypothesis, it comes to leaking a single bit b_m from that c_m part into the part of the ciphertext that is made public: $b_m \sqsubseteq u$.

Once again, we can straightforwardly find an instance of two encryption schemes such that:

- The first scheme $\mathcal{E}_1$ is a spilling primitive based on a (random) partitioned-ciphertext encryption scheme $\mathcal{E}'_1$, with the first bit b_{m_1} of its plaintext added at the beginning of the *public* part u_1 of its ciphertext: $u_1 := (b_{m_1}, u'_1)$. $\mathcal{E}_1$ is obviously also partitioned-ciphertext, with $c_{r_1} = c'_{r_1}$ and $c_{m_1} = (c'_{m_1}, b_{m_1})$.
- The second scheme in the spilling-cascade chain is IND-CPA secure.

This setting also leaks, in the hybrid ciphertext c^*, the bit b_{m_1} that was spilled by the scheme $\mathcal{E}_1$. This leakage is sufficient for the adversary to win with probability 1 the IND-CPA security experiment of the spilling-cascade scheme, with the same method as in the previous case.

This proves Statement 2 and terminates the proof of Theorem 1.

3.4 Properties of the Hybrid PKE

As recalled in Equation (5), the security bound of a FO transformation depends on the worst-case correctness δ^{wc} and the γ-spreadness of the underlying PKE. Therefore, it appears necessary to assess the value of these two properties for a hybrid PKE built from a spilling-cascade.

Lemma 1 (Hybrid Worst-Case Correctness and γ-Spreadness). *The combination in spilling-cascade of n encryption schemes $(\mathcal{E}_i = (\mathrm{KeyGen}_i, \mathrm{Enc}_i, \mathrm{Dec}_i))_{i\in[\![1,n]\!]}$, each of which being γ_i-spread and with a worst-case correctness δ_i^{wc}, yields a γ_{sc}-spread hybrid PKE with a worst-case correctness δ_{sc}^{wc} s.t.:*

$$\gamma_{sc} \geq \gamma_n \qquad \text{and} \qquad \delta_{sc}^{wc} \leq \sum\nolimits_{i=1}^{n} \delta_i^{wc} \tag{19}$$

This lemma, whose proof is located in Appendix A.2, underlines that a FO transformation applied on a spilling-cascade-built hybrid encryption scheme has a security bound that is similar to the ones of the FO transformations carried out on the component schemes. Indeed, the γ-spreadness – that we want as high as possible – is lower-bounded by the one of the last scheme in the spilling-cascade chain, while the worst-case correctness – which should be minimized – only grows linearly with the number of component schemes.

3.5 Functionalities of a Spilling-Cascade

We present hereunder two functionalities brought by the spilling-cascade combination of encryption schemes.

KEM Hybridization. As mentioned in the introduction, the spilling-cascade architecture can be used for KEM hybridization, in order to maintain the security of the hybrid encapsulation scheme even in case of failure of some of its components. A (spilling-)cascade KEM hybridization is realized by combining the PKEs of the KEMs we want to associate in order to generate a hybrid PKE, which in turn is transformed into a hybrid KEM with a FO transformation.

Authenticated Key Exchange with Spilling-Cascade. The spilling-cascade combination may also be used in the context of an authenticated key exchange, where the parties are implicitly authenticated by additional KEMs instead of signatures. This approach, stated for instance by [3], appears particularly interesting in the framework of post-quantum authentication, as most current PQ signatures still suffer from a larger bandwidth than their KEM counterparts. This observation has led to KEM-TLS [27], a variant of TLS where signatures are replaced by a KEM-based implicit authentication. KEM-TLS however uses a classical parallel combination of its KEMs, and therefore could be improved with the use of our spilling-cascade.

Figure 5 compares a standard implicit Unilateral Authenticated Key Exchange (UAKE) with a similar protocol using a spilling-cascade combiner.

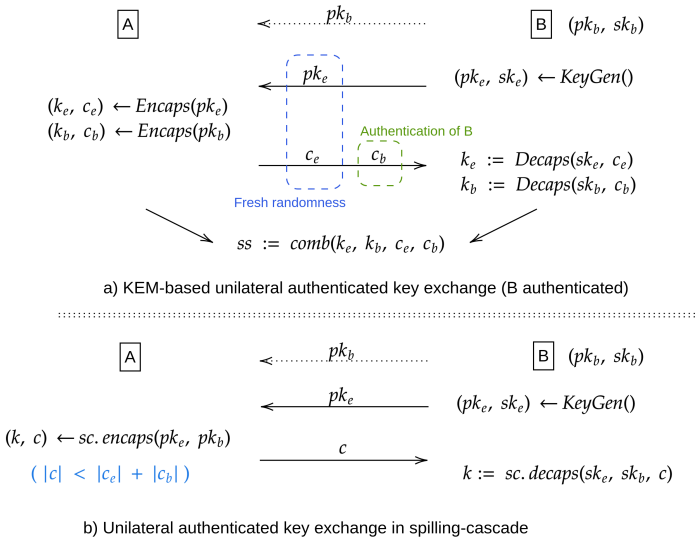

Fig. 5: KEM-based implicit UAKE, with authentication of party B. Figure *a* shows a standard authentication procedure, as proposed in [3]. Figure *b* details the variant with a spilling-cascade hybrid-KEM.

Here, the (spilling-)cascade architecture does no longer have to prevent the failure of some of the KEMs. It instead plays the role of a multi-key encryption scheme, for which several secret keys (the ephemeral one, used to refresh the randomness, as well as the recipient's static one, used as an authentication mean) are needed to decrypt a ciphertext. In this paradigm, one can rely on the same algorithm for the whole spilling-cascade. As a consequence, the initial constraint of Theorem 1, stating that the upstream PKE must be partitioned ciphertext, may be relaxed (since this property is only needed, in the security proof of the spilling-cascade, when the first encryption scheme is broken and the second one remains secure).

4 Instantiations

We now detail the instantiation of two hybrid schemes that we consider as particularly relevant in real world applications. The communication costs associated with these instantiations are studied in Section 5.

