Snake Mackerel: An Isogeny-Based AKEM Leveraging Randomness Reuse

Jonas Janneck[1], Jonas Meers[1], Massimo Ostuzzi[1], and Doreen Riepel[2]

[1] Ruhr University Bochum, Germany
{jonas.janneck,jonas.meers,massimo.ostuzzi}@rub.de
[2] CISPA Helmholz Center for Information Security, Saarbrücken, Germany
riepel@cispa.de

August 14, 2025

Abstract. An Authenticated Key Encapsulation Mechanism (AKEM) combines public-key encryption and digital signatures to provide confidentiality and authenticity. AKEMs build the core of Hybrid Public Key Encryption (RFC 9180) and serve as a useful abstraction for messaging applications like the Messaging Layer Security (MLS) protocol (RFC 9420) and Signal's X3DH protocol. To date, most existing AKEM constructions either rely on classical (non post-quantum) assumptions or on unoptimized black-box approaches leading to suboptimal efficiency.

In this work, we choose a different abstraction level to combine KEMs and identification schemes more efficiently by leveraging randomness reuse. We construct a generic scheme and identify the necessary security requirements on the underlying KEM and identification scheme when reusing parts of their randomness. This allows for a concrete instantiation from isogenies based on the POKÉ KEM (EUROCRYPT'25) and the SQIsignHD identification scheme (EUROCRYPT'24). To be used in our black-box construction, the identification scheme requires the more advanced security property of response non-malleability. Hence, we further show that a slight modification of SQIsignHD satisfies this notion, which might be of independent interest.

Putting everything together, our final scheme yields the most compact AKEM from PQ assumptions with public keys of 366 bytes and ciphertexts of 216 bytes while fulfilling the strongest confidentiality and authenticity notions.

Table of Contents

1 Introduction

Authenticated public-key encryption (APKE) was formally introduced in [ABH+21] but the concepts date back to the introduction of signcryption by Zheng [Zhe97]. The idea is to combine two of the most fundamental cryptographic primitives, namely (public key) encryption and signatures. Since it is usually easier to analyze the security of a key encapsulation mechanism (KEM) compared to a public key encryption (PKE) scheme, the natural adaptation is to consider an authenticated KEM (AKEM) as introduced in the RFC for Hybrid Public Key Encryption (HPKE) [BBLW22].

As both, KEM/PKE and signatures, often occur in larger protocols together, it can be beneficial to analyze their combination for two reasons. First, there can be more efficient constructions of a combined primitive compared to executing both components on their own. This was the main motivation in early literature, e.g., [Zhe97, ABF12, ZI98], which introduced schemes more efficient in running time and size than the execution of separated primitives. Second, modelling security is more subtle and complex since one of the primitives can influence or compromise security of the other, see for example [AR10].

Security Notions for AKEMs. There are two main security properties for AKEMs: Confidentiality and Authenticity. Both can be considered in two different settings. In the outsider setting, adversaries do not take part in the communication between two honest parties. In the insider setting, adversaries can corrupt one party taking part in the communication, and security guarantees should still hold for the other honest party. In the following, we will focus on the stronger insider setting. More explicitly, confidentiality in the insider setting means that an adversary cannot learn any valuable information from a ciphertext sent to an honest party even if they know the sender's secret key.[1] Insider authenticity means that it should be hard for an adversary to come up with a valid ciphertext which authenticates an honest party as the sender, even if they know (or choose) the receiver's secret key.

An additional property of AKEMs is sender deniability which means that a sender can plausibly deny having sent a (potentially incriminating) message to a third party judge. Note that the sender wants to authenticate themselves to the designated receiver at the same time. In the honest receiver setting, we assume that the receiver does not fake any ciphertexts to frame the sender and the judge is aware of this. Additionally, the judge does not know the receiver's secret keys but has knowledge about the sender's secret key; for example, the sender could be compelled by law enforcement.

Applications. When introduced, AKEMs were used to model and prove two modes of HPKE [BBLW22, ABH+21]. The RFC discusses several modes, e.g., the `AuthPSK` mode, analyzed in [AJKL23], which is specified to be used in the Message Layer Security (MLS) protocol [BBR+23], a standard for secure group messaging. While the current RFCs implement cryptography over prime-order groups, post-quantum (PQ) versions for MLS as well as for HPKE are under consideration and not released yet. Even though the community has standardized PQ KEMs and signature schemes with thorough analyses, using these building blocks in higher level protocols and more complex primitives can lead to the need of a more involved analysis. Additionally, AKEMs occur implicitly in higher level protocols, making them an interesting target to study. One example is the usage of signcryption in Apple's iMessage protocol which was identified and abstracted in a later step [BS20a]. Similarly, there is a connection between AKEMs and (PQ variants of) Signal's X3DH protocol [MP16], as observed and used in [BFG+20, CHN+24, Nio25][2]. The abstraction of a symmetric-key variant of signcryption has further revealed security issues in group messaging applications [JKS24, JK25].

It is crucial to understand these components and implications to the higher level protocols' security guarantees. Especially when moving to PQ security, this can ensure the secure replacement of building blocks. However, this may come with a loss in efficiency; mostly in terms of

[1] The concrete notion is even stronger by allowing the adversary to choose the sender's key arbitrarily.

[2] Brendel et al. [BFG+20] introduced the notion of a split KEM to replace X3DH. AKEMs can be seen as a generalization of split KEMs and in particular have the same syntax as symmetric split KEMs.

communication cost due to increased sizes, sometimes also in terms of running time. Hence, concrete optimized constructions have been considered both in the classical (DH) setting and in the lattice setting. We will discuss and compare these as well as black-box constructions in Section 1.2. To date, there is still no non-black-box construction for AKEMs from isogenies, leveraging the full potential. We aim to fill this gap by applying randomness reuse techniques to isogeny-based KEMs and signatures.

Isogenies. Isogeny-based cryptography is a branch of post-quantum cryptography that is gradually attracting researchers' interest for its very compact sizes. It can be divided in two different (but connected) frameworks: *oriented* and *non-oriented*.

The oriented framework is based on the class group action on special subsets of elliptic curves. The most prominent scheme in the oriented case is CSIDH [CLM+18], a NIKE that allows for an easy abstraction in the language of group actions [ADMP20]. While it is by far the most compact NIKE achieving post-quantum security, its main bottleneck remains computational efficiency compared to other post-quantum KEMs [CCSC+24]. Moreover, there exists a subexponential quantum attack on all oriented isogeny-based schemes [Kup05], and their quantum security is still an open question [Pei20, BS20b, CSCJR22].

The non-oriented framework, on the other hand, does not suffer from the same quantum attack. Furthermore, non-oriented constructions are typically much more efficient and compact. One example is the signature scheme SQISign introduced in 2020 [DKL+20] and currently submitted to the NIST competition [Cal]. Another non-oriented scheme is the SIDH NIKE, which was submitted to the first PQ NIST competition, but broken in 2022 during the fourth and last round. The break-through attacks [CD23, MMP+23, Rob23] drastically changed isogeny-based cryptography, introducing a new tool-box for isogeny-based constructions based on higher-dimensional (HD) isogenies [Rob24]. One of the first and most prominent schemes leveraging HD isogenies is SQISignHD [DLRW24], a successor of SQISign that improves on both efficiency and compactness. Additionally, the POKÉ scheme in [BM25] represents the most recent PKE proposal leveraging HD isogenies.

Contributions. We summarize our contributions as follows:

- We give a generic AKEM construction called SnakeMackerel (short: SnakeM) from a split-ciphertext KEM and an ID scheme, in which randomness is shared between (parts of) the KEM ciphertext and the commitment of the ID scheme.
- We give new definitions for the underlying building blocks to achieve insider confidentiality and authenticity for AKEMs as well as honest receiver deniability. Our definitions are as weak as possible, but we need to take into account challenges in isogeny-based cryptography which are not present in the traditional Diffie-Hellman setting. (See technical overview for more details.)
- We propose SQIsignHD$^+$, a modification of the SQIsignHD identification scheme that features non-malleable response isogenies.
- We construct an efficient AKEM from isogenies by combining the POKÉ KEM and SQIsignHD$^+$ identification scheme. This way, we get compact public keys and ciphertexts.

1.1 Technical Overview

Classical constructions are often used as a blueprint to construct PQ schemes. Our starting point is (a variant of) the insider-secure AKEM from Zheng [Zhe97]. In contrast to [Zhe97], we use an ElGamal KEM instead of a Diffie-Hellman NIKE for the sake of our exposition. The idea is the same but it is more general and fits our later construction better.

Randomness Reuse. The main idea for an efficiency gain is to share the randomness for both schemes, i.e., keep the randomness for the KEM and reuse it for the signature scheme. More specifically, consider the encapsulation algorithm of an ElGamal KEM:

$$(\mathsf{ct} = (g^r, \mathsf{pk}^r \cdot \mathsf{k}), \mathsf{k}) \leftarrow \mathsf{Encaps}(\mathsf{pk}; r),$$

where we make the randomness r explicit. Similarly, we write a Schnorr signature as

$$(\mathsf{com} = g^r, \mathsf{rsp} = r + \mathsf{sk} \cdot \mathsf{chl}) \leftarrow \mathsf{Sign}(\mathsf{sk}, m; r),$$

where $\mathsf{chl} = \mathsf{H}(\mathsf{com}, m)$ for some hash function H. For a separate execution, the AKEM ciphertext would consist of the KEM ciphertext and the signature. If we use the same randomness r for both primitives, we can save one element since the first part of the KEM and the commitment of the signature scheme have the same distribution. To enhance security, the two primitives can be interleaved by signing the KEM key along with some context, i.e., computing $\mathsf{H}(\mathsf{com}, (\mathsf{k}, \mathsf{ctxt}))$, where the context ctxt may include the KEM public key and other (public) information, and the final AKEM key is derived via another random oracle $\mathsf{H}'(\mathsf{com}, \mathsf{k}, \mathsf{ctxt}')$, where ctxt' may now also include rsp.

While in the above construction reusing the randomness preserves security (under slightly stronger assumptions, see also Appendix F), this is not generally true. Not only does it increase the complexity of the analysis, it can also introduce security vulnerabilities. The idea of reusing randomness for efficiency improvements also appears in other contexts, e.g., for multi-recipient PKE schemes [BBS03,BF07]. However, reusing the randomness between two different primitives, i.e., KEM and signatures, is generally expected to be more complex.

To see the concrete requirements for reusing the randomness, we provide a general construction based on a *split-ciphertext KEM* and a canonical identification (ID) scheme. Splitting the encapsulation and the resulting ciphertexts of the KEM is inspired by the above construction and opens the door for a reusability property. A similar approach was used in [ABF12,AGKS05]. Note that it is possible to interpret any KEM as having a split encapsulation and ciphertext; some can be even split naturally, e.g. [BDK+18, BM25]. At this point, we want to highlight one property that will be important for our abstraction, namely that in the above example the second part of the ciphertext is deterministically computed from the randomness. This will not be the case for us and will introduce additional challenges.

In the following, we will only use the notion of split-ciphertext KEMs and simply denote it as a KEM. For the construction to work, the KEM and the ID scheme need to be *compatible* in the sense that the first encapsulation of the KEM must be the same algorithm as the computation of the commitment of the ID scheme. Assuming compatibility, we then show that our construction SnakeM fulfills the following security notions in the random oracle model:

- Insider CCA: This is the strongest confidentiality notion for AKEMs and requires the underlying KEM to be one-way secure in the presence of a plaintext checking oracle.
- Insider Authenticity: This is the strongest authenticity notion for AKEMs and requires the underlying ID scheme to be secure against impersonation attacks, special soundness, and non-malleability. While the two former notions are well established for ID schemes, we introduce the latter one to catch an edge case that comes up for ID schemes that do not have a unique response. For an ID scheme to be non-malleable it should be hard given a valid transcript $(\mathsf{com}, \mathsf{chl}, \mathsf{rsp})$ to find a new response rsp' such that $(\mathsf{com}, \mathsf{chl}, \mathsf{rsp}')$ verifies as well. The notion can be seen as an ID equivalence of the gap between (plain) existential unforgeability and strong existential unforgeability of a signature scheme. Further, we need these notions in the presence of an encapsulation oracle.[3]
- Honest Receiver Deniability: This is the strongest notion of deniability when fulfilling insider authenticity at the same time, and our construction only requires one-wayness of the KEM. One of the main challenges is to handle the signature which is publicly verifiable and thus can break deniability trivially due to its non-repudiation. One technique is to encrypt the signature so that it can only be verified by the holder of the KEM secret key. We show that in our construction a simpler approach suffices. Namely, we only need to encrypt the response (of the underlying ID scheme) and, interestingly, we only require a simple XOR-padding instead of a full encryption scheme.

The ElGamal KEM satisfies all these properties and thus yields a very efficient and strongly secure AKEM scheme in the prime-order group setting. The two main reasons are that the two primitives are using the same or very similar mathematical structure, i.e. the underlying group, and that they are simple in the sense that the ciphertext/signature components are just

[3] Similarities and differences to a transcript oracle are discussed in Section 3.1 and Appendix F.

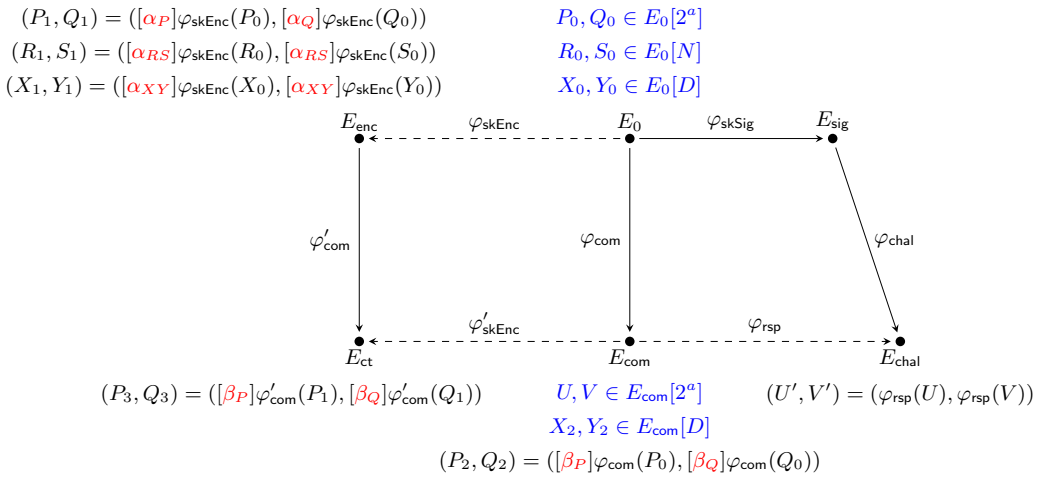

Fig. 1. SnakeM instantiated with POKÉ (left) and SQIsignHD (right). Blue values are computed deterministically, red values are secrets. Dashed lines represent HD isogenies. The shared key is $(X_3, Y_3) = A \cdot (\varphi'_{\mathsf{com}}(X_1), \varphi'_{\mathsf{com}}(Y_1))^\top = ([\alpha_{XY}]\varphi'_{\mathsf{skEnc}}(X_2), [\alpha_{XY}]\varphi'_{\mathsf{skEnc}}(Y_3))$ for some $A \in \mathrm{GL}(2, D)$ (cf. Section 4).

single group elements. However, for PQ secure schemes, the mathematical structures are more complicated and the schemes are getting more involved.

In [ABF12], Arriaga, Barbosa and Farshim provide a similar composition with randomness reuse for compatible PKE and signature schemes, instantiated over pairing groups. We want to note that (1) their syntax is insufficient to capture our isogeny-based instantiation due to extra randomness being required in the second ciphertext part of the KEM, (2) their security notions for the two building blocks are quite strong (e.g., they imply CCA security and strong unforgeability) and (3) they require a property called conditional injectivity (similar to non-malleability) for both KEM and signature that we do not need for the KEM and that we can relax for the signature (or, more specifically, ID) scheme.

Randomness Reuse with Isogenies. We instantiate our generic construction SnakeM with POKÉ and (a modified version of) SQIsignHD, yielding the construction in Figure 1, which we call SnakeM-Iso. Notably, the commitment isogeny φ_{com} and commitment curve E_{com} is shared amongst both schemes. In this case, leveraging randomness reuse presents itself with several challenges. On the one hand, we need to ensure compatibility between POKÉ and SQIsignHD. This mainly involves finding a suitable prime characteristic p that provides enough *accessible* torsion for POKÉ and SQIsignHD. On the other hand, SQIsignHD does not satisfy the non-malleability property required for SnakeM's strong authenticity guarantees, making a modification necessary. We discuss these challenges and their solutions in the next paragraphs.

Accessing Twisted Torsion. In order to unify the prime requirements of POKÉ and SQIsignHD while also achieving more compact sizes, we make use of the technique introduced in B-SIDH [Cos20]. The idea is that for each $\mathbb{F}_{p^2}$-rational curve E with accessible $(p+1)$-torsion, there exists another curve E^t, called the *quadratic twist*, with accessible $(p-1)$-torsion, such that E and E^t are isomorphic only over the quadratic extension $\mathbb{F}_{p^4}$. Thus, one could extend the base field and access torsion $p^2 - 1 = (p+1)(p-1)$ for efficient isogeny computations. Fortunately, most of these computations are *twist-agnostic*, meaning that they can be performed via x-only arithmetic over $\mathbb{F}_{p^2}$. Therefore, one can use the accessible torsion of both E and E^t without extending the field.