4.1 Spilling-Cascade Double-Hybridization

The first hybrid KEM consists in a $\{1\}$-spilling-cascade combination of the classical ElGamal cryptosystem (on elliptic curves) and the PKE of the PQ KEM Crystals Kyber. As ElGamal has partitioned ciphertexts (cf. proof in Appendix A.1), according to Theorem 1, the resulting hybrid PKE has IND-CPA security and the related KEM, after a FO transformation, is IND-CCA secure.

4.2 Spilling-Cascade Triple-Hybridization

The second hybrid KEM combiner, adapted to highly sensitive data, comes from a triple hybridization in {1,2}-spilling-cascade composition, with one classical encryption scheme and two post-quantum PKEs (cf. Figure 6). In this case, the choice of PQ algorithms is strongly constrained by the ciphertext length of the upstream PQ PKE and the input size of the downstream PQ PKE.

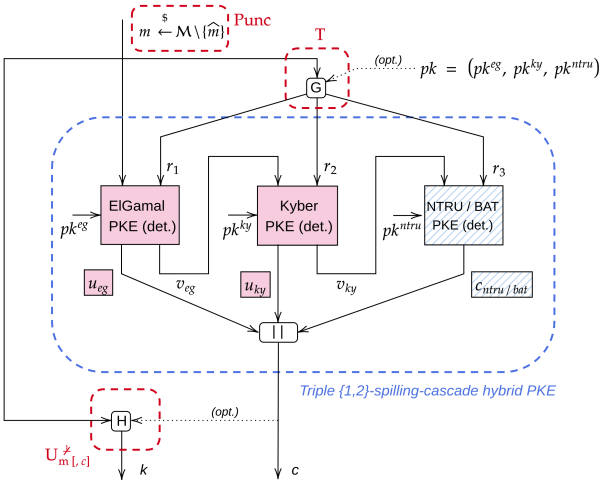

Fig. 6: Architecture of the triple hybridization in {1,2}-spilling-cascade ElGamal-Kyber-NTRU/BAT. The spilling PKEs, here both ElGamal and Kyber, are depicted in pink color. The dashed blue box represents the hybrid PKE and the dashed red boxes, the modular FO transformation.

In our instantiation, the classical algorithm and the upstream PQ PKE are the same as in the double hybridization: ElGamal and Kyber. The security of the combiner is ensured by the fact that Kyber, which is a spilling primitive in the scheme along with ElGamal, is also a partitioned-ciphertext encryption scheme (cf. proof in Appendix A.1).

Regarding the downstream PQ algorithm, we consider two candidates both based on NTRU lattices: NTRU-HRSS [20] and the more recent algorithm BAT [11]. We also implement a variant of NTRU-HRSS proposed by [25], that we call npNTRU. This one is based on a tweak by [23] where the probabilistic NTRU-HRSS PKE is turned into a deterministic one by including the randomness r into the input of the PKE, along with the plaintext m (since it is possible to recover r as well in the decryption process). This tweak therefore increases the input size of this PKE, which is beneficial for the efficiency of the spilling-cascade..

The criteria considered for these choices were:

– The good performances of these algorithms, that make them realistic alternatives to Kyber in case this one would be broken.

– Their differences with Kyber in terms of mathematical foundations, so that a vulnerability on Kyber or even on the Module-Learning with Error problem would not necessarily compromise them.
– More importantly, the input lengths[14] of their PKEs, of 140 bytes for NTRU-HRSS, 160 bytes for BAT and 280 bytes for npNTRU, permit the encapsulation of the 128 byte-long c_m part of Kyber's ciphertext ($c_{m_{ky}} \sqsubseteq v_{ky}$), which is the necessary condition to implement a secure spilling-cascade combination of these KEMs.

Nota: NTRU-HRSS PKE takes as input a bit string that is encoded afterwards into a ternary polynomial. Because the original encoding algorithm does not permit to recover, in the decryption process, an arbitrary input given to the PKE (which is not a problem when NTRU is only used as a KEM), we had to slightly modify this encoding algorithm, as presented in the full version.

4.3 Enhanced Variations

We study two possible enhancements of our spilling-cascade KEM hybridization, that are detailed in the full version.

Integrated Diffie-Hellman (IDH) KEM. This is a spilling-cascade combination of one or several PQ PKEs with a Diffie-Hellman on Elliptic Curve (ECDH) key agreement scheme, in replacement of the classical encryption scheme. The idea here is to be more efficient in terms of bandwidth, given the fact that an ECDH key agreement scheme produces a ciphertext[15] lighter than a similar encryption algorithm (twice smaller, for instance, than its ElGamal counterpart).

Integration of the Public Keys. Another enhancement consists in the encapsulation in one another of the public keys of the PKEs to be hybridized, in order to gain once again some bandwidth. We provide here a method to encapsulate an ECDH or ElGamal public key in Kyber's one. This proposal is unfortunately not generically applicable to all classical or PQ algorithms, since it depends on the structure of the wrapping public key. However, the fact that Kyber and the ECDH schemes are the most commonly used key exchange primitives, respectively in the classical and Post-Quantum settings, mitigates this drawback.

5 Communication Cost

Even if the practical communication cost of the spilling-cascade's ciphertext relies on the characteristics of the encryption schemes used in the chain, we state it constitutes the *optimal* generic hybridization method of PKEs regarding the communication cost of the output ciphertext.

[14] For the parameters corresponding to the NIST security level 3.
[15] We model here the "ciphertext" of an ECDH key agreement scheme by the public DH element output by the sender. See the full version for additional details.

5.1 Communication Cost of the Spilling-Cascade

Definition 9 (Expansion Rate of an Encryption Scheme). *Let us consider an encryption scheme $\mathcal{E}$ that, for a given set of parameters depending on the security parameter λ, has a message space $\mathcal{M}$ with elements of bit-length $|m_{\mathcal{E}}|_\lambda$ and outputs ciphertexts of bit-length $|c_{\mathcal{E}}|_\lambda$. We define the expansion rate of $\mathcal{E}$ for the security parameter λ[16] as the ratio:*

$$\boxed{\tau_{\mathcal{E},\lambda} = \frac{|c_{\mathcal{E}}|_\lambda}{|m_{\mathcal{E}}|_\lambda}} \tag{20}$$

Spilling-Cascade Ciphertext Bandwidth. Let $(\mathcal{E}_i)_{i\in[\![1,n]\!]}$ be n encryption schemes with respective input size, output size and expansion rate $|m_i|$, $|c_i|$ and τ_i. Let us combine these schemes in spilling-cascade, with an arbitrary subset $\mathcal{S}$ of the $n-1$ first primitives spilling a part of their ciphertext. As in Definition 6, we note u_i the spilling (i.e. public) part of $\mathcal{E}_i$'s ciphertext and v_i the part of its ciphertext that is encapsulated by $\mathcal{E}_{i+1}$ (if the scheme is not spilling, then $u_i = \varnothing$ and $v_i = c_i$).

The spilling predicate s_i, associated with an encryption scheme $\mathcal{E}_i$ in the spilling-cascade chain, is defined as: $s_0 = 0$, $s_n = 1$ and $\forall i \in [\![1, n-1]\!]$, $s_i = 1$ if $\mathcal{E}_i \in \mathcal{S}$ and 0 otherwise. The ciphertext size of the spilling-cascade combination of that schemes is:

$$|c_{sc}| = \sum\nolimits_{i=1}^{n-1} \big(s_i\,|u_i|\big) + |c_n| = \sum\nolimits_{i=1}^{n-1} s_i\big(\,|c_i| - |v_i|\,\big) + s_n\,|c_n| \tag{21}$$

For non-spilling schemes $\mathcal{E}_i \notin \mathcal{S}$, $s_i\,|u_i| = 0$. Moreover, according to Definition 6, spilling primitives $\mathcal{E}_j \in \mathcal{S}$ have an encapsulated ciphertext of the same bit-length as the elements of the message space of the following schemes: $|v_j| = |m_{j+1}|$. Consequently, $s_j(|c_j| - |v_j|) = s_j(\tau_j\,|m_j| - |m_{j+1}|)$ and:

$$\boxed{|c_{sc}| = \sum\nolimits_{i=1}^{n} (s_i\tau_i - s_{i-1})\,|m_i|} \tag{22}$$

Nota: In comparison, the ciphertext bandwidth of the parallel combination of that same encryption schemes is: $|c_{par}| = \sum_{i=1}^{n} \tau_i\,|m_i|$.

5.2 Optimality of the Ciphertext Size

We define below the concept of encryption scheme with *bare ciphertext* that aims to exclude, in our Theorem 2, any cryptosystem where part of the ciphertext (padding or error correcting mechanism, for instance) is not required for decryption. We also exclude from our analysis, through the concept of *efficiency*, combiners for encryption schemes that artificially decrease their bandwidth by truncating the ciphertext they output. In practice, to the best of our knowledge, all real-life cryptosystems have a bare ciphertext and all combiners for encryption schemes are efficient. Therefore, these two assumptions do not narrow the scope of Theorem 2.

[16] The security parameter λ may be omitted when it is well-defined and remains unchanged in a given context, which is the case in this paper.

Definition 10 (Encryption Scheme with Bare Ciphertext). *An encryption scheme $\mathcal{E}$ is said to have a bare ciphertext if any bit of the ciphertext c it outputs is necessary to successfully run its decryption algorithm* Dec.

Theorem 2 (Optimal Communication Cost). *For all sets $(\mathrm{PKE}_i)_{i=1}^{n}$ of n public-key cryptosystems with bare ciphertexts that are instantiable with a spilling-cascade, there exists no efficient, IND-CPA robust, combiner for PKEs that is compatible with these sets and that outputs a ciphertext smaller than that of the spilling-cascade, provided that one of the two following conditions applies:*

1. *This combination is fully-spilling: $\mathcal{S} = (\mathcal{E}_i)_{i=1}^{n-1}$ (cf. Definition 7);*
2. *Any non-spilling encryption scheme outputs ciphertexts of bit-length equal to the input size of the following primitive: $\forall \mathcal{E}_j \notin \mathcal{S}, |c_j| = |m_{j+1}|$.*

Nota: This theorem states the optimality of the spilling-cascade as a combination method generically applicable to all sets of PKEs respecting the security conditions from Theorem 1. On the contrary, we cannot exclude that there exists specific sets of PKEs that can be combined in a way that yields a ciphertext smaller than with the spilling-cascade. However, such combinations would lack the genericity that we require, and thus are not considered in Theorem 2.

Proof of Theorem 2. We proceed with a proof by contradiction. Let us assume that there exists an efficient IND-CPA robust combiner for PKEs $\mathscr{C}_{opt}$ (in the sense of Definition 5) that is compatible with any set of PKEs $(\mathrm{PKE}_i)_{i\in[\![1,n]\!]}$ instantiable with the spilling-cascade and that outputs a ciphertext smaller than that of the latter (cf. Equation (22)). Let us assume moreover that one of the two conditions from Theorem 2 is respected: either all the $n-1$ first schemes are spilling, when combined in spilling-cascade, or the non-spilling schemes do not need paddings to fit in the message space of the following encryption schemes.

$$|c_{opt}| \;<\; |c_{sc}| \;=\; \sum\nolimits_{i=1}^{n} (s_i\tau_i - s_{i-1})\,|m_i| \;=\; \sum\nolimits_{i=1}^{n-1} s_i\,(|c_i| - |m_{i+1}|) + |c_n| \quad (23)$$

As a reminder, the spilling predicate s is defined as: $s_0 = 0$, $s_n = 1$ and $\forall i \in [\![1, n-1]\!]\,, s_i = 1$ if $\mathcal{E}_i \in \mathcal{S}$ (spilling primitive) and 0 otherwise.

We now wonder whether the data comprised in $\mathscr{C}_{opt}$'s ciphertext is sufficient to recover the original plaintext m that was encrypted by $\mathscr{C}_{opt}$. Firstly, we prove below that the plaintext m encrypted by any IND-CPA robust combiner $\mathscr{C}$ can be recovered only if all its underlying decryption algorithms $(\mathrm{Dec}_i)_{i\in[\![1,n]\!]}$ are run successfully by the decryption algorithm $\mathrm{Dec}_{\mathscr{C}}$ of the hybrid scheme.