Non-Malleable Response Isogenies. In SQIsignHD the response is an isogeny $\varphi_{\mathsf{rsp}} : E_{\mathsf{com}} \to E_{\mathsf{chal}}$ between the commitment and challenge curve. Since the degree of φ_{rsp} is a (potentially non-smooth) integer q, SQIsignHD represents φ_{rsp} via a 4-dimensional isogeny Φ, eliminating any smoothness condition on q. The HD isogeny Φ is then represented as the tuple (q, U', V'), where (U', V') are the images of deterministically chosen points (U, V) under φ_{rsp}.

Table 1. Comparison of different AKEMs along with their security notions and whether they are post-quantum secure (PQ). Deniability properties marked with a "$*$" have not been formally proven in the respective work. † CSIDH parameters are chosen according to the most aggressive parameter set found in [CCSC+24] and discussed in more detail in Section 1.2.

Scheme (variant)	Confidentiality	Authenticity	Deniability	PQ	Size (in bytes)	
					ct	pk
Group-based						
DH-AKEM [ABH+21]	**Ins-CCA**	**Out-Aut**	**DR***	✗	32	32
Zheng [Zhe97, BSZ02]	**Ins-CCA**	**Ins-Aut**	**HR***	✗	64	64
Lattice-based						
EtStH-AKEM (BAT + Antrag) [AJKL23]	**Ins-CCA**	**Out-Aut**	—	✓	1 119	1 417
NIKE-AKEM (Swoosh) [AJKL23]	**Ins-CCA**	**Out-Aut**	**DR***	✓	$> 221\,184$	$> 221\,184$
EaNtH-AKEM (BAT + Swoosh)[4]	**Ins-CCA**	**Out-Aut**	**DR***	✓	473	$> 221\,705$
FrodoKEX+ [CHN+24]	**IND-1BatchCCA**	**UNF-1KCA**	**DR**	✓	72	21 300
Sparrow-KEM [Nio25]	**IND-1BatchCCA**	**UNF-1KCA**	**DR**	✓	40	2 592
Den. AKEM (BAT + Gandalf) [GJK24]	**Ins-CCA**	**Out-Aut**	**HR & DR**	✓	1 749	1 417
Isogeny-based						
EtStH-AKEM (POKE-4D + SQIsignHD) [AJKL23]	**Ins-CCA**	**Out-Aut**	—	✓	336	291
NIKE-AKEM (CSIDH) [AJKL23]	**Ins-CCA**	**Out-Aut**	**DR***	✓	256†	256†
EaNtH-AKEM (POKE-4D + CSIDH)[4]	**Ins-CCA**	**Out-Aut**	**DR***	✓	230	483
Den. AKEM (POKE-4D + Erebor) [GJK24]	**Ins-CCA**	**Out-Aut**	**HR & DR**	✓	586	291
SnakeM-Iso (Section 6)	**Ins-CCA**	**Ins-Aut**	**HR**	✓	216	366

However, it is possible to find $d \in \mathbb{N}$ such that the tuple $(d^2q, [d]U', [d]V')$ is still a valid representation of an isogeny $\varphi'_{\mathsf{rsp}} : E_{\mathsf{com}} \to E_{\mathsf{chal}}$ having the same domain and codomain. Crucially, the represented φ'_{rsp} is not *cyclic* anymore since it factors as $\varphi'_{\mathsf{rsp}} = \varphi_{\mathsf{rsp}} \circ [d]$. Therefore, it would be desirable to check the cyclicity of φ_{rsp} given its HD representation Φ. Although a generic cyclicity check is currently out of reach, we show that it is sufficient to check whether φ_{rsp} factors through a *small* scalar multiplication with $d^2 \leq \log p$, which requires minimal changes to SQIsignHD.

1.2 Performance and Comparison

In Table 1, we compare AKEMs from the literature to our construction SnakeM-Iso. We include constructions based on elliptic curves as a baseline. We then list lattice-based and isogeny-based AKEMs, their security guarantees and ciphertext and public key sizes. We instantiate black-box constructions with the most compact schemes we are aware of, as denoted in the table.

Black-Box Constructions. There are several black-box constructions, e.g. [AJKL23, GJK24, GHJ25, ADR02, AR10], that can be instantiated from PQ secure primitives. Namely, EtStH-AKEM, NIKE-AKEM, and EaNtH-AKEM from Table 1. A more detailed discussion of the schemes as well as a brief overview of early work in signcryption can be found in Appendix A.

Lattice-based AKEMs. Concrete, i.e. non-black-box, constructions for AKEMs (or Split KEMs[2]) were proposed from lattices [BFG+20, CHN+24, Nio25] as a building block for key exchange, for example FrodoKEX+ [CHN+24] and Sparrow-KEM [Nio25]. This is particularly useful in the PQ setting due to the lack of secure and efficient non-interactive key exchange (NIKE) schemes. Among the black-box constructions, sizes vary a lot and constitute the main bottleneck. Hence, to the best of our knowledge, we choose the smallest instantiations. For the KEM, we choose BAT [FKPY22], for the signature Falcon [PFH+20] using Antrag [ENS+23] trapdoor generation, for the NIKE we choose Swoosh [GdKQ+24], and for the ring signature Gandalf [GJK24] based on Antrag.

Isogeny-based AKEMs. The only isogeny-based AKEMs are based on existing black-box constructions. Like in the lattice setting, we choose the most efficient instantiations found in the literature for our comparison. Regarding the KEM, POKÉ outperforms other competitors like FESTA [BMP23] and M(D)-SIDH [FMP23]. Compared to QFESTA [NO24], POKÉ offers smaller ciphertexts but slightly larger public keys. Additionally, POKÉ is an order of magnitude

[4] The construction EaNtH, Encapsulate-and-NIKE-then-Hash, is briefly discussed in Appendix A.

faster than QFESTA. For the NIKE, CSIDH [CLM+18] is the only viable option as other class group actions, such as SCALLOP [DFK+23], take minutes to compute. Nevertheless, due to the ongoing debate about CSIDH's quantum security and the subsequent increase of parameters, even an optimal implementation of CSIDH takes several seconds to compute as well. Regarding signatures, SQIsignHD is the most compact isogeny-based signature scheme. In terms of efficiency it is, however, outperformed by the more recent SQIsign2D-West [BDD+24]. Lastly, for the ring signature we choose Erebor [BLL24], which represents the most compact PQ ring signature for ring sizes up to 31 users.

Our AKEM SnakeM-Iso. Our new construction SnakeM-Iso represents one of the most efficient PQ AKEMs to date. In terms of size of combined public key and ciphertext, it outperforms all lattice based constructions by at least a factor of $\times 4$, and it is also more compact than all other isogeny-based constructions. Compared to group-based constructions, SnakeM-Iso offers PQ security but cannot match the high performance and compactness of any group-based scheme – a trade-off that is common across all PQ constructions.

2 Preliminaries

2.1 Notation

Sets and Algorithms. Unless stated otherwise, we assume that elliptic curve points are specified by their x-coordinate only. We write $s \xleftarrow{\$} \mathcal{S}$ to denote the uniform sampling of s from the finite set $\mathcal{S}$. For an integer n, we define $[n] := \{1, \ldots, n\}$. Unless otherwise stated, algorithms are probabilistic, and we write $(y_1, \ldots) \xleftarrow{\$} \mathcal{A}(x_1, \ldots)$ to denote that $\mathcal{A}$ returns $(y_1, \ldots)$ when run on input $(x_1, \ldots)$. We write $\mathcal{A}^{\mathcal{B}}$ to denote that $\mathcal{A}$ has oracle access to $\mathcal{B}$ during its execution and by **return** we denote the termination of oracle $\mathcal{B}$ or any other subroutine. Moreover, **output** denotes the termination of the main routine. For an algorithm $\mathcal{A}$, we write $x \in \mathcal{A}$ to denote that x is in the support of $\mathcal{A}$. By "log" we denote the logarithm of base 2, by "ln" of base e. For a Boolean statement B, the notation $[\![\mathrm{B}]\!]$ refers to a bit that is 1 if the statement is true and 0 otherwise.

Games and Notions. Throughout the paper we use code-based games [BR06], where $\Pr[\mathsf{G} \Rightarrow 1]$ denotes the probability that the final output of game G is 1. We implicitly assume that all primitives are defined over public parameters which are given to all its subroutines and with respect to a random oracle space [BR93]. For every primitive we introduce, we denote by $\mathsf{derive}(\mathsf{sk})$ the public key derivation algorithm, which takes as input a secret key sk and outputs a (deterministic) public key pk. We may abuse notation and also allow tuples of secret keys as input, that is $(\mathsf{pk}_0, \mathsf{pk}_1) = \mathsf{derive}(\mathsf{sk}_0, \mathsf{sk}_1)$.

2.2 Prime Numbers for Isogeny Computations

In this section we make some definitions regarding prime numbers. Further isogeny-related preliminaries can be found in Appendix B.

We start by defining prime numbers that are particularly suitable for efficiently computing rational isogenies of large degree.

Definition 2.1 (B-SIDH prime). *Let $p = 2^a c - 1$ be a prime number. We call p a* B-SIDH prime *if $p^2 - 1 = 2^a N D$ for some smooth $N \in \mathbb{N}$.*

The literature on how to compute B-SIDH primes is quite extensive [Cos20, Ste24, CMN21, SEMR24a]. There exist many examples of such primes with various trade-offs between the smoothness of N, the proportional size of 2^a and the overall size of p.

Additionally, some isogeny computations internally rely on Cornacchia's algorithm [Cor08] to solve Diophantine equations. The following definition characterizes primes that allow for an easy solution of those equations.

Definition 2.2. *For some $n \in \mathbb{N}$ an integer q is called* 2^n-good *if $2^n - q$ is a prime congruent 1 modulo 4.*

2.3 Authenticated KEMs

In this section, we introduce syntax and security of authenticated KEMs (AKEMs).

Definition 2.3 (Authenticated Key Encapsulation Mechanism). *An* authenticated key encapsulation mechanism AKEM *is defined as a tuple* $\mathsf{AKEM} := (\mathsf{Gen}, \mathsf{AEncaps}, \mathsf{ADecaps})$ *of the following* PPT *algorithms.*

$(\mathsf{sk}, \mathsf{pk}) \xleftarrow{\$} \mathsf{Gen}$**:** *The probabilistic generation algorithm* Gen *returns a secret key* sk *and a corresponding public key* pk*. We implicitly assume the existence of a shared key space* $\mathcal{K}$*.*

$(\mathsf{ct}, \mathsf{k}) \xleftarrow{\$} \mathsf{AEncaps}(\mathsf{sk}_{\mathsf{SND}}, \mathsf{pk}_{\mathsf{RCV}})$**:** *Given a (sender's) secret key* $\mathsf{sk}_{\mathsf{SND}}$ *and a (receiver's) public key* $\mathsf{pk}_{\mathsf{RCV}}$[5]*, the probabilistic encapsulation algorithm* $\mathsf{AEncaps}$ *returns a ciphertext* ct *and a shared key* $\mathsf{k} \in \mathcal{K}$*.*

$\mathsf{k} \leftarrow \mathsf{ADecaps}(\mathsf{pk}_{\mathsf{SND}}, \mathsf{sk}_{\mathsf{RCV}}, \mathsf{ct})$**:** *Given a (sender's) public key* $\mathsf{pk}_{\mathsf{SND}}$*, a (receiver's) secret key* $\mathsf{sk}_{\mathsf{RCV}}$*, and a ciphertext* ct*, the deterministic decapsulation algorithm* $\mathsf{ADecaps}$ *returns a shared key* $\mathsf{k} \in \mathcal{K}$ *or a failure symbol* $\perp$*.*

The correctness error δ_{AKEM} *is defined as the smallest value such that for all* $(\mathsf{sk}_{\mathsf{SND}}, \mathsf{pk}_{\mathsf{SND}}), (\mathsf{sk}_{\mathsf{RCV}}, \mathsf{pk}_{\mathsf{RCV}}) \in \mathsf{Gen}$ *it holds that*

$$\Pr\left[\mathsf{ADecaps}(\mathsf{pk}_{\mathsf{SND}}, \mathsf{sk}_{\mathsf{RCV}}, \mathsf{ct}) \neq \mathsf{k} \,\middle|\, (\mathsf{ct}, \mathsf{k}) \xleftarrow{\$} \mathsf{AEncaps}(\mathsf{sk}_{\mathsf{SND}}, \mathsf{pk}_{\mathsf{RCV}})\right] \leq \delta_{\mathsf{AKEM}}.$$

We now turn to defining security. As previous works, we consider confidentiality, authenticity and deniability, which we will discuss in more detail below. In this work, we will focus on the single-user and single-challenge setting. One can also define corresponding multi-user multi-challenge notions analogously to the ones in [ABH+21, GJK24]. The single notions generically imply the multi-user multi-challenge notions using a hybrid argument incurring a loss in the number of users and the number of challenges.

Confidentiality. We consider the strongest possible notion of CCA security for an AKEM, in particular that of insider security [ABH+21]. In comparison to [ABH+21], we slightly strengthen the notion by allowing the adversary to choose the challenge receiver to be also the sender when issuing a challenge query; this corresponds to a ciphertext that a party encrypted to itself. Even though this is only a small technical change, it enables a direct and tight proof showing that security in the insider setting implies security in the outsider setting which matches intuition.

Definition 2.4 (Insider CCA). *For an* AKEM*,* Insider-CCA *is defined via the game in Figure 2. We define the advantage of an adversary* $\mathcal{A}$ *as*

$$\mathsf{Adv}^{(q_{\mathtt{Enc}}, q_{\mathtt{Dec}})\text{-}\mathsf{Ins\text{-}CCA}}_{\mathsf{AKEM}}(\mathcal{A}) := \left|\Pr\left[(q_{\mathtt{Enc}}, q_{\mathtt{Dec}})\text{-}\mathbf{Ins\text{-}CCA}_{\mathsf{AKEM}}(\mathcal{A}) \Rightarrow 1\right] - \frac{1}{2}\right|.$$

Authenticity. We also consider the strongest notions of authenticity, namely insider authenticity similar to [GJK24] with the difference that we again allow for "encrypt-to-self" challenges. An insider adversary can choose a receiver's secret key by themselves which models a scenario in which it should be hard to distinguish between a real and a random key even for the designated receiver party. Due to our strengthened notion including the "encrypt-to-self" case one can directly reduce outsider security to insider security which matches intuition.

Definition 2.5 (Insider Authenticity). *For an* AKEM*,* Insider-Authenticity *is defined via the game in Figure 3. We define the advantage of an adversary* $\mathcal{A}$ *as*

$$\mathsf{Adv}^{(q_{\mathtt{Enc}}, q_{\mathtt{Dec}})\text{-}\mathsf{Ins\text{-}Aut}}_{\mathsf{AKEM}}(\mathcal{A}) := \left|\Pr\left[(q_{\mathtt{Enc}}, q_{\mathtt{Dec}})\text{-}\mathbf{Ins\text{-}Aut}_{\mathsf{AKEM}}(\mathcal{A}) \Rightarrow 1\right] - \frac{1}{2}\right|.$$

[5] While we make sender and receiver explicit in the notation here, a secret/public key can be used for sending and receiving.

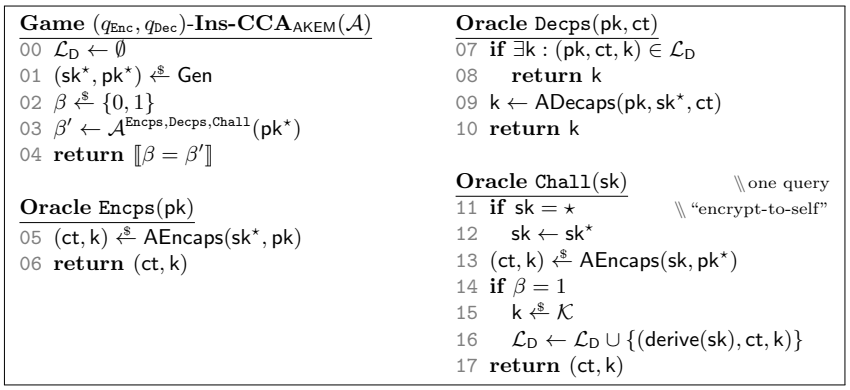

Fig. 2. Game defining Ins-CCA security for an authenticated key encapsulation mechanism $\mathsf{AKEM} \coloneqq (\mathsf{Gen}, \mathsf{AEncaps}, \mathsf{ADecaps})$ with adversary $\mathcal{A}$ making at most $q_{\mathtt{Enc}}$ queries to `Encps` and at most $q_{\mathtt{Dec}}$ queries to `Decps`.