Proposition 1. *If a combiner $\mathscr{C}$ for PKEs is IND-CPA robust, in the sense of Definition 5, then its decryption algorithm $\mathrm{Dec}_{\mathscr{C}}$ runs the decryption algorithms Dec_i of* all *its component PKEs $(\mathrm{PKE}_i)_{i\in[\![1,n]\!]}$.*

Proof. We also prove this proposition by contradiction. Let us consider a combiner for PKEs $\mathscr{C}$ that is IND-CPA robust and that has a decryption algorithm $\mathrm{Dec}_{\mathscr{C}}$ that does not run the decryption algorithm of PKE_j – but it may use the $(\mathrm{KeyGen}_i)_{i\in[\![1,n]\!]}$ and $(\mathrm{Enc}_i)_{i\in[\![1,n]\!]}$ algorithms if needed, including those of the j-th scheme.

According to Definition 5, the hybrid PKE generated by the combiner must remain IND-CPA secure in the case where PKE_j is IND-CPA secure and all the other schemes are broken by the adversary. However, in such a case, the adversary appears able to break the IND-CPA security of the hybrid scheme, since the only IND-CPA secure component scheme PKE_j does not have its decryption algorithm Dec_j involved in the decryption process of the combiner. We note that the Enc_j algorithm – that is potentially run by $\text{Dec}_{\mathscr{C}}$ – does not secure the hybrid scheme because it uses as input a public key, known by the adversary[17]. The hybrid PKE is therefore not IND-CPA secure, which contradicts the initial hypothesis and proves that an IND-CPA robust combiner must run all its component decryption algorithms.

Focus on the Decryption Process of $\mathscr{C}_{opt}$. The decryption algorithm of the combiner $\mathscr{C}_{opt}$ can run its component decryption algorithms either in parallel or sequentially. During that process, each algorithm Dec_i "consumes" a ciphertext of size $|c_i|$ from the hybrid ciphertext c_{opt} and outputs a plaintext m'_i of size $|m_i|$.

In the general case, this plaintext m'_i can be used as additional data to "feed" another decryption algorithm from the combiner; this is however possible only when the decryptions occur sequentially. Therefore, we consider in the rest of that proof the more conservative hypothesis of sequential decryptions within the combiner $\mathscr{C}_{opt}$, and without loss of generality, we consider a decryption order from Dec_n to Dec_1.

Consequently, after running the $n-1$ first decryption algorithms, we have consumed $\sum_{i=2}^{n} \big(|c_i| - |m_i| \big)$ data, which leaves available, for the last decryption algorithm Dec_1, data of size α s.t.:

$$\begin{aligned}
\alpha &= |c_{opt}| - \sum\nolimits_{i=2}^{n} \big(|c_i| - |m_i| \big) \\
&< |c_{sc}| - \sum\nolimits_{i=2}^{n} \big(|c_i| - |m_i| \big) \\
&< \sum\nolimits_{i=1}^{n-1} s_i \big(|c_i| - |m_{i+1}| \big) + |c_n| - \sum\nolimits_{i=2}^{n} \big(|c_i| - |m_i| \big) \\
&< s_1(|c_1| - |m_2|) + \sum\nolimits_{i=2}^{n-1} \big(\underbrace{s_i(|c_i| - |m_{i+1}|) + |m_i| - |c_i|}_{\alpha_i} \big) + |m_n| \qquad (24)
\end{aligned}$$

As any existing non-spilling scheme has a ciphertext size $|c_i| = |m_{i+1}|$ (due to condition 2 of Theorem 2), the associated $\alpha_i = |m_i| - |c_i| = |m_i| - |m_{i+1}|$ appears identical to the value of that term for spilling schemes. Therefore:

$$\alpha < s_1(|c_1| - |m_2|) + |m_2| < |c_1| \qquad (25)$$

Because PKE_1 has a bare ciphertext – as stated as a prerequisite of Theorem 2 –, its decryption algorithm Dec_1 needs an input of size at least $|c_1|$. As shown in Equation (25), the data available for Dec_1 is therefore too small to successfully retrieve the plaintext m_1 that was encrypted with Enc_1. Thus, as

[17] This is a major difference with a combiner for secret-key cryptosystems, that can use the encryption algorithms of its component schemes even for the decryption process.

the combiner $\mathscr{C}_{opt}$ needs all its component decryption algorithms to work successfully in order to be IND-CPA robust, the failure of Dec_1 makes impossible to recover the original plaintext m and the combiner $\mathscr{C}$ turns out to be non-functional. This indicates that the ciphertext output by the spilling-cascade is the smallest possible for an efficient IND-CPA robust combiner for PKEs with bare ciphertexts, which terminates the proof of Theorem 2.

5.3 Practical Efficiency.

In our instantiations of KEM hybridization (cf. Figure 7), the best optimization brought by our spilling-cascade scheme exceeds 13 %, in the case of the triple hybridization ElGamal_Kyber_npNTRU, with a reduction of 306 bytes in the output ciphertext.

	Hybridization	Parallel comb. |ct| (bytes)	Spill-casc. comb. |ct| (bytes)	Gain of spill-casc.
KEM hybridization	ElGamal_Kyber	1,152	1,120	2.8 %
	ElGamal_BAT	1,070	1,006	6.0 %
	ElGamal_Kyber_NTRU	2,290	2,121	7.4 %
	ElGamal_Kyber_BAT	2,158	1,966	8.9 %
	ElGamal_Kyber_npNTRU	2,290	1,984	13.4 %
Unilateral Authenticated Key Echange	Kyber_Kyber	2,176	2,144	1.5 %
	NTRU_NTRU	2,276	2,139	6.0 %
	BAT_BAT	2,012	1,852	8.0 %
	npNTRU_npNTRU	2,276	2,002	12.0 %

Fig. 7: Ciphertext communication costs of parallel and spilling-cascade combinations for different types of hybridizations and KEM-based authentications.

Even though its bandwidth improvement remains moderate, the spilling-cascade combination offers interesting perspectives for lighter constructions in the future, as its performances closely depend on the primitives involved, which may appear in the future more appropriate to such a combination. Furthermore, even a small decrease in the size of the data transmitted may avoid reaching the maximum transmission size of a packet in a network, and thus its fragmentation.