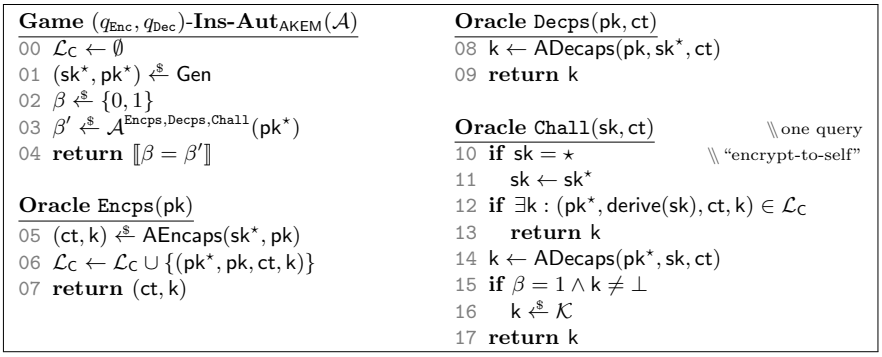

Fig. 3. Game defining Ins-Aut security for an authenticated key encapsulation mechanism $\mathsf{AKEM} \coloneqq (\mathsf{Gen}, \mathsf{AEncaps}, \mathsf{ADecaps})$ with adversary $\mathcal{A}$ making at most $q_{\mathtt{Enc}}$ queries to `Encps` and at most $q_{\mathtt{Dec}}$ queries to `Decps`.

Deniability. We base our definition for deniability on [GJK24] but use a two-user setting to allow for a simplified analysis. From there, one can get multi-user multi-challenge security by a hybrid argument. Gajland et al. [GJK24] discuss two different settings for deniability; in one the receiver is dishonest, in the other one the receiver is honest. Further they note that deniability in the dishonest receiver setting and insider authenticity are mutually exclusive. Since we aim for a scheme fulfilling insider authenticity, we focus on the honest receiver setting. In the honest receiver setting, a sender can plausibly deny having sent a message to an honest sender in the presence of a third party judge. This holds even if the sender's keys leak but not the (honest) receiver ones.[6]

Definition 2.6 (Deniability). *For an* AKEM *and a simulator* Sim, honest receiver deniability *is defined via the game in Figure 4. We define the advantage of an adversary* $\mathcal{A}$ *as*

$$\mathsf{Adv}^{\mathsf{HR\text{-}Den}}_{\mathsf{AKEM},\mathsf{Sim}}(\mathcal{A}) \coloneqq \left| \Pr[\mathbf{HR\text{-}Den}_{\mathsf{AKEM},\mathsf{Sim}}(\mathcal{A}) \Rightarrow 1] - \frac{1}{2} \right|.$$

2.4 Split-Ciphertext KEM

To leverage randomness reuse in our higher level scheme, we need to use a modification of classical KEMs. We split the encapsulation procedure and the resulting ciphertext into two

[6] The case were the receiver's keys leak as well is not a meaningful security notion (see [DHM+20, GJK24] for more details).

$\underline{\textbf{Game HR-Den}_{\mathsf{AKEM},\mathsf{Sim}}(\mathcal{A})}$
00 $(\mathsf{sk}_1, \mathsf{pk}_1) \xleftarrow{\$} \mathsf{Gen}$
01 $(\mathsf{sk}_2, \mathsf{pk}_2) \xleftarrow{\$} \mathsf{Gen}$
02 $(\mathsf{ct}_0, \mathsf{k}_0) \xleftarrow{\$} \mathsf{AEncaps}(\mathsf{sk}_1, \mathsf{pk}_2)$
03 $(\mathsf{ct}_1, \mathsf{k}_1) \xleftarrow{\$} \mathsf{Sim}(\mathsf{pk}_1, \mathsf{pk}_2)$
04 $\beta \xleftarrow{\$} \{0,1\}$
05 $\beta' \leftarrow \mathcal{A}(\mathsf{sk}_1, \mathsf{pk}_1, \mathsf{pk}_2, \mathsf{ct}_\beta, \mathsf{k}_\beta)$
06 **return** $[\![\beta = \beta']\!]$

Fig. 4. Game defining honest-receiver deniability for an AKEM $\mathsf{AKEM} := (\mathsf{Gen}, \mathsf{AEncaps}, \mathsf{ADecaps})$ and a simulator Sim for adversary $\mathcal{A}$.

parts and call this primitive a split-ciphertext KEM. Note that every KEM can be considered as a split-ciphertext KEM. However, as discussed in the technical overview, for some constructions there is a natural way how to split the ciphertext which makes these constructions particularly useful for our main scheme. A similar design was introduced in [ABF12, AGKS05].

Definition 2.7 (Split-Ciphertext KEM). *A* split-ciphertext key encapsulation mechanism KEM *is defined as a tuple* $\mathsf{KEM} := (\mathsf{Gen}, \mathsf{Encaps}_0, \mathsf{Encaps}_1, \mathsf{Decaps})$ *of the following* PPT *algorithms.*

$(\mathsf{sk}, \mathsf{pk}) \xleftarrow{\$} \mathsf{Gen}$**:** *The probabilistic generation algorithm* Gen *returns a secret key* sk *and a corresponding public key* pk*. We implicitly assume the existence of a shared key space* $\mathcal{K}$ *and partial randomness space* $\mathcal{R}$*.*

$(\mathsf{ct}_0, R) \xleftarrow{\$} \mathsf{Encaps}_0$ *: The probabilistic first encapsulation algorithm* Encaps_0 *returns a first ciphertext part* ct_0 *and a partial randomness* $R \in \mathcal{R}$*.*

$(\mathsf{ct}_1, K) \xleftarrow{\$} \mathsf{Encaps}_1(\mathsf{pk}, R)$**:** *Given a public key* pk *and partial randomness* $R \in \mathcal{R}$*, the probabilistic*[7] *second encapsulation algorithm* Encaps_1 *returns a ciphertext* ct_1 *and a shared key* $K \in \mathcal{K}$*.*

$K \leftarrow \mathsf{Decaps}(\mathsf{sk}, \mathsf{ct}_0, \mathsf{ct}_1)$**:** *Given a secret key* sk *and a first and second ciphertext* ct_0 *and* ct_1*, the deterministic decapsulation algorithm* Decaps *returns a shared key* $K \in \mathcal{K}$ *or a failure symbol* $\perp$*.*

The correctness error δ_{KEM} *is defined as the smallest value such that for every* $(\mathsf{sk}, \mathsf{pk}) \in \mathsf{Gen}$ *it holds that*

$$\Pr\left[\mathsf{Decaps}(\mathsf{sk}, \mathsf{ct}_0, \mathsf{ct}_1) \neq K \;\middle|\; \begin{array}{l}(\mathsf{ct}_0, R) \xleftarrow{\$} \mathsf{Encaps}_0 \\ (\mathsf{ct}_1, K) \xleftarrow{\$} \mathsf{Encaps}_1(\mathsf{pk}, R)\end{array}\right] \leq \delta_{\mathsf{KEM}}.$$

Further we define the *spreadness* of the second ciphertext part, denoted by γ_{KEM}, as the smallest value such that for every $(\mathsf{sk}, \mathsf{pk}) \in \mathsf{Gen}$, every $R \in \mathcal{R}$, and every $\mathsf{ct}^\star$, it holds that

$$\Pr\left[\mathsf{ct}^\star = \mathsf{ct}_1 \middle| (\mathsf{ct}_1, \cdot) \xleftarrow{\$} \mathsf{Encaps}_1(\mathsf{pk}, R)\right] \leq \gamma_{\mathsf{KEM}}.$$

Note that if the second encapsulation Encaps_1 is deterministic there is not spreadness and it holds $\gamma_{\mathsf{KEM}} = 1$.

We also recall two standard definitions, adapted to split-ciphertext KEMs.

Definition 2.8 (OW). *For a split-ciphertext KEM* KEM*,* one-wayness *is defined via the game in Figure 5. We define the advantage of an adversary* $\mathcal{A}$ *as*

$$\mathsf{Adv}^{\mathsf{OW}}_{\mathsf{KEM}}(\mathcal{A}) := \Pr[\mathbf{OW}_{\mathsf{KEM}}(\mathcal{A}) \Rightarrow 1].$$

Definition 2.9 (OW-KCA). *For a split-ciphertext KEM* KEM*,* one-wayness under key checking attacks *is defined via the game in Figure 5. We define the advantage of an adversary* $\mathcal{A}$ *as*

$$\mathsf{Adv}^{q_{\mathtt{Check}}\text{-}\mathsf{OW}\text{-}\mathsf{KCA}}_{\mathsf{KEM}}(\mathcal{A}) := \Pr[q_{\mathtt{Check}}\text{-}\mathbf{OW}\text{-}\mathbf{KCA}_{\mathsf{KEM}}(\mathcal{A}) \Rightarrow 1].$$

[7] The algorithm is probabilistic even given R, i.e., there can be more randomness involved.

Games $\mathbf{OW}_{\mathsf{KEM}}(\mathcal{A})/q_{\texttt{Check}}$-$\mathbf{OW\text{-}KCA}_{\mathsf{KEM}}(\mathcal{A})$	$\underline{\texttt{Check}(K, \mathsf{ct}_0, \mathsf{ct}_1)}$ $\quad \backslash\backslash$ **OW-KCA**
00 $(\mathsf{sk}^\star_{\mathsf{KEM}}, \mathsf{pk}^\star_{\mathsf{KEM}}) \xleftarrow{\$} \mathsf{Gen}$	05 $K' \leftarrow \mathsf{Decaps}(\mathsf{sk}^\star_{\mathsf{KEM}}, (\mathsf{ct}_0, \mathsf{ct}_1))$
01 $(\mathsf{ct}^\star_0, R) \xleftarrow{\$} \mathsf{Encaps}_0$	06 **return** $[\![K = K']\!]$
02 $(\mathsf{ct}^\star_1, K^\star) \xleftarrow{\$} \mathsf{Encaps}_1(\mathsf{pk}^\star_{\mathsf{KEM}}, R)$	
03 $K \xleftarrow{\$} \mathcal{A}^{\texttt{Check}}(\mathsf{pk}^\star_{\mathsf{KEM}}, \mathsf{ct}^\star_0, \mathsf{ct}^\star_1)$	
04 **return** $[\![K = K^\star]\!]$	

Fig. 5. Games defining OW and OW-KCA security for a split-ciphertext KEM $\mathsf{KEM} = (\mathsf{Gen}, \mathsf{Encaps}_0, \mathsf{Encaps}_1, \mathsf{Decaps})$ and an adversary $\mathcal{A}$ making at most $q_{\texttt{Check}}$ queries to `Check`.

Game $q_{\texttt{Trans}}$-$\mathbf{Rsp\text{-}NM}_{\mathsf{ID}}(\mathcal{A})$	**Oracle** `Trans`
00 $(\mathsf{sk}, \mathsf{pk}) \xleftarrow{\$} \mathsf{Gen}$	07 $(\mathsf{com}, \mathsf{st}) \xleftarrow{\$} \mathsf{Com}$
01 $\mathcal{L} \leftarrow \emptyset$	08 $\mathsf{chl} \xleftarrow{\$} \mathsf{ChlSet}$
02 $(\mathsf{com}, \mathsf{chl}, \mathsf{rsp}, \mathsf{rsp}^*) \leftarrow \mathcal{A}^{\texttt{Trans}}(\mathsf{pk})$	09 $\mathsf{rsp} \leftarrow \mathsf{Rsp}(\mathsf{sk}, \mathsf{com}, \mathsf{chl}, \mathsf{st})$
03 **if** $(\mathsf{com}, \mathsf{chl}, \mathsf{rsp}) \in \mathcal{L}$ **and** $(\mathsf{com}, \mathsf{chl}, \mathsf{rsp}^*) \notin \mathcal{L}$	10 $\mathcal{L} \leftarrow \mathcal{L} \cup \{(\mathsf{com}, \mathsf{chl}, \mathsf{rsp})\}$
04 **if** $\mathsf{Ver}(\mathsf{pk}, \mathsf{com}, \mathsf{chl}, \mathsf{rsp}^*) = 1$	11 **return** $(\mathsf{com}, \mathsf{chl}, \mathsf{rsp})$
05 **return** 1	
06 **return** 0	

Fig. 6. Game defining Rsp-NM for an ID scheme $\mathsf{ID} = (\mathsf{Gen}, \mathsf{Com}, \mathsf{Rsp}, \mathsf{Ver})$ with challenge set ChlSet and an adversary $\mathcal{A}$ making at most $q_{\texttt{Trans}}$ queries to `Trans`.

2.5 Identification Scheme

We define standard identification (ID) schemes with a slight modification: The commitment algorithm Com does not get the secret key as input. This is not a restriction for many ID schemes since they do not use the secret key for the commitment anyway. Further standard security notions for ID schemes like special soundness (SS) and impersonation against key-only attacks (IMP-KOA) can be found in Appendix C.

Definition 2.10 (Identification Scheme). *An* identification scheme ID *is defined as a tuple* $\mathsf{ID} := (\mathsf{Gen}, \mathsf{Com}, \mathsf{Rsp}, \mathsf{Ver})$ *of the following* PPT *algorithms:*

$(\mathsf{sk}, \mathsf{pk}) \xleftarrow{\$} \mathsf{Gen}$**:** *The probabilistic generation algorithm* Gen *returns a secret key* sk *and a corresponding public key* pk*. We implicitly assume the existence of a challenge set* ChlSet*.*

$(\mathsf{com}, \mathsf{st}) \xleftarrow{\$} \mathsf{Com}$**:** *The probabilistic commitment algorithm* Com *returns a commitment* com *and a state* st*.*

$\mathsf{rsp} \xleftarrow{\$} \mathsf{Rsp}(\mathsf{sk}, \mathsf{com}, \mathsf{chl}, \mathsf{st})$**:** *Given a secret key* sk*, a commitment* com*, a challenge* $\mathsf{chl} \in \mathsf{ChlSet}$ *and a state* st*, the response algorithm* Rsp *returns a response* rsp*.*

$0/1 \leftarrow \mathsf{Ver}(\mathsf{pk}, \mathsf{com}, \mathsf{chl}, \mathsf{rsp})$**:** *Given a public key* pk*, a commitment* com*, a challenge* chl*, and a response* rsp*, the deterministic verification algorithm* Ver *returns* 0 *(reject) or* 1 *(accept).*

The correctness error δ_{ID} *is defined as the smallest value such that for all* $(\mathsf{sk}, \mathsf{pk}) \in \mathsf{Gen}$ *it holds that*

$$\Pr\left[\mathsf{Ver}(\mathsf{pk}, \mathsf{com}, \mathsf{chl}, \mathsf{rsp}) \neq 1 \,\middle|\, \begin{array}{r}(\mathsf{com}, \mathsf{st}) \xleftarrow{\$} \mathsf{Com} \\ \mathsf{rsp} \xleftarrow{\$} \mathsf{Rsp}(\mathsf{sk}, \mathsf{com}, \mathsf{chl}, \mathsf{st})\end{array}\right] \leq \delta_{\mathsf{ID}}.$$

Further, we define *commitment spreadness* as

$$\gamma_{\mathsf{ID}} := \max_{\mathsf{com}^\star} \Pr\left[\mathsf{com}^\star = \mathsf{com} \mid (\mathsf{com}, \cdot) \xleftarrow{\$} \mathsf{Com}\right].$$

We also define a non-malleability property for transcripts.

Definition 2.11 (Response Non-Malleability). *Let* ID *be an identification scheme. Consider the game* $q_{\texttt{Trans}}$-$\mathbf{Rsp\text{-}NM}_{\mathsf{ID}}$ *in Figure 6. We define the advantage of an adversary* $\mathcal{A}$ *as*

$$\mathsf{Adv}^{q_{\texttt{Trans}}\text{-}\mathsf{Rsp\text{-}NM}}_{\mathsf{ID}}(\mathcal{A}) := \Pr[q_{\texttt{Trans}}\text{-}\mathbf{Rsp\text{-}NM}_{\mathsf{ID}}(\mathcal{A}) \Rightarrow 1].$$

Remark 2.12. Response non-malleability is weaker than the notion of Computational Unique Response (CUR) in [DFPS23], meaning that we put additional restrictions on the output of the adversary. In particular, in the response non-malleability game the transcript for which the adversary computes a forgery must be honestly generated by the `Trans` oracle, which is not required for CUR.