References

1. Aguilar-Melchor, C., Aragon, N., Bettaieb, S., Bidoux, L., Blazy, O., Deneuville, J.C., Gaborit, P., Persichetti, E., Zémor, G., Bos, J., Dion, A., Lacan, J., Robert, J.M., Veron, P.: HQC. Tech. rep., National Institute of Standards and Technology (2022), available at https://csrc.nist.gov/Projects/post-quantum-cryptography/round-4-submissions
2. An, J.H., Dodis, Y., Rabin, T.: On the security of joint signature and encryption. In: Knudsen, L.R. (ed.) Advances in Cryptology – EUROCRYPT 2002. Lecture Notes in Computer Science, vol. 2332, pp. 83–107. Springer Berlin Heidelberg, Germany, Amsterdam, The Netherlands (Apr 28 – May 2, 2002). https://doi.org/10.1007/3-540-46035-7_6
3. Bos, J., Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schanck, J.M., Schwabe, P., Stehlé, D.: CRYSTALS – Kyber: a CCA-secure module-lattice-based KEM. Cryptology ePrint Archive, Report 2017/634 (2017), https://eprint.iacr.org/2017/634
4. Canetti, R., Krawczyk, H., Nielsen, J.B.: Relaxing chosen-ciphertext security. In: Boneh, D. (ed.) Advances in Cryptology – CRYPTO 2003. Lecture Notes in Computer Science, vol. 2729, pp. 565–582. Springer Berlin Heidelberg, Germany, Santa Barbara, CA, USA (Aug 17–21, 2003). https://doi.org/10.1007/978-3-540-45146-4_33
5. Chen, C., Danba, O., Hoffstein, J., Hulsing, A., Rijneveld, J., Schanck, J.M., Schwabe, P., Whyte, W., Zhang, Z., Saito, T., Yamakawa, T., Xagawa, K.: NTRU. Tech. rep., National Institute of Standards and Technology (2020), available at https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions
6. D'Anvers, J.P., Karmakar, A., Roy, S.S., Vercauteren, F., Mera, J.M.B., Van Beirendonck, M., Basso, A.: SABER. Tech. rep., National Institute of Standards and Technology (2020), available at https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions
7. Dodis, Y., Katz, J.: Chosen-ciphertext security of multiple encryption. In: Kilian, J. (ed.) TCC 2005: 2nd Theory of Cryptography Conference. Lecture Notes in Computer Science, vol. 3378, pp. 188–209. Springer Berlin Heidelberg, Germany, Cambridge, MA, USA (Feb 10–12, 2005). https://doi.org/10.1007/978-3-540-30576-7_11
8. Duman, J., Hövelmanns, K., Kiltz, E., Lyubashevsky, V., Seiler, G.: Faster lattice-based KEMs via a generic fujisaki-okamoto transform using prefix hashing. In: Vigna, G., Shi, E. (eds.) ACM CCS 2021: 28th Conference on Computer and Communications Security. pp. 2722–2737. ACM Press, Virtual Event, Republic of Korea (Nov 15–19, 2021). https://doi.org/10.1145/3460120.3484819
9. ElGamal, T.: A public-key cryptosystem and a signature scheme based on discrete logarithms. In: Blakley, G.R., Chaum, D. (eds.) Advances in Cryptology. pp. 10–18. Springer Berlin Heidelberg, Berlin, Heidelberg (1985)
10. Even, S., Goldreich, O.: On the power of cascade ciphers. ACM Tras. Computer Systems, 3:108–116 (1985)
11. Fouque, P.A., Kirchner, P., Pornin, T., Yu, Y.: BAT: Small and fast KEM over NTRU lattices. IACR Transactions on Cryptographic Hardware and Embedded Systems **2022**(2), 240–265 (2022). https://doi.org/10.46586/tches.v2022.i2.240-265

12. Fujisaki, E., Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. In: Wiener, M.J. (ed.) Advances in Cryptology – CRYPTO'99. Lecture Notes in Computer Science, vol. 1666, pp. 537–554. Springer Berlin Heidelberg, Germany, Santa Barbara, CA, USA (Aug 15–19, 1999). https://doi.org/10.1007/3-540-48405-1_34
13. Giacon, F., Heuer, F., Poettering, B.: KEM combiners. In: Abdalla, M., Dahab, R. (eds.) PKC 2018: 21st International Conference on Theory and Practice of Public Key Cryptography, Part I. Lecture Notes in Computer Science, vol. 10769, pp. 190–218. Springer, Cham, Switzerland, Rio de Janeiro, Brazil (Mar 25–29, 2018). https://doi.org/10.1007/978-3-319-76578-5_7
14. Goldreich, O., Lustig, Y., Naor, M.: On chosen ciphertext security of multiple encryptions. Cryptology ePrint Archive, Report 2002/089 (2002), https://eprint.iacr.org/2002/089
15. Herzberg, A.: Folklore, practice and theory of robust combiners. Cryptology ePrint Archive, Report 2002/135 (2002), https://eprint.iacr.org/2002/135
16. Hofheinz, D., Hövelmanns, K., Kiltz, E.: A modular analysis of the Fujisaki-Okamoto transformation. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017: 15th Theory of Cryptography Conference, Part I. Lecture Notes in Computer Science, vol. 10677, pp. 341–371. Springer, Cham, Switzerland, Baltimore, MD, USA (Nov 12–15, 2017). https://doi.org/10.1007/978-3-319-70500-2_12
17. Hohenberger, S., Lewko, A.B., Waters, B.: Detecting dangerous queries: A new approach for chosen ciphertext security. In: Pointcheval, D., Johansson, T. (eds.) Advances in Cryptology – EUROCRYPT 2012. Lecture Notes in Computer Science, vol. 7237, pp. 663–681. Springer Berlin Heidelberg, Germany, Cambridge, UK (Apr 15–19, 2012). https://doi.org/10.1007/978-3-642-29011-4_39
18. Hövelmanns, K., Kiltz, E., Schäge, S., Unruh, D.: Generic authenticated key exchange in the quantum random oracle model. In: Kiayias, A., Kohlweiss, M., Wallden, P., Zikas, V. (eds.) PKC 2020: 23rd International Conference on Theory and Practice of Public Key Cryptography, Part II. Lecture Notes in Computer Science, vol. 12111, pp. 389–422. Springer, Cham, Switzerland, Edinburgh, UK (May 4–7, 2020). https://doi.org/10.1007/978-3-030-45388-6_14
19. Huguenin-Dumittan, L., Vaudenay, S.: FO-like combiners and hybrid post-quantum cryptography. In: Conti, M., Stevens, M., Krenn, S. (eds.) CANS 21: 20th International Conference on Cryptology and Network Security. Lecture Notes in Computer Science, vol. 13099, pp. 225–244. Springer, Cham, Switzerland, Vienna, Austria (Dec 13–15, 2021). https://doi.org/10.1007/978-3-030-92548-2_12
20. Hülsing, A., Rijneveld, J., Schanck, J.M., Schwabe, P.: High-speed key encapsulation from NTRU. In: Fischer, W., Homma, N. (eds.) Cryptographic Hardware and Embedded Systems – CHES 2017. Lecture Notes in Computer Science, vol. 10529, pp. 232–252. Springer, Cham, Switzerland, Taipei, Taiwan (Sep 25–28, 2017). https://doi.org/10.1007/978-3-319-66787-4_12
21. Maurer, U.M., Massey, J.L.: Cascade ciphers: The importance of being first. Journal of Cryptology **6**, 55–61 (1993), https://doi.org/10.1007/BF02620231
22. Naehrig, M., Alkim, E., Bos, J., Ducas, L., Easterbrook, K., LaMacchia, B., Longa, P., Mironov, I., Nikolaenko, V., Peikert, C., Raghunathan, A., Stebila, D.: FrodoKEM. Tech. rep., National Institute of Standards and Technology (2020), available at https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions
23. Pipher, J.H.J., Silverman, J.H.: Ntru: A new high speed public-key cryptosystem. In: draft from at CRYPTO 96 rump session (1996), http://web.securityinnovation.com/hubfs/files/ntru-orig.pdf