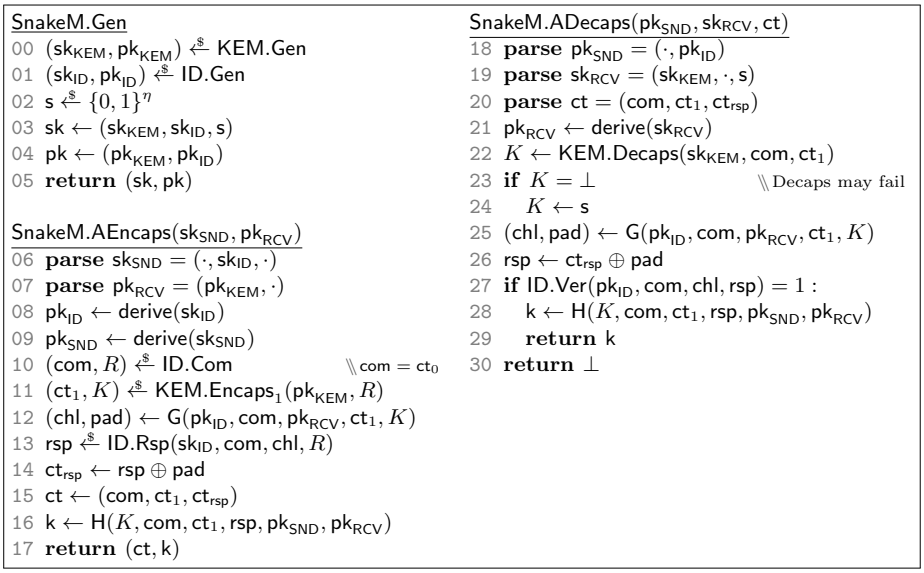

Fig. 7. Construction of $\mathsf{SnakeM}[\mathsf{KEM}, \mathsf{ID}, \mathsf{G}, \mathsf{H}]$.

3 SnakeMackerel (SnakeM)

We present a generic construction of an AKEM built from a split-ciphertext KEM and an identification scheme. Since we want to reuse part of the randomness to increase concrete efficiency, both underlying building blocks need to be compatible.

Definition 3.1 (Compatibility). *A split-ciphertext KEM* $\mathsf{KEM} = (\mathsf{Gen}, \mathsf{Encaps}_0, \mathsf{Encaps}_1, \mathsf{Decaps})$ *and an identification scheme* $\mathsf{ID} = (\mathsf{Gen}, \mathsf{Com}, \mathsf{Rsp})$ *are* compatible *iff* $\mathsf{Encaps}_0 = \mathsf{Com}$*, i.e., the algorithms are exactly the same.*

Our construction SnakeMackerel (short: SnakeM) can be found in Figure 7. More specifically, $\mathsf{SnakeM}[\mathsf{KEM}, \mathsf{ID}, \mathsf{G}, \mathsf{H}]$ is constructed from a split-ciphertext KEM KEM, an identification scheme ID, and two hash functions G and H where KEM and ID are compatible, $\mathsf{G} : \{0,1\}^* \to \mathsf{ChlSet}$, and $\mathsf{H} : \{0,1\}^* \to \mathcal{K}$ with $\mathcal{K}$ being the shared key space of the resulting AKEM. We include a seed in the secret key which is used for implicit rejections of KEM ciphertexts;[8] this not necessary to fulfill our security guarantees but allows for uniformly random outputs in case of invalid KEM ciphertexts which can be helpful in higher level protocols. We further use RspSpace to denote the response space of the ID scheme. In SnakeM, we encrypt the responses by interpreting every element in RspSpace as a bit-string and XORing them with a uniformly random padding value.

The correctness of SnakeM follows directly.

Lemma 3.2 (Correctness). *It holds that* $\delta_{\mathsf{SnakeM}[\mathsf{KEM},\mathsf{ID},\mathsf{G},\mathsf{H}]} \leq \delta_{\mathsf{KEM}} + \delta_{\mathsf{ID}}$.

Remark 3.3 (Public Key Reuse). One could further save space in the public key by using the same key pair for both roles, namely as sender and receiver. We note that this may be applicable to our final isogeny-based construction, although we would need a non-trivial notion of public-key compatibility in addition to the current compatibility notion, since the signature public key is only contained in (but not exactly distributed in the same way as) the encryption public key. Also, we expect security notions to be more contrived, which is why we leave a formal analysis for future work. Previously, public key reuse was also studied in [PSST11]. However, they rely on identity-based encryption (IBE) instead of standard PKE/KEM, and we are not aware of any isogeny-based IBE scheme.

[8] The final AKEM does not fully implicitly reject ciphertexts, i.e., it does still explicitly reject if the (partially encrypted) signature does not verify. Interestingly, it seems hard to even define authenticity for fully implicitly rejecting AKEMs (in a black-box way) since, intuitively, every ciphertext will yield some output unequal to $\bot$.

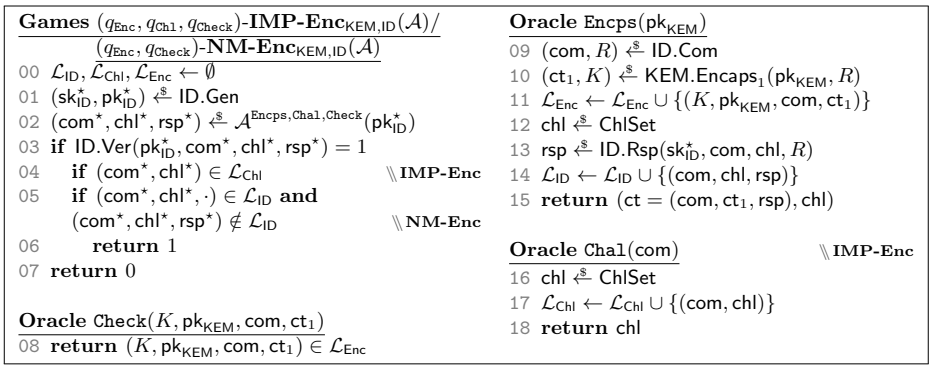

Fig. 8. Games defining IMP-Enc and NM-Enc for an adversary $\mathcal{A}$ making at most $q_{\mathtt{Enc}}$ queries to `Encps`, at most $q_{\mathtt{Chl}}$ to `Chal`, and at most $q_{\mathtt{Check}}$ to `Check`.

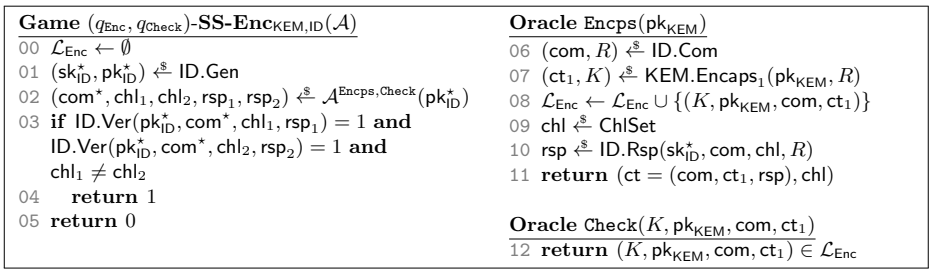

Fig. 9. Game defining SS-Enc for an adversary $\mathcal{A}$ making at most $q_{\mathtt{Enc}}$ queries to `Encps` and at most $q_{\mathtt{Check}}$ queries to `Check`.

3.1 New Notions

We first introduce some additional notions we need for the following security proofs. They all have in common that they define security for an ID scheme in the presence of an encapsulation oracle and a (split-ciphertext) KEM. This encapsulation oracle can be seen as a strengthening of a common transcript oracle for ID schemes alone. In addition to an ID transcript, it outputs the second part of a KEM encapsulation using the same commitment randomness as the ID commitment. We elaborate more on the necessity of this approach in Appendix F. Apart from this change, we define impersonation security (IMP) and special soundness (SS) as usual. Another property we need in our proofs is non-malleability (NM) of the response which is not covered by the other notions. It says that it should be hard, given an ID transcript containing commitment, challenge, and response, to come up with a different response such that the new response verifies together with the given commitment and challenge.

Definition 3.4 (IMP-Enc). *For a KEM* KEM *and an ID scheme* ID*,* impersonation under encapsulations *is defined via the game in Figure 8. We define the advantage of an adversary* $\mathcal{A}$ *as*

$$\mathsf{Adv}^{(q_{\mathtt{Enc}},q_{\mathtt{Chl}},q_{\mathtt{Check}})\text{-IMP-Enc}}_{\mathsf{KEM},\mathsf{ID}}(\mathcal{A}) := \Pr[(q_{\mathtt{Enc}}, q_{\mathtt{Chl}}, q_{\mathtt{Check}})\text{-}\mathbf{IMP\text{-}Enc}_{\mathsf{KEM},\mathsf{ID}}(\mathcal{A}) \Rightarrow 1].$$

Definition 3.5 (NM-Enc). *For a KEM* KEM *and an ID scheme* ID*,* non-malleability under encapsulations *is defined via the game in Figure 8. We define the advantage of an adversary* $\mathcal{A}$ *as*

$$\mathsf{Adv}^{(q_{\mathtt{Enc}},q_{\mathtt{Check}})\text{-NM-Enc}}_{\mathsf{KEM},\mathsf{ID}}(\mathcal{A}) := \Pr[(q_{\mathtt{Enc}}, q_{\mathtt{Check}})\text{-}\mathbf{NM\text{-}Enc}_{\mathsf{KEM},\mathsf{ID}}(\mathcal{A}) \Rightarrow 1].$$

Definition 3.6 (SS-Enc). *For a KEM* KEM *and an ID scheme* ID*,* special soundness under encapsulations *is defined via the game in Figure 9. We define the advantage of an adversary* $\mathcal{A}$ *as*

$$\mathsf{Adv}^{(q_{\mathtt{Enc}},q_{\mathtt{Check}})\text{-SS-Enc}}_{\mathsf{KEM},\mathsf{ID}}(\mathcal{A}) := \Pr[(q_{\mathtt{Enc}}, q_{\mathtt{Check}})\text{-}\mathbf{SS\text{-}Enc}_{\mathsf{KEM},\mathsf{ID}}(\mathcal{A}) \Rightarrow 1].$$

3.2 Security

We now give theorem statements for confidentiality, authenticity and deniability, along with proof sketches. The full proofs can be found in Appendices D.1 to D.3, respectively.

Theorem 3.7 (Insider CCA). *For any adversary $\mathcal{A}$ against* Ins-CCA *security of* SnakeM[KEM, ID, G, H] *making at most q_{G} random oracle queries to* G *and at most q_{H} random oracle queries to* H*, there exists an adversary $\mathcal{B}$ against* OW-KCA *such that*

$$\mathsf{Adv}^{(q_{\mathtt{Enc}},q_{\mathtt{Dec}})\text{-Ins-CCA}}_{\mathsf{SnakeM}[\mathsf{KEM},\mathsf{ID},\mathsf{G},\mathsf{H}]}(\mathcal{A}) \leq \mathsf{Adv}^{(q_{\mathsf{G}}+q_{\mathsf{H}})\text{-OW-KCA}}_{\mathsf{KEM}}(\mathcal{B}) + \delta_{\mathsf{SnakeM}}.$$

The full proof can be found in Appendix D.1.

Proof (Sketch). The proof is essentially based on the reduction to OW-KCA of the underlying KEM which embeds the one-wayness challenge in the AKEM challenge oracle and can recognize the correct key in any random oracle query. To make the simulation of the challenge oracle sound, the reduction can choose a random encryption of the response. This works because an adversary can only decrypt if they query one random oracle on the challenge KEM key which implies a success against the OW-KCA game. The check oracle from OW-KCA is further needed to answer decapsulation queries in conjunction with programming the random oracles appropriately. All other queries can be answered by the reduction since it can sample their own ID key pair. □

Theorem 3.8 (Insider Authenticity). *For any adversary $\mathcal{A}$ against* Ins-Aut *security of* SnakeM[KEM, ID, G, H] *making at most q_{G} random oracle queries to* G *and at most q_{H} random oracle queries to* H*, there exist an adversary $\mathcal{B}$ against* IMP-Enc*, an adversary $\mathcal{C}$ against* SS-Enc*, and an adversary $\mathcal{D}$ against* NM-Enc *such that*

$$\begin{aligned}\mathsf{Adv}^{(q_{\mathtt{Enc}},q_{\mathtt{Dec}})\text{-Ins-Aut}}_{\mathsf{SnakeM}[\mathsf{KEM},\mathsf{ID},\mathsf{G},\mathsf{H}]}(\mathcal{A}) \leq\ & \mathsf{Adv}^{(q_{\mathtt{Enc}},q_{\mathsf{G}},q_{\mathsf{G}}+q_{\mathsf{H}})\text{-IMP-Enc}}_{\mathsf{KEM},\mathsf{ID}}(\mathcal{B}) + \mathsf{Adv}^{(q_{\mathtt{Enc}},q_{\mathsf{G}}+q_{\mathsf{H}})\text{-SS-Enc}}_{\mathsf{KEM},\mathsf{ID}}(\mathcal{C}) \\ & + \mathsf{Adv}^{(q_{\mathtt{Enc}},q_{\mathsf{G}},q_{\mathsf{G}}+q_{\mathsf{H}})\text{-NM-Enc}}_{\mathsf{KEM},\mathsf{ID}}(\mathcal{D}) + \frac{q_{\mathtt{Enc}}\cdot q_{\mathsf{G}}}{|\mathsf{ChlSet}|} \\ & + q_{\mathtt{Enc}}\cdot\delta_{\mathsf{ID}} + q_{\mathtt{Enc}}(q_{\mathsf{G}}+q_{\mathsf{H}})\cdot\gamma_{\mathsf{ID}}\gamma_{\mathsf{KEM}},\end{aligned}$$

where ChlSet *is the challenge space of* ID*.*

The full proof can be found in Appendix D.2.

Proof (Sketch). The proof is mainly based on three similar reductions. Each captures an attack against the ID scheme; impersonation, special soundness, and non-malleability. If none of these cases occurs in the challenge query (and no random oracle collision either), the adversary has no significant advantage in winning the game. This is because the three properties together with collision resistance of the random oracle cover all cases in which the challenge oracle would output something meaningful (queries from a previous encapsulation query, for example, do not help to guess the challenge bit). To simulate the complete experiment, the reductions need to answer encapsulation queries which is possible due to their definition. Decapsulation queries can be answered since the reduction can sample their own challenge KEM key pair. □

In the previous theorem, we showed that SnakeM fulfills insider authenticity. As mentioned in [GJK24], **dishonest** receiver deniability cannot be achieved for such a strong authenticity property. However, they leave it as an open question if one can achieve **honest** receiver deniability together with the strongest notion of authenticity. In the remainder of the section, we show that SnakeM actually fulfills the strongest possible variant of honest receiver deniability based on rather weak assumptions.

Theorem 3.9 (Honest Receiver Deniability). *There exists a* PPT *simulator* Sim *such that for any* HR-Den *adversary* $\mathcal{A}$ *against* SnakeM[KEM, ID, G, H] *making at most* q_{G} *random oracle queries to* G *and at most* q_{H} *random oracle queries to* H*, there exists an adversary* $\mathcal{B}$ *against* OW *security such that*

$$\mathsf{Adv}^{\mathsf{HR\text{-}Den}}_{\mathsf{SnakeM}[\mathsf{KEM},\mathsf{ID},\mathsf{G},\mathsf{H}],\mathsf{Sim}}(\mathcal{A}) \leq (q_{\mathsf{G}} + q_{\mathsf{H}}) \cdot \mathsf{Adv}^{\mathsf{OW}}_{\mathsf{KEM}}(\mathcal{B}).$$

The full proof can be found in Appendix D.3.

Proof (Sketch). We can construct a simulator which computes an honest KEM encapsulation. Note that by the compatibility of KEM and ID scheme, the commitment algorithm is exactly the same as the first part of the encapsulation which makes the first two parts of the ciphertext perfectly correct. The last part of the ciphertext, which is the encryption of the response, can be simulated easily since a random element of the response space is indistinguishable from a real encryption if the adversary does not query the random oracle to get the corresponding encryption padding. However, if this happens, the adversary is able to break OW security of the underlying KEM because the random oracle input includes the KEM key. □

Remark 3.10. In the proof, we guess the random oracle query for which the reduction wins the OW game. Note that a tighter bound can be achieved when reducing to OW-KCA security.

4 POKÉ as Split-Ciphertext KEM

In this section we adapt (the four dimensional variant of) the POKÉ KEM [BM25] to a suitable split-ciphertext KEM as in Definition 2.7 that is compatible with the ID scheme underlying SQIsignHD. We start by describing the public parameters of POKÉ.

Let $p = 2^a \cdot c - 1$ be a B-SIDH prime with $N \mid (p^2 - 1)$ its smooth factor. Let $E_0/\mathbb{F}_{p^2}$ be the supersingular elliptic curve with $j(E_0) = 1728$, $\langle P_0, Q_0\rangle = E_0[2^a]$, $\langle R_0, S_0\rangle = E_0[N]$ and $\langle X_0, Y_0\rangle = E_0[D]$. We require pairwise coprimality between 2, N and D. The public parameters are described by the tuple $(p, E_0, P_0, Q_0, R_0, S_0, X_0, Y_0)$.[9]

Remark 4.1. The shape of the prime p varies slightly compared to [BM25], mainly to ensure compatibility with the SQIsignHD identification scheme. In particular, we incorporate the B-SIDH approach as part of the N-torsion is defined over $\mathbb{F}_{p^4}$ [Cos20]. Such an approach was already considered in [BM25] but not formally implemented.

In Figure 10 we describe how key generation, encapsulation and decapsulation are performed. Compared to [BM25], the presented version uses no KDF for key derivation. Additionally, we incorporate a different notation to make this section consistent with the rest of the paper (see also Figure 1). In particular, we rephrase POKÉ as a split-ciphertext KEM (Definition 2.7).

Security of POKÉ. The IND-CPA security of (the hashed version of) POKÉ was proven assuming the computational POKE (C-POKE) problem [BM25, Problem 7 in the ePrint version] and in the random oracle model. To instantiate SnakeM we require a slightly stronger assumption than this, namely OW-KCA security (Definition 2.9), which has an additional key checking oracle. Note that this assumption is also sufficient to prove IND-CCA security of the POKÉ KEM in the ROM (similar to the strong Diffie-Hellman assumption for hashed ElGamal in prime-order groups [ABR01]), which is why we get Insider-CCA security. We discuss the assumption in more detail in Section 6.1 and Appendix E.

[9] Note that all components can be derived deterministically from p which is why we omit them when defining advantage functions.

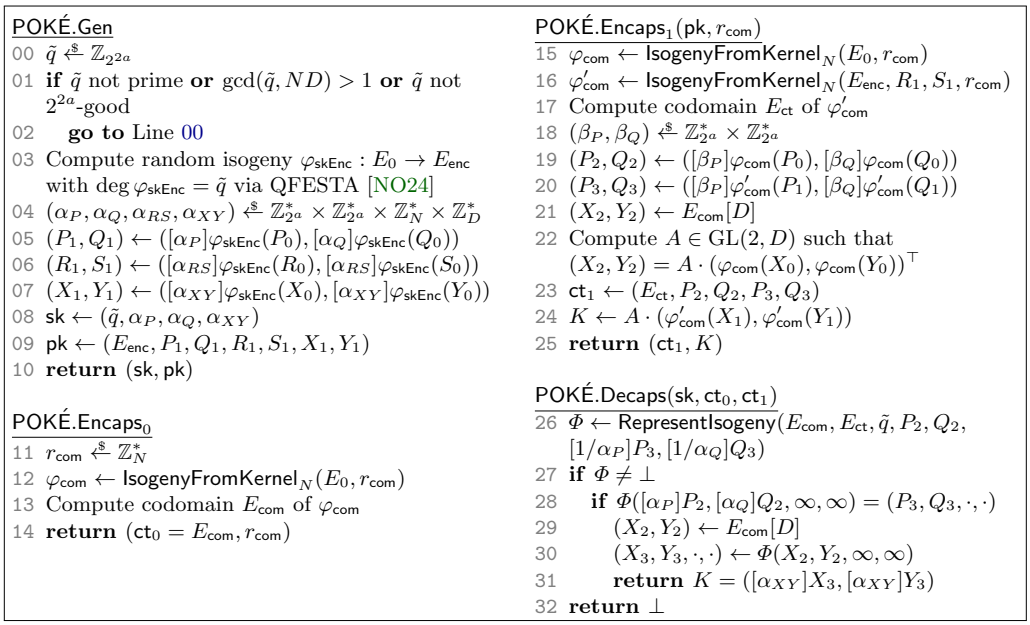

Fig. 10. The POKÉ scheme written as a split-ciphertext KEM.

5 SQIsignHD with Non-Malleability

In this section, we introduce a slightly modified version of the SQIsignHD identification scheme [DLRW24] to make it compatible with POKÉ. To this end, we first give a description of our variant SQIsignHD$^+$ in Section 5.1 and explain the differences between SQIsignHD and SQIsignHD$^+$. In Section 5.2 we prove that SQIsignHD$^+$ transcripts are non-malleable.

5.1 The SQIsignHD$^+$ Identification Scheme

Let p be a B-SIDH prime with $N \mid (p^2 - 1)$ its smooth factor. The SQIsignHD$^+$ identification scheme consists of the subroutines $(\mathsf{SQI.Gen}, \mathsf{SQI.Com}, \mathsf{SQI.Rsp}, \mathsf{SQI.Ver})$, which we present in Figure 11. The challenge set is $\mathsf{ChlSet} = \mathbb{Z}_N^*$ and, for $E \in \mathcal{SS}\ell(\mathbb{F}_{p^2})$, an element $r \in \mathsf{ChlSet}$ corresponds to the rational isogeny $\varphi_{\mathsf{chal}} \leftarrow \mathsf{IsogenyFromKernel}(E, r)$. Compared to SQIsignHD, there are a couple of changes that we address separately.

Changing the Underlying Prime. The original SQIsignHD uses a prime of the form $p = 2^f 3^{f'} c - 1$ with $2^f \approx 3^{f'} \approx 2^\lambda$, meaning that any curve $E/\mathbb{F}_{p^2}$ has accessible $2^f 3^{f'}$-torsion. The 2^f-torsion is then used for the response computation, whereas the $3^{f'}$-torsion is essentially used for the secret key, challenge and commitment. For security reasons, it is required that $\deg \varphi_{\mathsf{skSig}} \approx \deg \varphi_{\mathsf{com}} \approx p$ and, since $3^{f'} \approx \sqrt{p}$, some torsion point gymnastics are required to achieve such large degrees. For SQIsignHD$^+$, we instead propose to use a B-SIDH prime $p = 2^a c - 1$ (as per Definition 2.1) such that $N \mid (p^2 - 1)$ is a smooth factor of appropriate size. This way, all curves $E/\mathbb{F}_{p^4}$ have accessible $2^a N$-torsion, where the N-torsion now serves the same role as the $3^{f'}$-torsion in SQIsignHD. In particular, we can choose φ_{com} to be an isogeny of degree N.

Changing the Degree of the Challenge. In SQIsignHD the main requirement for the challenge isogeny φ_{chal} is that the challenge space has λ bits of entropy. This condition implies $\deg \varphi_{\mathsf{chal}} \approx 2^\lambda$, meaning that the $3^{f'}$-torsion is sufficient for the challenge computation. However, for SQIsignHD$^+$ we are already using a B-SIDH prime, meaning that we can use (a subgroup of) the rational N-torsion to compute φ_{chal}.

Making the Response Non-Malleable. In order to achieve strong authenticity guarantees, we require some form of non-malleability from SQIsignHD$^+$ transcripts. We achieve this

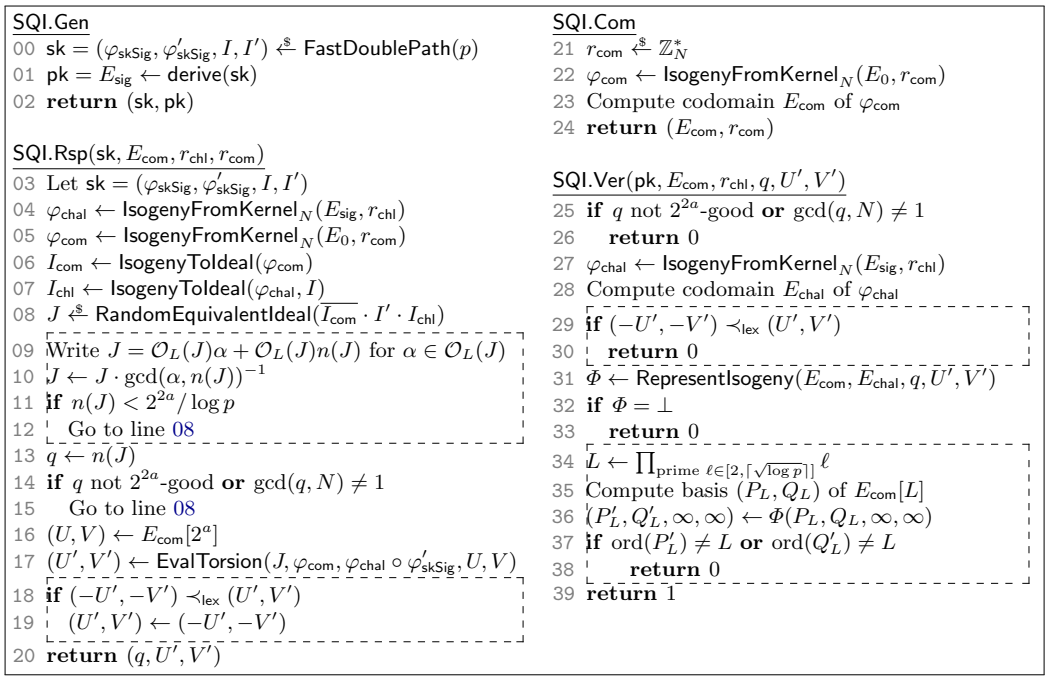

Fig. 11. The subroutines of the $\mathsf{SQIsignHD}^+$ identification scheme, including the necessary modification (in dashed boxes) to achieve response non-malleability. These modifications are further explained in Section 5.1.

non-malleability by making the response isogeny cyclic, effectively preventing an adversary to compute another valid response from an honestly generated one. Additionally, we reject response isogenies where the resulting degree is too small. We will now explain how these computations are carried out (cf. Figure 11) and give the corresponding security proof in Section 5.2.

In order to guarantee that the isogeny corresponding to the ideal J is cyclic, we need to ensure that $J \not\subseteq n\mathcal{O}_L(J)$ for all $n > 1$. Writing J in terms of an $\mathcal{O}_L$-basis as $J = \mathcal{O}_L(J)\alpha + \mathcal{O}_L(J)n(J)$ for some $\alpha \in \mathcal{O}_L(J)$, it follows that J is cyclic if and only if α and $n(J)$ have no common factor. Thus, dividing out $\gcd(\alpha, n(J))$ will result in an ideal representing a cyclic isogeny. Lastly, we reject those ideals that have a norm below a certain rather arbitrarily chosen bound (see also Remark 5.9). This is captured in the following lemma.

Lemma 5.1. *The output isogeny φ_{rsp} of $\mathsf{SQI.Rsp}$ is cyclic and has degree at least $2^{2a}/\log p$.*

Furthermore, we include some additional checks during verification. Concretely, we first check whether (U', V') is the lexicographically smaller tuple among $(\pm U', \pm V')$ (cf. Line 29). Additionally, we would like to certify that the response isogeny φ_{rsp} is cyclic. However, there is currently no efficient way to check the cyclicity of an isogeny φ embedded into a higher-dimensional isogeny Φ. Fortunately, for our construction it is sufficient to check whether φ_{rsp} factors through a small scalar multiplication $[\ell]$ bounded by $\sqrt{\log p}$. After fixing a basis (P_L, Q_L) of $E_{\mathsf{com}}[L]$ where L the product of all primes up to $\sqrt{\log p}$, this can be done efficiently by checking whether the order of both the images of P_L and Q_L is still L.

Remark 5.2. A valid concern might be that the rejection sampling in Line 11 leads to a longer signing time. We implemented the check within the $\mathsf{SQIsignHD}$ reference implementation[10] and determined that within 3000 trials, Line 11 rejected only 3 times. We therefore conclude that Line 11 does not lead to a significantly longer signing time. See also Remark 5.9. Furthermore, it is conceivable to run the rejection sampling a constant number of times to achieve consistent signing times.

[10] https://github.com/Pierrick-Dartois/SQISignHD-lib

$\underline{\mathsf{SimTrans}(\mathsf{pk}, \varphi_{\mathsf{hint}})}$ $\quad \backslash\backslash\ \mathsf{pk} = (E_{\mathsf{sig}}), \varphi_{\mathsf{hint}} \leftarrow \mathcal{H}(E_{\mathsf{sig}})$

00 $r_{\mathsf{chl}} \xleftarrow{\$} \mathbb{Z}_N^*$

01 $\varphi_{\mathsf{chal}} \leftarrow \mathsf{IsogenyFromKernel}(E_{\mathsf{sig}}, r_{\mathsf{chl}})$ $\quad \backslash\backslash\ \varphi_{\mathsf{chal}} : E_{\mathsf{sig}} \to E_{\mathsf{chal}}$

02 $\widehat{\varphi_{\mathsf{rsp}}} \leftarrow [\varphi_{\mathsf{chal}}]_* \varphi_{\mathsf{hint}}$ $\quad \backslash\backslash\ \widehat{\varphi_{\mathsf{rsp}}} : E_{\mathsf{chal}} \to E_{\mathsf{com}}$

03 $(\mathsf{com}, \mathsf{chl}) \leftarrow (E_{\mathsf{com}}, r_{\mathsf{chl}})$

04 $q \leftarrow \deg \widehat{\varphi_{\mathsf{rsp}}}$

05 Compute φ_{rsp} as the dual isogeny of $\widehat{\varphi_{\mathsf{rsp}}}$

06 $(U, V) \leftarrow E_{\mathsf{com}}[2^a]$ $\quad \backslash\backslash$ deterministic basis

07 $(U', V') \leftarrow (\varphi_{\mathsf{rsp}}(U), \varphi_{\mathsf{rsp}}(V))$

08 $\mathsf{rsp} \leftarrow (q, U', V')$

09 **return** $(\mathsf{com}, \mathsf{chl}, \mathsf{rsp})$

Fig. 12. The transcript simulator SimTrans.

5.2 Security

We now introduce the required hardness assumptions and models to prove the security of SQIsignHD$^+$. We adapted the language slightly compared to [DLRW24] for an easier exposition. In particular, some definitions are dedicated to B-SIDH primes as opposed to more general primes, however this change has no impact on the hardness of the assumptions or the correctness of the proofs. Furthermore, we apply the recent technique from [ABD$^+$25] to SQIsignHD$^+$, removing the somewhat artificial RUGDIO oracle in the process.

First, we introduce the hardness assumption on which the special soundness of the SQIsignHD$^+$ identification protocol relies. As in [ABD$^+$25] we require a *hint assisted* version of the One Endomorphism Problem. To this end, let $\mathcal{H}(E)$ be the distribution that outputs a uniformly random isogeny $\varphi : E \to E'$ of 2^{2a}-good degree $q \geq 2^{2a}/\log p$ prime to N such that:

1. E' is uniformly random among all supersingular elliptic curves over $\mathbb{F}_{p^2}$, and
2. The conditional distribution of φ given E' is uniform among all cyclic isogenies $\psi : E \to E'$ of 2^{2a}-good degree $q \geq 2^{2a}/\log p$ prime to N.

Compared to [ABD$^+$25] our definition of $\mathcal{H}$ includes a different distribution of isogenies. More concretely, the isogeny is cyclic and of certain minimum length. This is motivated by the fact that due to Lemma 5.1 the response isogenies output by SQIsignHD$^+$ are of this form. Furthermore, $\mathcal{H}$ only outputs a single isogeny as opposed to the two isogenies in [ABD$^+$25] since no auxiliary isogeny is required for the four-dimensional representation of the response in SQIsignHD$^+$.

Definition 5.3 (One Endomorphism Problem with Hints). *Let $\mathbb{F}_{p^2}$ be a finite field and $t \in \mathbb{N}$. We define the advantage of an adversary $\mathcal{A}$ winning* Hint-OneEnd *as*

$$\mathsf{Adv}^{t\text{-Hint-OneEnd}}_{p,\mathcal{H}}(\mathcal{A}) := \Pr\left[\tau \in \mathrm{End}(E) \setminus \mathbb{Z} \;\middle|\; \begin{array}{c} E \xleftarrow{\$} \mathcal{E}\ell\ell(\mathbb{F}_{p^2}) \\ \varphi_1, \ldots, \varphi_t \xleftarrow{\$} \mathcal{H}(E) \\ \tau \leftarrow \mathcal{A}(E, \varphi_1, \ldots, \varphi_t) \end{array}\right].$$

Having access to hints now allows us to construct a transcript simulator for SQIsignHD$^+$ (Figure 12). This simulator is not new and is implicitly contained in [DLRW24, Theorem 21]. To prove that the simulated transcripts are computationally indistinguishable from real transcripts, we need the following notion.

Definition 5.4 (Hint Indistinguishability). *Let p be a B-SIDH prime with $N \mid p^2 - 1$ and fix a starting curve E_0. Consider the game* **HNT-IND** *in Figure 13 for hint distribution $\mathcal{H}$. We define the advantage of an adversary $\mathcal{A}$ winning the* **HNT-IND** *game as*

$$\mathsf{Adv}^{t\text{-HNT-IND}}_{p,\mathcal{H}}(\mathcal{A}) := \left|\Pr[t\text{-}\mathbf{HNT\text{-}IND}_{p,\mathcal{H}}(\mathcal{A}) \Rightarrow 1] - \frac{1}{2}\right|.$$

Game t-HNT-IND$_{p,\mathcal{H}}(\mathcal{A})$

$b \xleftarrow{\$} \{0,1\}$
$E_1 \xleftarrow{\$} \mathcal{E}\ell\ell(\mathbb{F}_{p^2})$
for $i \in [t]$
  $r \xleftarrow{\$} \mathbb{Z}_N^*$
  $\varphi_i \leftarrow \mathsf{IsogenyFromKernel}(E_1, r)$ $\quad \backslash\backslash\, \varphi_i : E_1 \to E_{3,i}$
  if $b = 0$
    $r' \xleftarrow{\$} \mathbb{Z}_N^*$
    $\varphi_i' \leftarrow \mathsf{IsogenyFromKernel}(E_0, r')$ $\backslash\backslash\, \varphi_i' : E_0 \to E_{2,i}$
    Compute an efficient representation of a cyclic isogeny $\psi_i : E_{2,i} \to E_{3,i}$ of 2^{2a}-good degree $q \geq 2^{2a}/\log p$ prime to N
  else
    $\phi_i \xleftarrow{\$} \mathcal{H}(E_1)$
    $\widehat{\psi}_i \leftarrow [\varphi_i]_* \phi_i$ $\quad \backslash\backslash\, \widehat{\psi}_i : E_{3,i} \to E_{2,i}$
    Compute dual ψ_i of $\widehat{\psi}_i$
$b' \leftarrow \mathcal{A}(E_1, \varphi_1, \ldots, \varphi_t, \psi_1, \ldots, \psi_t)$
return $[\![b = b']\!]$

Fig. 13. Game defining HNT-IND.