24. Pöppelmann, T., Alkim, E., Avanzi, R., Bos, J., Ducas, L., de la Piedra, A., Schwabe, P., Stebila, D., Albrecht, M.R., Orsini, E., Osheter, V., Paterson, K.G., Peer, G., Smart, N.P.: NewHope. Tech. rep., National Institute of Standards and Technology (2019), available at https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-2-submissions
25. Saito, T., Xagawa, K., Yamakawa, T.: Tightly-secure key-encapsulation mechanism in the quantum random oracle model. In: Nielsen, J.B., Rijmen, V. (eds.) Advances in Cryptology – EUROCRYPT 2018, Part III. Lecture Notes in Computer Science, vol. 10822, pp. 520–551. Springer, Cham, Switzerland, Tel Aviv, Israel (Apr 29 – May 3, 2018). https://doi.org/10.1007/978-3-319-78372-7_17
26. Schwabe, P., Avanzi, R., Bos, J., Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schanck, J.M., Seiler, G., Stehlé, D., Ding, J.: CRYSTALS-KYBER. Tech. rep., National Institute of Standards and Technology (2022), available at https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022
27. Schwabe, P., Stebila, D., Wiggers, T.: Post-quantum TLS without handshake signatures. In: Ligatti, J., Ou, X., Katz, J., Vigna, G. (eds.) ACM CCS 2020: 27th Conference on Computer and Communications Security. pp. 1461–1480. ACM Press, Virtual Event, USA (Nov 9–13, 2020). https://doi.org/10.1145/3372297.3423350
28. Targhi, E.E., Unruh, D.: Post-quantum security of the Fujisaki-Okamoto and OAEP transforms. In: Hirt, M., Smith, A.D. (eds.) TCC 2016-B: 14th Theory of Cryptography Conference, Part II. Lecture Notes in Computer Science, vol. 9986, pp. 192–216. Springer Berlin Heidelberg, Germany, Beijing, China (Oct 31 – Nov 3, 2016). https://doi.org/10.1007/978-3-662-53644-5_8
29. Zhang, C., Cash, D., Wang, X., Yu, X., Chow, S.S.M.: Combiners for chosen-ciphertext security. In: Dinh, T.N., Thai, M.T. (eds.) Computing and Combinatorics. pp. 257–268. Springer International Publishing, Cham (2016)
30. Zhang, R., Hanaoka, G., Shikata, J., Imai, H.: On the security of multiple encryption or CCA-security+CCA-security=CCA-security? In: Bao, F., Deng, R., Zhou, J. (eds.) PKC 2004: 7th International Workshop on Theory and Practice in Public Key Cryptography. Lecture Notes in Computer Science, vol. 2947, pp. 360–374. Springer Berlin Heidelberg, Germany, Singapore (Mar 1–4, 2004). https://doi.org/10.1007/978-3-540-24632-9_26

A Omitted Proofs

A.1 Proofs for Partitioned-Ciphertext Algorithms

We detail hereunder the proofs of the partitioned-ciphertext property of ElGamal and Kyber encryption schemes, used in the practical instantiations of Section 4.

Proposition 2. *ElGamal public-key encryption algorithm [9] is a partitioned-ciphertext encryption scheme.*

Proof. We recall that in a cyclic group G of order q, with a generator g, an ElGamal ciphertext corresponding to a plaintext $m \in \mathcal{M} = G$ if of the form: $\boxed{c_{eg} := (c_{r_{eg}} = g^r, c_m = m.X^r)}$, where $X := g^x$ is the recipient's public-key and $r \leftarrow\!\!\$\ \mathbb{Z}_q$ a secret random element drawn by the sender.

For a given set of parameters $\{G, q, g\}$, the sizes of the components $c_{r_{eg}}$ and $c_{m_{eg}}$ clearly remain constant for all messages $(m, m') \in \mathcal{M}^2$ (these messages are such that $|m| = |m'|$, since every plaintext is an element of the group G). Furthermore, $c_{r_{eg}}$ does not depend on the encrypted plaintext m but only on the randomness r, which completes the proof.