Remark 5.5. In [ABD+25] the authors argue that (their version of) HNT-IND reduces to distinguishing the distribution of the curves $E_{2,i}$. Applied to our settings this amounts to distinguishing whether $E_{2,i}$ is N-isogenous to E_0. This problem is information-theoretically hard if $N \in \Omega(p^2)$ [DLRW24, Proposition 29] and believed to be computationally hard for $N \in \Theta(p)$. Note that in our setting $N \in \Theta(p^{3/2})$ (cf. Section 6.2).

The following two statements are simple adaptations of the respective correctness and security guarantees of SQIsignHD paired with the technique of [ABD+25], hence in the interest of space we refer the reader to [DLRW24, ABD+25] for the corresponding proofs.

Proposition 5.6. *The transcripts output by* SimTrans *(Figure 12) are valid transcripts with respect to the public key* pk*. Furthermore, the transcripts are computationally indistinguishable from honestly generated transcripts assuming the hardness of* HNT-IND*.*

Proposition 5.7. *The* SQIsignHD$^+$ *identification scheme is correct and special sound under the hardness of* Hint-OneEnd*. Furthermore, under the hardness of* HNT-IND *the* SQIsignHD$^+$ *identification scheme satisfies the Honest-Verifier Zero-Knowledge property.*

5.3 Non-Malleability of SQIsignHD$^+$

We now prove that the SQIsignHD$^+$ identification protocol satisfies response non-malleability (Definition 2.11). In the particular case of SQIsignHD$^+$, response non-malleability means that for an honestly generated representation of a response isogeny $\varphi_{\mathsf{rsp}} : E_{\mathsf{com}} \to E_{\mathsf{chal}}$ it should be hard to construct a second, distinct representation of an isogeny $\varphi'_{\mathsf{rsp}} : E_{\mathsf{com}} \to E_{\mathsf{chal}}$ having the same domain and codomain.

We highlight that for the response non-malleability proof the verification does not need to check that a response isogeny is cyclic (which would be a hard task). Instead, the verification only checks whether the response isogeny contains a small scalar factor $[\ell]$, for some prime $\ell \leq \sqrt{\log p}$ (cf. Line 37). Indeed, this is sufficient since in the response non-malleability game the forgery has to be with respect to an honestly generated transcript. Since for those transcripts the response isogenies are cyclic by Lemma 5.1, the verification only needs to make sure that the adversary did not append a scalar multiplication to an existing response.

Theorem 5.8 (SQIsignHD$^+$ is non-malleable). *For any adversary $\mathcal{A}$ against the response non-malleability of the* SQIsignHD$^+$ *identification scheme there exist adversaries $\mathcal{B}$ and $\mathcal{C}$ against* HNT-IND *and* Hint-OneEnd *with*

$$\mathsf{Adv}^{q_{\mathtt{Trans}}\text{-}\mathsf{Rsp}\text{-}\mathsf{NM}}_{\mathsf{SQIsignHD}^+}(\mathcal{A}) \leq \mathsf{Adv}^{q_{\mathtt{Trans}}\text{-}\mathsf{HNT}\text{-}\mathsf{IND}}_{p,\mathcal{H}}(\mathcal{B}) + \mathsf{Adv}^{q_{\mathtt{Trans}}\text{-}\mathsf{Hint}\text{-}\mathsf{OneEnd}}_{p,\mathcal{H}}(\mathcal{C}).$$

Proof. We prove the statements via a sequence of games. Let G_0 be the response non-malleability game $q_{\mathtt{Trans}}$-**Rsp-NM**$_{\mathsf{SQIsignHD}^+}$ with respect to the SQIsignHD$^+$ identification scheme.

Game G_1. This game is identical to G_0 except that the `Trans` oracle is simulated via SimTrans (Figure 12) and the provided hints from Hint-OneEnd. It easily follows from Proposition 5.6 that there exists an adversary $\mathcal{B}$ such that

$$|\Pr[\mathsf{G}_0(\mathcal{A}) \Rightarrow 1] - \Pr[\mathsf{G}_1(\mathcal{A}) \Rightarrow 1]| \leq \mathsf{Adv}_{p,\mathcal{H}}^{q_{\mathtt{Trans}}\text{-HNT-IND}}(\mathcal{B}).$$

Final reduction. We construct a reduction $\mathcal{C}$ to Hint-OneEnd such that

$$\Pr[\mathsf{G}_1(\mathcal{A}) \Rightarrow 1] \leq \mathsf{Adv}_{p,\mathcal{H}}^{q_{\mathtt{Trans}}\text{-Hint-OneEnd}}(\mathcal{C}).$$

$\mathcal{C}$ gets as input a Hint-OneEnd challenge E as well as hints $\varphi_1, \ldots, \varphi_{q_{\mathtt{Trans}}}$. It sets $(\mathsf{sk}, \mathsf{pk}) = (\bot, E)$ and sends pk to $\mathcal{A}$. It then simulates the `Trans` oracle via the provided hints, which can be done without having access to the secret key.

At some point the adversary $\mathcal{A}$ outputs a tuple $(\mathsf{com}, \mathsf{chl}, \mathsf{rsp}, \mathsf{rsp}^*)$ with $(\mathsf{com}, \mathsf{chl}, \mathsf{rsp}) \in \mathcal{L}$ and $(\mathsf{com}, \mathsf{chl}, \mathsf{rsp}^*) \notin \mathcal{L}$ (which implies $\mathsf{rsp} \neq \mathsf{rsp}^*$). In the particular case of SQIsignHD$^+$, the response is a tuple $(E_{\mathsf{com}}, q, U', V')$ that represents an isogeny $\varphi : E_{\mathsf{com}} \to E_{\mathsf{chal}}$. Thus, adversary $\mathcal{A}$ provides an alternative representation $(E_{\mathsf{com}}, q^*, U^*, V^*)$ for an isogeny $\varphi^* : E_{\mathsf{com}} \to E_{\mathsf{chal}}$ having the same domain and codomain. Since $\mathcal{C}$ knows efficient representations for both φ, φ^* it can compute the endomorphism $\tau = \varphi^* \circ \hat{\varphi} \in \mathrm{End}(E_{\mathsf{chal}})$. As long as τ is a non-trivial endomorphism, this readily breaks the Hint-OneEnd problem for the curve E_{chal} (which can further be transformed into an endomorphism of E_{sig}). Therefore the only thing left to prove is that τ is not a scalar.

Assume that $\tau = [k]$ for some $k \in \mathbb{Z}$. Since φ is cyclic by the definition of the hints and $\varphi^* \circ \hat{\varphi} = [k]$, necessarily $\varphi^* = [m]\hat{\varphi}$ for some $m \in \mathbb{Z}$. (Note that the case $\varphi^* = [-1] \circ \varphi$ is excluded because the signature must contain the lexicographically smaller tuple among $(\pm U', \pm V')$.) It follows that

$$2^{2a} \geq \deg \varphi^* > \deg \varphi \geq 2^{2a} / \log p,$$

which implies $m \leq \sqrt{\log p}$ (note that the first inequality follows from q^* being 2^{2a}-good). This small scalar multiplication is however detected during verification (cf. Line 37) which contradicts the fact that $(\mathsf{com}^*, \mathsf{chl}^*, \mathsf{rsp}^*)$ is an accepting transcript. We conclude that τ is a non-scalar endomorphism of E_{chal}.

Finally, reduction $\mathcal{C}$ returns $\omega = \widehat{\varphi_{\mathsf{chal}}} \circ \tau \circ \varphi_{\mathsf{chal}}$, where φ_{chal} is the isogeny contained in chl. It now follows that ω is a non-scalar endomorphism of E and thus a valid solution to Hint-OneEnd, which proves the statement. Collecting all the probabilities yields the desired advantage. □

Remark 5.9. The condition $\deg \varphi_{\mathsf{rsp}} \geq 2^{2a} / \log p$ can be relaxed to $2^{2a}/B$ for some $B \in \mathcal{O}(\log(p))$ without compromising security. Indeed, the main criterion is that the check in Lines 34 to 37 still has to be efficient.

6 Instantiating SnakeM with POKÉ and SQIsignHD$^+$

In this section we discuss various aspects of the concrete instantiation of SnakeM with POKÉ and SQIsignHD$^+$, which we call SnakeM-Iso.

We first highlight that POKÉ and SQIsignHD$^+$ satisfy compatibility as defined in Definition 3.1. Indeed, both Encaps_0 and SQI.Com sample a uniform N-isogeny $\varphi_{\mathsf{com}} : E_0 \to E_{\mathsf{com}}$ with kernel $\langle R_0 + [r_{\mathsf{com}}]S_0 \rangle$ and output the tuple $(E_{\mathsf{com}}, r_{\mathsf{com}})$. Furthermore, both protocols were adapted to be compatible with a B-SIDH prime p; the security requirements and the resulting shape of p are discussed in Section 6.2. Subsequently, the public and secret key of SnakeM-Iso are simply the concatenation of the corresponding POKÉ and SQIsignHD$^+$ keys, together with

a seed s:

$$\mathsf{sk} = (\underbrace{\varphi_{\mathsf{skSig}}, \varphi'_{\mathsf{skSig}}, I, I'}_{\mathsf{SQIsignHD}^+}, \underbrace{\widetilde{q}, \alpha_P, \alpha_Q, \alpha_{XY}}_{\mathsf{POKÉ}}, \mathsf{s})$$
$$\mathsf{pk} = (\underbrace{E_{\mathsf{sig}}}_{\mathsf{SQIsignHD}^+}, \underbrace{E_{\mathsf{enc}}, P_1, Q_1, R_1, S_1, X_1, Y_1}_{\mathsf{POKÉ}})$$

The ciphertext, on the other hand, is the tuple $(\mathsf{com}, \mathsf{ct}_1, \mathsf{ct}_{\mathsf{rsp}})$, where $\mathsf{ct}_{\mathsf{rsp}}$ corresponds to a symmetric encryption of the $\mathsf{SQIsignHD}^+$ response rsp. Concretely for SnakeM-Iso, we thus have

$$\mathsf{com} = E_{\mathsf{com}}, \qquad \mathsf{ct}_1 = (E_{\mathsf{ct}}, P_2, Q_2, P_3, Q_3), \qquad \mathsf{rsp} = (q, U', V').$$

6.1 Security Analysis

We now analyse the concrete hardness of various computational and statistical terms that appear in the bounds of Theorems 3.7 to 3.9.

The most fundamental hardness assumption in isogeny-based cryptography is the *Isogeny Path Problem* (IsoPath), which asks to find any isogeny between two elliptic curves $E, E' \in \mathcal{E}\ell\ell(\mathbb{F}_{p^2})$. The best known algorithms that solve IsoPath have complexity $\mathcal{O}(p^{1/2})$ [DG16], therefore we may assume $\mathsf{Adv}_p^{\mathsf{IsoPath}}(\mathcal{A}) \approx p^{-1/2}$ for any efficient adversary $\mathcal{A}$. Furthermore, IsoPath is tightly equivalent to OneEnd [PW24], and in [ABD+25] it is argued that Hint-OneEnd is as hard as OneEnd, therefore we can assume $\mathsf{Adv}_{p,\mathcal{H}}^{t\text{-}\mathsf{Hint\text{-}OneEnd}}(\mathcal{A}) \approx p^{-1/2}$ as well.

Confidentiality and Deniability. The only notion required for both confidentiality and deniability is OW-KCA for POKÉ (cf. Theorems 3.7 and 3.9). More specifically, this corresponds to the C-POKE problem in [BM25] with an additional key checking oracle `Check`, as defined in Definition 2.9. In [BM25], the authors argue that the currently best strategy to solve C-POKE involves either breaking IsoPath or guessing the shared key. The latter essentially consists of guessing three scalars in $\mathbb{Z}_D^*$, resulting in a complexity of $\mathcal{O}(D^3)$. We furthermore conjecture that the additional `Check` oracle does not offer any significant advantage to an adversary as adaptive attacks are effectively avoided (cf. Appendix E.3).

Authenticity. For the authenticity proof we require that $\mathsf{SQIsignHD}^+$ satisfies SS-Enc, IMP-Enc and NM-Enc (cf. Theorem 3.8). Apart from the additional `Encps` oracle, these notions correspond to SS, IMP-KOA and Rsp-NM. In [DLRW24] it is proved that SS reduces tightly to OneEnd. Furthermore, IMP-KOA security directly follows from SS security as shown, for instance, in [KMP16]. Unfortunately, this generic reduction has a square root loss due to the inherent rewinding step (see also Remark C.4). However, as shown in [AABN02] the quadratic loss can be accounted for by increasing the commitment entropy. Hence, for implicitly we assume that IMP-KOA and SS have the same bit security. Additionally, in Theorem 5.8 we prove that Rsp-NM reduces to Hint-OneEnd and HNT-IND. For the bit security of HNT-IND we refer to Remark 5.5, which leads us to assume that HNT-IND does not lower the bit security of $\mathsf{SQIsignHD}^+$. Lastly, in Appendix E we argue that the additional `Encps` oracle present in SS-Enc, IMP-Enc and NM-Enc does not help an adversary in any significant way.

Statistical Terms. In Theorem 3.8 there are some statistical terms to consider. We first observe that $|\mathsf{ChlSet}| = |\mathbb{Z}_N^*| \gg p$ using the well-known approximation of the Euler totient function [HW08, Theorem 326] and the definition of a B-SIDH prime. Since $\mathsf{SQIsignHD}^+$ is perfectly correct, we also have that the corresponding correctness error $\delta_{\mathsf{SQIsignHD}^+} = 0$. Lastly, we bound the spreadness of $\mathsf{SQIsignHD}^+$ and POKÉ by $\gamma_{\mathsf{SQIsignHD}^+} \leq 1$ and $\gamma_{\mathsf{POKÉ}} \leq \left(\prod_i (\ell_i + 1)\ell_i^{e_i - 1}\right)^{-1}$, where $N = \prod_i \ell_i^{e_i}$. The latter corresponds to the number of outgoing N-isogenies from E_0 and, under the assumption that only few N-isogenies lead to the same codomain, the number of possible curves E_{ct}.

6.2 The Prime Characteristic

Both $\mathsf{SQIsignHD}^+$ and POKÉ are designed to be compatible with a B-SIDH prime p where $(p^2 - 1) = 2^a N D$. Nevertheless, there are some constraints regarding the shape of the prime

to ensure correctness. In the case of $\mathsf{SQIsignHD}^+$, we require that $2^a > \sqrt{\deg \varphi_{\mathsf{rsp}}}$ to ensure that we can always represent the response isogeny φ_{rsp} as a 4-dimensional 2^a-isogeny using the meet-in-the-middle strategy from [Rob24, Appendix B.2]. Since $\deg \varphi_{\mathsf{rsp}} \leq \sqrt{8p}/\pi$ we get that $2^a \geq \sqrt[4]{p}$ is sufficient. Furthermore, we require that N is smooth in order to efficiently compute isogenies of degree N. Lastly, for point compression we mandate that D is smooth enough so that computing discrete logarithms in $E[D]$ is feasible, see also Section 6.3.

Regarding security, the discussion in Section 6.1 implies $p \in \Omega(2^{2\lambda})$ to ensure that Hint-OneEnd has λ bits of classical security. Furthermore, from Appendix E we gather that $N \in \Theta(2^{3\lambda})$ in order to argue the hardness of SS-Enc, IMP-Enc and NM-Enc. Additionally, we require that $D \in \Theta(2^{\lambda/3})$ as argued in [BM25] to ensure the hardness of OW-KCA for POKÉ. Lastly, all statistical terms are in the order of $2^{-2\lambda}$.

Despite the numerous requirements, the 255-bit prime

$$p = 2^{75} \cdot 3^{58} \cdot 233^2 \cdot 857^2 \cdot 2837^2 \cdot 28463^2 - 1$$

satisfies all these constraints for $\lambda = 128$. For higher security levels, the requirement that (p^2-1) is essentially completely smooth might be harder to satisfy for low smoothness bounds. However, by increasing the size of p slightly (e.g. $\log p \approx 2.4\lambda$), the required shape of the prime is very close to a classical SQIsign v1.0 prime. Indeed, a SQIsign v1.0 prime satisfies $2^a T \mid p^2-1$ where T is a smooth factor of size $T \approx p^{5/4}$. Since $p \approx 2^{2.4\lambda}$ this yields a smooth factor T of size $2^{3\lambda}$, matching the requirements mentioned above. The requirement that D is smooth enough for point compression is also easier to satisfy as $(p^2-1)/(2^a N)$ is now of size $2^{1.2\lambda}$, which is likely to contain a smooth enough part of size $2^{\lambda/3}$. For instance, the slightly larger 286-bit prime

$$p = 2^{75} \cdot 3^{54} \cdot 11^2 \cdot 13^2 \cdot 29^2 \cdot 31^2 \cdot 79^2 \cdot 151^2 \cdot 613^2 \cdot 1187^2 \cdot 9491^2 - 1$$

also satisfies all constraints for the $\lambda = 128$ bit security level while also featuring a lower smoothness bound for N. Therefore, finding suitable primes either for higher security levels or with better size/smoothness trade-offs can be done by adapting well-known methods for finding SQIsign v1.0 primes [BSC+23, SEMR24b].

6.3 Compactness

The public key of SnakeM-Iso consists of the two curves E_{enc} and E_{sig} as well as the torsion points $P_1, Q_1 \in E_{\mathsf{enc}}[2^a]$, $R_1, S_1 \in E_{\mathsf{enc}}[N]$ and $X_1, Y_1 \in E_{\mathsf{enc}}[D]$. Each curve is represented by its j-invariant with $2\log p$ bits. After choosing a deterministic basis (C_0, C_1) for $E_{\mathsf{enc}}[2^a]$, the points P_1, Q_1 are represented by $4\log(2^a)$ bits encoding the scalars $\alpha, \beta, \gamma, \delta \in \mathbb{Z}_{2^a}$ such that $P_1 = \alpha C_0 + \beta C_1$ and $Q_1 = \gamma C_0 + \delta C_1$. Similarly, the other pairs of torsion points can be represented with $4\log N$ and $4\log D$ bits, respectively. The 2^a-torsion points can be further compressed using the Weil pairing, provided we require $\alpha_P = \alpha_Q^{-1} \deg(\varphi_{\mathsf{skEnc}})^{-1}$ modulo 2^a. This way, we can compute the pairing of P_1 and Q_1 as

$$e_{2^a}(P_1, Q_1) = e_{2^a}(P_0, Q_0)^{\deg(\varphi_{\mathsf{skEnc}})\alpha_P \alpha_Q} = e_{2^a}(P_0, Q_0).$$

Thus, after computing $k \in \mathbb{Z}_{2^a}^*$ such that $e_{2^a}(P_0, Q_0) = e_{2^a}(C_0, C_1)^k$, we obtain the relation

$$\alpha\delta - \beta\gamma = k \mod 2^a,$$

from which it is possible to recover one of the four scalars describing P_1 and Q_1 in the deterministic basis.

In conclusion, since $p \in \Omega(2^{2\lambda})$, $2^a \in \Theta(2^{\lambda/2})$, $N \in \Theta(2^{3\lambda})$ and $D \in \Theta(2^{\lambda/3})$ by the definition of a B-SIDH prime, we get

$$|\mathsf{pk}| = 4\log p + 3\log(2^a) + 4\log N + 4\log D = \frac{137}{6}\lambda.$$

On the other hand, the ciphertext consists of the curves $E_{\mathsf{com}}, E_{\mathsf{ct}}$, the torsion points $P_2, Q_2 \in E_{\mathsf{com}}[2^a]$, $P_3, Q_3 \in E_{\mathsf{ct}}[2^a]$, $X_2, Y_2 \in E_{\mathsf{com}}[D]$, $U', V' \in E_{\mathsf{chal}}[2^a]$ and the degree $q \leq \sqrt{8p}/\pi$. First, by fixing a deterministic basis of the D-torsion on E_{com}, it is possible to omit the points X_2 and Y_2, as in this way the change of basis matrix becomes implicitly defined. The two curves require $2\log p$ bits each and the rest of the torsion points can be compressed by using the Weil pairing. Eventually, we get

$$|\mathsf{ct}| = 4\log p + 9\log(2^a) + \log(q) = \frac{27}{2}\lambda.$$

Acknowledgments. Jonas Janneck was supported by the European Union (ERC AdG REWORC - 101054911). Jonas Meers was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC 2092 CASA – 390781972. Massimo Ostuzzi is part of the Quantum-Safe Internet (QSI) ITN, which received funding from the European Union's Horizon-Europe programme as Marie Sklodowska-Curie Action (PROJECT 101072637 - HORIZON - MSCA-2021-DN-01). Massimo Ostuzzi is furthermore supported by the BMBF through project Quantum and Benchmarks for Resource Allocation (QuBRA).

References

AABN02. Michel Abdalla, Jee Hea An, Mihir Bellare, and Chanathip Namprempre. From identification to signatures via the Fiat-Shamir transform: Minimizing assumptions for security and forward-security. In Lars R. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 418–433. Springer, Berlin, Heidelberg, April / May 2002.

ABD+25. Marius A. Aardal, Andrea Basso, Luca De Feo, Sikhar Patranabis, and Benjamin Wesolowski. A complete security proof of SQIsign. In *CRYPTO 2025*, LNCS. Springer, Cham, August 2025.

ABF12. Afonso Arriaga, Manuel Barbosa, and Pooya Farshim. On the joint security of signature and encryption schemes under randomness reuse: Efficiency and security amplification. In Feng Bao, Pierangela Samarati, and Jianying Zhou, editors, *ACNS 12International Conference on Applied Cryptography and Network Security*, volume 7341 of *LNCS*, pages 206–223. Springer, Berlin, Heidelberg, June 2012.

ABH+21. Joël Alwen, Bruno Blanchet, Eduard Hauck, Eike Kiltz, Benjamin Lipp, and Doreen Riepel. Analysing the HPKE standard. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part I*, volume 12696 of *LNCS*, pages 87–116. Springer, Cham, October 2021.

ABR01. Michel Abdalla, Mihir Bellare, and Phillip Rogaway. The oracle Diffie-Hellman assumptions and an analysis of DHIES. In David Naccache, editor, *CT-RSA 2001*, volume 2020 of *LNCS*, pages 143–158. Springer, Berlin, Heidelberg, April 2001.

ADMP20. Navid Alamati, Luca De Feo, Hart Montgomery, and Sikhar Patranabis. Cryptographic group actions and applications. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part II*, volume 12492 of *LNCS*, pages 411–439. Springer, Cham, December 2020.

ADR02. Jee Hea An, Yevgeniy Dodis, and Tal Rabin. On the security of joint signature and encryption. In Lars R. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 83–107. Springer, Berlin, Heidelberg, April / May 2002.

AGKS05. Masayuki Abe, Rosario Gennaro, Kaoru Kurosawa, and Victor Shoup. Tag-KEM/DEM: A new framework for hybrid encryption and a new analysis of Kurosawa-Desmedt KEM. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 128–146. Springer, Berlin, Heidelberg, May 2005.

AJKL23. Joël Alwen, Jonas Janneck, Eike Kiltz, and Benjamin Lipp. The pre-shared key modes of HPKE. In Jian Guo and Ron Steinfeld, editors, *ASIACRYPT 2023, Part VI*, volume 14443 of *LNCS*, pages 329–360. Springer, Singapore, December 2023.

AR10. Jee Hea An and Tal Rabin. Security for signcryption: the two-user model. *Practical Signcryption*, pages 21–42, 2010.

BBLW22. Richard Barnes, Karthikeyan Bhargavan, Benjamin Lipp, and Christopher A. Wood. Hybrid Public Key Encryption. RFC 9180, February 2022.

BBR+23. Richard Barnes, Benjamin Beurdouche, Raphael Robert, Jon Millican, Emad Omara, and Katriel Cohn-Gordon. The Messaging Layer Security (MLS) Protocol. RFC 9420, July 2023.

BBS03. Mihir Bellare, Alexandra Boldyreva, and Jessica Staddon. Randomness re-use in multi-recipient encryption schemeas. In Yvo Desmedt, editor, *PKC 2003*, volume 2567 of *LNCS*, pages 85–99. Springer, Berlin, Heidelberg, January 2003.

BDD+24. Andrea Basso, Pierrick Dartois, Luca De Feo, Antonin Leroux, Luciano Maino, Giacomo Pope, Damien Robert, and Benjamin Wesolowski. SQIsign2D-West - the fast, the small, and the safer. In *ASIACRYPT 2024, Part III*, LNCS, pages 339–370. Springer, Singapore, December 2024.

BDK+18. Joppe W. Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS - kyber: A CCA-secure module-lattice-based KEM. In *2018 IEEE European Symposium on Security and Privacy*, pages 353–367. IEEE Computer Society Press, April 2018.

BF07. Manuel Barbosa and Pooya Farshim. Randomness reuse: Extensions and improvements. In Steven D. Galbraith, editor, *11th IMA International Conference on Cryptography and Coding*, volume 4887 of *LNCS*, pages 257–276. Springer, Berlin, Heidelberg, December 2007.

BFG+20. Jacqueline Brendel, Marc Fischlin, Felix Günther, Christian Janson, and Douglas Stebila. Towards post-quantum security for Signal's X3DH handshake. In Orr Dunkelman, Michael J. Jacobson, Jr., and Colin O'Flynn, editors, *SAC 2020*, volume 12804 of *LNCS*, pages 404–430. Springer, Cham, October 2020.

BLL24. Giacomo Borin, Yi-Fu Lai, and Antonin Leroux. Erebor and durian: Full anonymous ring signatures from quaternions and isogenies. *CiC*, 1(4):4, 2024.

BM25. Andrea Basso and Luciano Maino. POKÉ: A compact and efficient PKE from higher-dimensional isogenies. In *EUROCRYPT 2025, Part II*, LNCS, pages 94–123. Springer, Cham, June 2025.

BMP23. Andrea Basso, Luciano Maino, and Giacomo Pope. FESTA: Fast encryption from supersingular torsion attacks. In Jian Guo and Ron Steinfeld, editors, *ASIACRYPT 2023, Part VII*, volume 14444 of *LNCS*, pages 98–126. Springer, Singapore, December 2023.

BR93. Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In Dorothy E. Denning, Raymond Pyle, Ravi Ganesan, Ravi S. Sandhu, and Victoria Ashby, editors, *ACM CCS 93*, pages 62–73. ACM Press, November 1993.

BR06. Mihir Bellare and Phillip Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 409–426. Springer, Berlin, Heidelberg, May / June 2006.

BS20a. Mihir Bellare and Igors Stepanovs. Security under message-derived keys: Signcryption in iMessage. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part III*, volume 12107 of *LNCS*, pages 507–537. Springer, Cham, May 2020.

BS20b. Xavier Bonnetain and André Schrottenloher. Quantum security analysis of CSIDH. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part II*, volume 12106 of *LNCS*, pages 493–522. Springer, Cham, May 2020.

BSC+23. Giacomo Bruno, Maria Corte-Real Santos, Craig Costello, Jonathan Komada Eriksen, Michael Meyer, Michael Naehrig, and Bruno Sterner. Cryptographic smooth neighbors. In Jian Guo and Ron Steinfeld, editors, *ASIACRYPT 2023, Part VII*, volume 14444 of *LNCS*, pages 190–221. Springer, Singapore, December 2023.

BSZ02. Joonsang Baek, Ron Steinfeld, and Yuliang Zheng. Formal proofs for the security of signcryption. In David Naccache and Pascal Paillier, editors, *PKC 2002*, volume 2274 of *LNCS*, pages 80–98. Springer, Berlin, Heidelberg, February 2002.

Cal. Call for additional digital signature schemes for the post-quantum cryptography standardization process. https://csrc.nist.gov/Projects/pqc-dig-sig/round-1-additional-signatures. Accessed: 30.04.2024.

CCSC+24. Fabio Campos, Jorge Chávez-Saab, Jesús-Javier Chi-Domínguez, Michael Meyer, Krijn Reijnders, Francisco Rodríguez-Henríquez, Peter Schwabe, and Thom Wiggers. Optimizations and practicality of high-security CSIDH. *CiC*, 1(1):5, 2024.

CD23. Wouter Castryck and Thomas Decru. An efficient key recovery attack on SIDH. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part V*, volume 14008 of *LNCS*, pages 423–447. Springer, Cham, April 2023.

CHN+24. Daniel Collins, Loïs Huguenin-Dumittan, Ngoc Khanh Nguyen, Nicolas Rolin, and Serge Vaudenay. K-waay: Fast and deniable post-quantum X3DH without ring signatures. In

Davide Balzarotti and Wenyuan Xu, editors, *USENIX Security 2024*. USENIX Association, August 2024.

CLM+18. Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. CSIDH: An efficient post-quantum commutative group action. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part III*, volume 11274 of *LNCS*, pages 395–427. Springer, Cham, December 2018.

CMN21. Craig Costello, Michael Meyer, and Michael Naehrig. Sieving for twin smooth integers with solutions to the prouhet-tarry-escott problem. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part I*, volume 12696 of *LNCS*, pages 272–301. Springer, Cham, October 2021.

Cor08. Giuseppe Cornacchia. Su di un metodo per la risoluzione in numeri interi dell' equazione $\sum_{h=0}^{n} c_h x^{n-h} y^h = P$. Giornale di Matematiche di Battaglini, 1908.

Cos20. Craig Costello. B-SIDH: Supersingular isogeny Diffie-Hellman using twisted torsion. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part II*, volume 12492 of *LNCS*, pages 440–463. Springer, Cham, December 2020.

CSCJR22. Jorge Chávez-Saab, Jesús-Javier Chi-Domínguez, Samuel Jaques, and Francisco Rodríguez-Henríquez. The SQALE of CSIDH: sublinear Vélu quantum-resistant isogeny action with low exponents. *Journal of Cryptographic Engineering*, 12(3):349–368, September 2022.

CSD+23. Jorge Chavez-Saab, Maria Corte-Real Santos, Luca De Feo, Jonathan Komada Eriksen, Basil Hess, David Kohel, Antonin Leroux, Patrick Longa, Michael Meyer, Lorenz Panny, Sikhar Patranabis, Christophe Petit, Francisco Rodríguez Henríquez, Sina Schaeffler, and Benjamin Wesolowski. SQIsign. Technical report, National Institute of Standards and Technology, 2023. available at https://csrc.nist.gov/Projects/pqc-dig-sig/round-1-additional-signatures.

Deu41. Max Deuring. Die typen der multiplikatorenringe elliptischer funktionenkörper. *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, 14(1):197–272, Dec 1941.

DFK+23. Luca De Feo, Tako Boris Fouotsa, Péter Kutas, Antonin Leroux, Simon-Philipp Merz, Lorenz Panny, and Benjamin Wesolowski. SCALLOP: Scaling the CSI-FiSh. In Alexandra Boldyreva and Vladimir Kolesnikov, editors, *PKC 2023, Part I*, volume 13940 of *LNCS*, pages 345–375. Springer, Cham, May 2023.

DFPS23. Julien Devevey, Pouria Fallahpour, Alain Passelègue, and Damien Stehlé. A detailed analysis of Fiat-Shamir with aborts. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part V*, volume 14085 of *LNCS*, pages 327–357. Springer, Cham, August 2023.

DG16. Christina Delfs and Steven D. Galbraith. Computing isogenies between supersingular elliptic curves over $\mathbb{F}_p$. *DCC*, 78(2):425–440, 2016.

DHM+20. Ivan Damgård, Helene Haagh, Rebekah Mercer, Anca Nitulescu, Claudio Orlandi, and Sophia Yakoubov. Stronger security and constructions of multi-designated verifier signatures. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020, Part II*, volume 12551 of *LNCS*, pages 229–260. Springer, Cham, November 2020.

DKL+20. Luca De Feo, David Kohel, Antonin Leroux, Christophe Petit, and Benjamin Wesolowski. SQISign: Compact post-quantum signatures from quaternions and isogenies. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part I*, volume 12491 of *LNCS*, pages 64–93. Springer, Cham, December 2020.

DLRW24. Pierrick Dartois, Antonin Leroux, Damien Robert, and Benjamin Wesolowski. SQIsignHD: New dimensions in cryptography. In Marc Joye and Gregor Leander, editors, *EUROCRYPT 2024, Part I*, volume 14651 of *LNCS*, pages 3–32. Springer, Cham, May 2024.

ENS+23. Thomas Espitau, Thi Thu Quyen Nguyen, Chao Sun, Mehdi Tibouchi, and Alexandre Wallet. Antrag: Annular NTRU trapdoor generation - making mitaka as secure as falcon. In Jian Guo and Ron Steinfeld, editors, *ASIACRYPT 2023, Part VII*, volume 14444 of *LNCS*, pages 3–36. Springer, Singapore, December 2023.

FKPY22. Pierre-Alain Fouque, Paul Kirchner, Thomas Pornin, and Yang Yu. BAT: Small and fast KEM over NTRU lattices. *IACR TCHES*, 2022(2):240–265, 2022.

FMP23. Tako Boris Fouotsa, Tomoki Moriya, and Christophe Petit. M-SIDH and MD-SIDH: Countering SIDH attacks by masking information. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part V*, volume 14008 of *LNCS*, pages 282–309. Springer, Cham, April 2023.