Proposition 3. *Crystals Kyber public-key encryption algorithm [3] is a partitioned-ciphertext encryption scheme.*

Proof. The ciphertext output by Kyber is of the form $\mathbf{c}_{kyber} := (\mathbf{u}, v)$, with

$$\boxed{\begin{array}{l} c_{r_{ky}} := \mathbf{u} = \text{Compress}(A^T.\mathbf{r} + \mathbf{e_1}) \\ c_{m_{ky}} := v = \text{Compress}(\mathbf{t}^T.\mathbf{r} + e_2 + \left\lfloor \frac{q}{2} \right\rceil .m) \end{array}}$$

- The modulus $q = 3329$ is a fixed public parameter;
- $\mathbf{t}$ and $A = \text{Expand}(\rho)$ are issued from the recipient's public-key $\boxed{pk := \mathbf{t} \mid\mid \rho}$ (with $\rho \leftarrow\!\!\$\ \{0,1\}^{256}$);
- $\mathbf{e_1}$, e_2 and $\mathbf{r}$ are sampled from a secret seed $r \leftarrow\!\!\$\ \{0,1\}^{256}$.

As any semantically-secure encryption scheme, Kyber is "length-uniform" (cf. [15]), which implies that:
$\forall (pk, pk') \in \mathcal{PK}^2, \forall (m, m') \in \mathcal{M}^2, |m| = |m'| \Rightarrow |\text{Enc}(pk, m)| = |\text{Enc}(pk', m')|$.
Furthermore, we also have a message space with elements of constant size (32 bytes). Consequently: $\exists \kappa \in \mathbb{N} : \forall m \in \mathcal{M}, |c_{r_{ky}}| + |c_{m_{ky}}| = \kappa$.

Furthermore, the element $c_{r_{ky}}$ does not depend on the plaintext m: it is a function of the recipient's public key (since the matrix A directly comes from the random seed $\rho \in pk$) and of the random seed r used to deterministically produce $\mathbf{r}$, $\mathbf{e_1}$ and e_2. It thus fulfills condition 2 of Definition 8 and implies that $|c_{r_{ky}}|$ is constant over $\mathcal{M}$. According to the previous paragraph, we can deduce that $|c_{m_{ky}}|$ is constant over $\mathcal{M}$ as well, which terminates the proof of the theorem.

A.2 Proof of Worst-Case Correctness and γ-Spreadness

The proof of Lemma 1 is realized in the framework of a double PKE combination. The general case of n combined schemes is straightforwardly deduced by induction.

Let us consider the hybrid PKE PKE_{sc} resulting from the combination in $\{1\}$-spilling-cascade of component PKEs PKE_1 and PKE_2 The hybrid key-pair yielded by the hybrid key generation algorithm is simply the tuple of key-pairs of the underlying PKEs: $(pk_{sc}, sk_{sc}) := ((pk_1, pk_2), (sk_1, sk_2))$.

Following the previous notations of this paper, we call u_1 the element of the ciphertext $c_1 := (u_1, v_1)$ output by Enc_1 that is part of the hybrid ciphertext, and v_1 the part of c_1 that is encapsulated by PKE_2 in the spilling-cascade chain. Furthermore, let π_u and π_v be the functions that output respectively u_1 and v_1 from the encryption of a plaintext $m_1 \in \mathcal{M}_1$: $u_1 := \pi_u(\text{Enc}_1(pk_1, m; r_1))$ and $v_1 := \pi_v(\text{Enc}_1(pk_1, m; r_1))$. The sets of all possible values u_1 and v_1 for a given plaintext m from the message space of PKE_1 are called $\mathcal{U}_1^m$ and $\mathcal{V}_1^m$:

$$\begin{cases} \mathcal{U}_1^m := \{u_1 := \pi_u(\text{Enc}_1(pk_1, m; r_1)), r_1 \in \mathcal{R}_1\}_{m \in \mathcal{M}_1} \\ \mathcal{V}_1^m := \{v_1 := \pi_v(\text{Enc}_1(pk_1, m; r_1)), r_1 \in \mathcal{R}_1\}_{m \in \mathcal{M}_1} \end{cases}$$

We proceed separately to the proofs of the two properties of γ-spreadness and worst-case correctness.

Proof of the γ-spreadness. As PKE_2 is γ_2-spread, $\forall (pk_2, sk_2) \leftarrow \text{KeyGen}_2(1^\lambda)$, $\forall m_2 \in \mathcal{M}_2$, $\alpha_2^{m_2} = \underset{c_2 \in \mathcal{C}_2}{\text{Max}} \left\{ \Pr_{r_2} [\text{Enc}_2(pk_2, m_2; r_2) = c_2] \right\} \leq 2^{-\gamma_2}$.

Regarding the hybrid scheme:
$\forall m \in \mathcal{M}_{sc} = \mathcal{M}_1, \forall (pk_1, sk_1) \leftarrow \text{KeyGen}_1(1^\lambda), \forall (pk_2, sk_2) \leftarrow \text{KeyGen}_2(1^\lambda)$:

$$\alpha^m = \underset{c_{sc} \in \mathcal{C}_{sc}}{\text{Max}} \left\{ \Pr_{r_1, r_2} [\text{Enc}_{sc}((pk_1, pk_2), m; (r_1, r_2)) = c_{sc}] \right\} \tag{26}$$

$$= \underset{u_1 \in \mathcal{U}_1^m, c_2 \in \mathcal{C}_2}{Max} \left\{ \Pr_{r_1} [\text{Enc}_1(pk_1, m_1; r_1) = (u_1, v_1)] \right.$$
$$\left. \times \Pr_{r_1, r_2} [\text{Enc}_2(pk_2, v_1; r_2) = c_2 \mid \text{Enc}_1(pk_1, m_1; r_1) = (u_1, v_1)] \right\} \tag{27}$$

Let v_1^* be the element of $\mathcal{V}_1^m$ that maximizes the probability that its encryption by PKE_2 yields c_2: $\forall v_1 \in \mathcal{V}_1^m, v_1 \neq v_1^*$,

$$\Pr_{r_1, r_2} [\text{Enc}_2(pk_2, v_1; r_2) = c_2 \mid \text{Enc}_1(pk_1, m_1; r_1) = (u_1, v_1)]$$
$$\leq \Pr_{r_2} [\text{Enc}_2(pk_2, v_1^*; r_2) = c_2] \tag{28}$$

As $\forall m \in \mathcal{M}_1$, $\mathcal{V}_1^m \subseteq \mathcal{M}_2$ and PKE_2 is γ_2-spread, we have:

$$\alpha^m \leq \underset{c_2 \in \mathcal{C}_2}{\text{Max}} \left\{ \Pr_{r_2} [\text{Enc}_2(pk_2, v_1^*; r_2) = c_2] \right\} = \alpha_2^{v_1^*} \tag{29}$$

$$\leq 2^{-\gamma_2} \tag{30}$$

This proves that if PKE_2 is γ_2-spread, the hybrid PKE is γ_{sc}-spread, with $\gamma_{sc} \geq \gamma_2$.

Proof of the Worst-Case Correctness. Let $\text{Fail}(\text{PKE}_i, (pk_i, sk_i), m_i; r_i)$ be the event of failure by PKE_i of the decryption of a ciphertext $c_i := \text{Enc}_i(pk_i, m_i; r_i)$, with $m_i \in \mathcal{M}_i$, $r_i \in \mathcal{R}_i$ and $(pk_i, sk_i) \leftarrow \text{KeyGen}_i(1^\lambda)$. We have:

$$\begin{aligned}\text{Fail}_{sc} = &\ \text{Fail}(\text{PKE}_{sc}, (pk_{sc}, sk_{sc}), m \in \mathcal{M}_1, (r_1, r_2) \in \mathcal{R}_1 \times \mathcal{R}_2) &(31)\\ \subseteq &\ \text{Fail}(\text{PKE}_1, (pk_1, sk_1), m; r_1) \cup \text{Fail}(\text{PKE}_2, (pk_2, sk_2), v_1; r_2) &(32)\end{aligned}$$

Consequently, the probability of failure of the hybrid PKE is bounded as: $p_{sc} \leq p_1 + p_2$, with:

$$\begin{aligned}p_{sc} = &\Pr_{r_1,r_2}[\text{Dec}_{sc}(sk_{sc}, \text{Enc}_{sc}(pk_{sc}, m; (r_1, r_2))) \neq m] &(33)\\ p_1 = &\Pr_{r_1}[\text{Dec}_1(sk_1, \text{Enc}_1(pk_1, m; r_1)) \neq m] &(34)\\ p_2 = &\Pr_{r_1,r_2}[\text{Dec}_2(sk_2, \text{Enc}_2(pk_2, v_1; r_2)) \neq v_1] &(35)\\ = &\sum\nolimits_{r_1} \Big(\Pr_{r_2}[\text{Dec}_2(sk_2, \text{Enc}_2(pk_2, v_1; r_2)) \neq v_1 \mid v_1 = \pi_v(\text{Enc}_1(pk_1, m; r_1))]\\ &\times \Pr_{r_1}[v_1 = \pi_v(\text{Enc}_1(pk_1, m; r_1))]\Big) &(36)\end{aligned}$$

Let us now consider the value $\tilde{v_1} \in \mathcal{V}_1^m$ that maximizes the probability of failure of the decryption algorithm Dec_2:

$$\begin{aligned}&\forall v_1 \in \mathcal{V}_1^m, v_1 \neq \tilde{v_1}, \Pr_{r_2}[\text{Dec}_2(sk_2, \text{Enc}_2(pk_2, v_1; r_2)) \neq v_1]\\ &\qquad\leq \Pr_{r_2}[\text{Dec}_2(sk_2, \text{Enc}_2(pk_2, \tilde{v_1}; r_2)) \neq \tilde{v_1}] &(37)\end{aligned}$$

We have:

$$\begin{aligned}p_2^{max} = \underset{m\in\mathcal{M}_1}{\text{Max}}\{p_2\} \leq &\ \underset{m\in\mathcal{M}_1}{Max}\Big\{\Pr_{r_2}[\text{Dec}_2(sk_2, \text{Enc}_2(pk_2, \tilde{v_1}; r_2)) \neq \tilde{v_1}]\\ &\times \sum\nolimits_{r_1} \Pr_{r_1}[v_1 = \pi_v(\text{Enc}_1(pk_1, m; r_1))]\Big\} &(38)\\ \leq &\ \underset{m\in\mathcal{M}_1}{\text{Max}}\left\{\Pr_{r_2}[\text{Dec}_2(sk_2, \text{Enc}_2(pk_2, \tilde{v_1}; r_2)) \neq \tilde{v_1}]\right\} &(39)\end{aligned}$$

Since the set $\mathcal{V}_1^m$ is included in the message space of PKE_2 ($\mathcal{V}_1^m \subseteq \mathcal{M}_2$), we can bound the above inequality as follows:

$$p_2^{max} \leq \underset{m_2\in\mathcal{M}_2}{\text{Max}}\left\{\Pr_{r_2}[\text{Dec}_2(sk_2, \text{Enc}_2(pk_2, m_2; r_2)) \neq m_2]\right\} \quad (40)$$

Finally, we have:

$$\begin{aligned}\delta_{sc}^{wc} = &\underset{(pk_{sc},sk_{sc})}{\mathbb{E}}\left(\underset{m\in\mathcal{M}_1}{\text{Max}}\{p_{sc}\}\right) \leq \underset{(pk_{sc},sk_{sc})}{\mathbb{E}}\left(\underset{m\in\mathcal{M}_1}{\text{Max}}\{p_1\} + \underset{m\in\mathcal{M}_1}{\text{Max}}\{p_2\}\right) &(41)\\ \leq &\underset{(pk_1,sk_1)}{\mathbb{E}}\left(\underset{m\in\mathcal{M}_1}{\text{Max}}\{p_1\}\right) + \underset{(pk_{sc},sk_{sc})}{\mathbb{E}}\left(\underset{m\in\mathcal{M}_1}{\text{Max}}\{p_2\} = p_2^{max}\right) &(42)\\ \leq &\underset{(pk_1,sk_1)}{\mathbb{E}}\left(\underset{m\in\mathcal{M}_1}{\text{Max}}\{p_1\}\right)\\ &+ \underset{(pk_2,sk_2)}{\mathbb{E}}\left(\underset{m_2\in\mathcal{M}_2}{\text{Max}}\left\{\Pr_{r_2}[\text{Dec}_2(sk_2, \text{Enc}_2(pk_2, m_2; r_2)) \neq m_2]\right\}\right) &(43)\\ \delta_{sc}^{wc} \leq &\ \delta_1^{wc} + \delta_2^{wc} &(44)\end{aligned}$$