GdKQ+24. Phillip Gajland, Bor de Kock, Miguel Quaresma, Giulio Malavolta, and Peter Schwabe. SWOOSH: Efficient lattice-based non-interactive key exchange. In Davide Balzarotti and Wenyuan Xu, editors, *USENIX Security 2024*. USENIX Association, August 2024.

GHJ25. Phillip Gajland, Vincent Hwang, and Jonas Janneck. Shadowfax: A deniability-preserving AKEM combiner. Cryptology ePrint Archive, Report 2025/154, 2025.

GJK24. Phillip Gajland, Jonas Janneck, and Eike Kiltz. Ring signatures for deniable AKEM: Gandalf's fellowship. In Leonid Reyzin and Douglas Stebila, editors, *CRYPTO 2024, Part I*, volume 14920 of *LNCS*, pages 305–338. Springer, Cham, August 2024.

GPST16. Steven D. Galbraith, Christophe Petit, Barak Shani, and Yan Bo Ti. On the security of supersingular isogeny cryptosystems. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part I*, volume 10031 of *LNCS*, pages 63–91. Springer, Berlin, Heidelberg, December 2016.

HW08. G H Hardy and E M Wright. *An introduction to the theory of numbers*. Oxford University Press, London, England, 6 edition, June 2008.

JK25. Joseph Jaeger and Akshaya Kumar. Analyzing group chat encryption in MLS, session, signal, and matrix. LNCS, pages 272–301. Springer, Cham, June 2025.

JKS24. Joseph Jaeger, Akshaya Kumar, and Igors Stepanovs. Symmetric signcryption and E2EE group messaging in keybase. In Marc Joye and Gregor Leander, editors, *EUROCRYPT 2024, Part III*, volume 14653 of *LNCS*, pages 283–312. Springer, Cham, May 2024.

KMP16. Eike Kiltz, Daniel Masny, and Jiaxin Pan. Optimal security proofs for signatures from identification schemes. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part II*, volume 9815 of *LNCS*, pages 33–61. Springer, Berlin, Heidelberg, August 2016.

Kup05. Greg Kuperberg. A subexponential-time quantum algorithm for the dihedral hidden subgroup problem. *SIAM Journal on Computing*, 35(1):170–188, 2005.

MMP+23. Luciano Maino, Chloe Martindale, Lorenz Panny, Giacomo Pope, and Benjamin Wesolowski. A direct key recovery attack on SIDH. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part V*, volume 14008 of *LNCS*, pages 448–471. Springer, Cham, April 2023.

MOXZ24. Tomoki Moriya, Hiroshi Onuki, Maozhi Xu, and Guoqing Zhou. Adaptive attacks against FESTA without input validation or constant-time implementation. In Markku-Juhani Saarinen and Daniel Smith-Tone, editors, *Post-Quantum Cryptography - 15th International Workshop, PQCrypto 2024, Part II*, pages 3–19. Springer, Cham, June 2024.

MP16. Moxie Marlinspike and Trevor Perrin. The X3DH Key Agreement Protocol. 2016.

Nio25. Guilhem Niot. Practical deniable post-quantum X3DH: A lightweight split-KEM for k-waay. In *ASIACCS 2025*. ACM Press, 2025.

NO24. Kohei Nakagawa and Hiroshi Onuki. QFESTA: Efficient algorithms and parameters for FESTA using quaternion algebras. In Leonid Reyzin and Douglas Stebila, editors, *CRYPTO 2024, Part V*, volume 14924 of *LNCS*, pages 75–106. Springer, Cham, August 2024.

Pei20. Chris Peikert. He gives C-sieves on the CSIDH. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part II*, volume 12106 of *LNCS*, pages 463–492. Springer, Cham, May 2020.

PFH+20. Thomas Prest, Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Ricosset, Gregor Seiler, William Whyte, and Zhenfei Zhang. FALCON. Technical report, National Institute of Standards and Technology, 2020. available at https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions.

PSST11. Kenneth G. Paterson, Jacob C. N. Schuldt, Martijn Stam, and Susan Thomson. On the joint security of encryption and signature, revisited. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 161–178. Springer, Berlin, Heidelberg, December 2011.

PW24. Aurel Page and Benjamin Wesolowski. The supersingular endomorphism ring and one endomorphism problems are equivalent. In Marc Joye and Gregor Leander, editors, *EUROCRYPT 2024, Part VI*, volume 14656 of *LNCS*, pages 388–417. Springer, Cham, May 2024.

Rob23. Damien Robert. Breaking SIDH in polynomial time. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part V*, volume 14008 of *LNCS*, pages 472–503. Springer, Cham, April 2023.

Rob24. Damien Robert. On the efficient representation of isogenies (a survey). Cryptology ePrint Archive, Report 2024/1071, 2024.
SEMR24a. Maria Corte-Real Santos, Jonathan Komada Eriksen, Michael Meyer, and Krijn Reijnders. AprèsSQI: Extra fast verification for SQIsign using extension-field signing. In Marc Joye and Gregor Leander, editors, *EUROCRYPT 2024, Part I*, volume 14651 of *LNCS*, pages 63–93. Springer, Cham, May 2024.
SEMR24b. Maria Corte-Real Santos, Jonathan Komada Eriksen, Michael Meyer, and Francisco Rodríguez-Henríquez. Finding practical parameters for isogeny-based cryptography. *CiC*, 1(3):39, 2024.
Sil09. Joseph H Silverman. *The Arithmetic of Elliptic Curves*. Graduate texts in mathematics. Springer, Dordrecht, 2009.
Ste24. Bruno Sterner. Towards optimally small smoothness bounds for cryptographic-sized smooth twins and their isogeny-based applications. LNCS, pages 178–202. Springer, Cham, August 2024.
Vél71. Jacques Vélu. Isogénies entre courbes elliptiques. *Comptes-Rendus de l'Académie des Sciences*, 273:238–241, 1971.
Voi21. John Voight. *Quaternion Algebras*. Graduate texts in mathematics. Springer, 2021.
Zhe97. Yuliang Zheng. Digital signcryption or how to achieve cost(signature & encryption) $\ll$ cost(signature) + cost(encryption). In Burton S. Kaliski, Jr., editor, *CRYPTO'97*, volume 1294 of *LNCS*, pages 165–179. Springer, Berlin, Heidelberg, August 1997.
ZI98. Yuliang Zheng and Hideki Imai. How to construct efficient signcryption schemes on elliptic curves. *Information processing letters*, 68(5):227–233, 1998.

A Further Related Work

A.1 APKE

Early literature mainly focuses on APKE instead of its KEM counterpart. The most fundamental generic constructions are Encrypt-and-Sign (EaS), Encrypt-then-Sign (EtS), and Sign-then-Encrypt (StE) [AR10, ADR02]. While EaS does not achieve security in any setting, for the two others it depends on considering outsider or insider security. The security of the first component, i.e., encrypt in EtS or sign in StE, can be enhanced by the last component when considering outsider security. In the insider setting, on the other hand, security of the first component might even not be preserved. For example, in EtS the underlying PKE can be CCA secure but in the insider setting an adversary can compute a new signature and can easily break the CCA security of the APKE. Hence we can neither hope for **Ins-CCA** security of EtS nor for **Ins-Aut** security of StE. While StE (fulfilling **Ins-CCA** and **Out-Aut**) could keep up with most of the other schemes in Table 1 from a security point of view, the problem is that it cannot be translated to the AKEM setting since it is unclear what to sign, as we do not have a message. Since there is a large efficiency gain by relying on hybrid encryption rather than direct (A)PKEs, we do not list this option in the table. We further do not add EtS due to the lack of **Ins-CCA**; however, we list the hashed adaption EtStH.

A.2 AKEM

Alwen et al. [AJKL23] proposed two constructions, denoted by EtStH-AKEM and NIKE-AKEM. The former, Encapsulate-then-Sign-then-Hash, is very similar to our construction, while the latter follows the construction paradigm of DH-AKEM [ABH+21] utilizing two NIKEs, one for confidentiality and one for (implicit) authentication. However, there is another black-box construction combining ideas of both which we think is worth mentioning and including in the comparison: Namely, one can rely on the KEM for confidentiality as in EtStH, but then use a NIKE for authentication. We denote the construction by Encapsulate-and-NIKE-then-Hash, or short EaNtH. In the prime-order group setting, there is no advantage in such a construction because the NIKE-AKEM construction is strictly better. Computation time and ciphertext sizes are at least as good as for EaNtH but the public key of EaNtH is double the size; it would consist of the public key of the KEM and of the NIKE, whereas the

NIKE-AKEM only needs one NIKE public key. In the PQ secure constructions, however, the situation is different. On the one hand, we only have very large lattice-based NIKEs and replacing one NIKE with a KEM improves the size significantly. On the other hand, we have compact isogeny-based NIKEs with the disadvantage of a high computational overhead [CCSC+24].

B Isogeny Preliminaries

B.1 Isogenies and the Deuring Correspondence

In this section, we recall some preliminaries about isogenies and their relation to quaternion algebras through the Deuring correspondence. For a general introduction to either topic we refer the reader to Silverman [Sil09] and Voight [Voi21], respectively.

Isogenies. Let q be a prime power and E be an elliptic curve defined over $\mathbb{F}_q$. We denote the point at infinity with ∞_E. For an extension field $K \supseteq \mathbb{F}_q$ we denote the set of K-rational points by $E(K)$. For an integer n we denote the multiplication-by-n map by $[n]$. Its kernel is the *n-torsion subgroup* $E[n] = \{P \in E : [n]P = \infty_E\}$ and we call an elliptic curve *supersingular* if $E[p] = \{\infty_E\}$. The set of all supersingular elliptic curves is denoted by $\mathcal{Ell}(\mathbb{F}_q)$.

An isogeny is a morphism $\varphi : E \to E'$ between elliptic curves E, E' such that $\varphi(\infty_E) = \infty_{E'}$. The degree of φ is its degree as a morphism and we call φ *separable* if $\gcd(p, \deg \varphi) = 1$. In this work, we will only consider separable isogenies. We call two elliptic curves *isogenous* if there exists an isogeny between them. An isogeny is an isomorphism of elliptic curves if it has an inverse (which may be defined over the algebraic closure of $\mathbb{F}_q$). In that case, the inverse is again an isogeny. Isomorphic curves have the same *j-invariant*, which is a simple algebraic expression in the coefficients of the curve equation. Thus, one can check whether two elliptic curves are isomorphic by comparing their j-invariant.

An isogeny from E to itself is called *endomorphism*. The set $\mathrm{End}(E)$ of endomorphisms of E (defined over the algebraic closure of the base field) forms a ring under addition and composition and it is thus called the *endomorphism ring*. The multiplication-by-n map $[n]$ is always an endomorphism of E, hence $\mathbb{Z}$ always embeds in $\mathrm{End}(E)$. Any isogeny $\varphi : E \to E'$ is also a group homomorphism from E to E' with finite kernel. In the case where φ is separable we have $\deg \varphi = |\ker \varphi|$. Conversely, any finite subgroup $G \subset E$ corresponds to a separable isogeny $\varphi : E \to E'$ with kernel $\ker \varphi = G$, where φ and E' are unique up to post-composition with an isomorphism. Since E' is essentially uniquely determined by $\ker \varphi$, we will write $E' = E/G$. One can compute φ and E/G via Vélu's formula [Vél71], which can be evaluated in polynomial time in the size of the kernel.

Quaternion Algebras. For a prime p, let $\mathcal{B}_{p,\infty}$ be the unique (up to isomorphism) quaternion algebra ramified at p and ∞. If $p \equiv 3 \mod 4$, we have $\mathcal{B}_{p,\infty} = \mathbb{Q} + i\mathbb{Q} + j\mathbb{Q} + k\mathbb{Q}$, where $i^2 = -1$, $j^2 = -p$ and $k = ij = -ji$. Every quaternion algebra has a *canonical involution* that sends $\alpha = a + bi + cj + dk$ to its *conjugate* $\bar{\alpha} = a - bi - cj - dk$. We further define the *reduced norm* and *reduced trace* as $\mathrm{nrd}(\alpha) = \alpha\bar{\alpha}$ and $\mathrm{tr}(\alpha) = \alpha + \bar{\alpha}$, respectively.

A *fractional ideal* I is a $\mathbb{Z}$-lattice of full rank in $\mathcal{B}_{p,\infty}$. The *norm* $n(I)$ of I is defined as the generator of the ideal $\{\mathrm{nrd}(\alpha) : \alpha \in I\}$. An *order* $\mathcal{O}$ is a subring of $\mathcal{B}_{p,\infty}$ that is also an ideal. We call $\mathcal{O}$ a *maximal* order if it is not properly contained in any other order. The left order of a fractional ideal I is defined as $\mathcal{O}_L(I) = \{\alpha \in \mathcal{B}_{p,\infty} : \alpha I \subset I\}$ and similar for the right order $\mathcal{O}_R(I)$. A fractional ideal is integral if its contained in its left (or right) order. Any integral ideal can be written as $I = \mathcal{O}_L(I)n(I) + \mathcal{O}_L(I)\alpha$, for some $\alpha \in \mathcal{O}_L(I)$ such that $\gcd(\mathrm{nrd}(\alpha), n(I)^2) = n(I)$. An (integral) ideal I is called an *$\mathcal{O}\mathcal{O}'$-connecting ideal* if $\mathcal{O}_L(I) = \mathcal{O}$ and $\mathcal{O}_R(I) = \mathcal{O}'$.

For two ideals I, J with $\mathcal{O}_L(J) = \mathcal{O}_R(I)$ we define their product IJ as the product of all pairs in $I \times J$. The ideal norm then satisfies $n(IJ) = n(I)n(J)$. An ideal is invertible if there exists an ideal I^{-1} such that $II^{-1} = \mathcal{O}_L(I)$ and $I^{-1}I = \mathcal{O}_R(I)$. We define the conjugate $\bar{I}$ as the ideal containing all the conjugates of elements in I, which furthermore satisfies $I\bar{I} = n(I)\mathcal{O}_L(I)$ and $\bar{I}I = n(I)\mathcal{O}_R(I)$ when I is invertible. Lastly, two ideals I, J are called *equivalent* if there

exists $\beta \in \mathcal{B}_{p,\infty}^*$ such that $I = J\beta$. This yields an equivalence class of left $\mathcal{O}$-ideals and the set of such equivalence classes is denote by $cl(\mathcal{O})$.

Deuring Correspondence. In [Deu41], Deuring proved that the endomorphism ring $\mathrm{End}(E)$ of a supersingular elliptic curve over $\mathbb{F}_{p^2}$ is isomorphic to a maximal order $\mathcal{O}$ in the quaternion algebra $\mathcal{B}_{p,\infty}$. Moreover, this correspondence is a category equivalence, for which an isogeny $\varphi : E \to E'$ corresponds to an $\mathcal{O}\mathcal{O}'$-connecting ideal I_φ with $\mathcal{O} \cong \mathrm{End}(E)$ and $\mathcal{O}' \cong \mathrm{End}(E')$. Furthermore, composition of isogenies corresponds to ideal multiplication, the dual isogeny corresponds to the conjugate ideal and $n(I_\varphi) = \deg \varphi_I$.

B.2 Isogeny Representations

Recently, new ways of representing an isogeny have emerged. This leads to the notion of an *efficient isogeny representation*. We present the definition found in [BDD+24].

Definition B.1 (Efficient Isogeny Representation). *Let $\mathcal{V}, \mathcal{E}$ be two algorithms. Furthermore, let $\varphi : E \to E'$ be an isogeny of degree d defined over $\mathbb{F}_q$. An* efficient representation *of φ (with respect to $\mathcal{V}$ and $\mathcal{E}$) is a bit string $D_\varphi \in \{0,1\}^*$ of length $\mathcal{O}(\mathrm{poly}\log(dq))$ such that:*

- *$\mathcal{V}(E, E', d, D_\varphi)$ returns in time $\mathcal{O}(\mathrm{poly}\log(dq))$ whether D_φ is a valid encoding of a d-isogeny between E and E', and*
- *$\mathcal{E}(E, E', d, D_\varphi, P)$ returns in time $\mathcal{O}(\mathrm{poly}(k\log(dq)))$ the image $\varphi(P)$ of a point $P \in E(\mathbb{F}_{q^k})$.*

In this paper, we use three specific instances of an isogeny representation: the kernel generator representation, the ideal representation and the High-Dimensional (HD) representation (as defined by Robert [Rob24]). The latter representation is universal, meaning that any other efficient representation can be transformed into the HD representation. We informally introduce these representations and refer the reader to [Rob24] for further details. In all cases, let $\varphi : E \to E'$ be an isogeny of degree d.

- **Kernel Generator Representation:** The isogeny φ is represented by two points $P, Q \in E$ (which might be defined over a field extension) such that $\ker \varphi = \langle P, Q\rangle$. If φ is cyclic then $Q = \infty$ and it follows that $\mathrm{ord}(P) = d$.
- **Ideal Representation:** Let $\mathcal{O} \cong \mathrm{End}(E)$ be the endomorphism ring of E. The isogeny φ is represented as an (integral) left ideal $I \subseteq \mathcal{O}$ that has right order $\mathcal{O}' \cong \mathrm{End}(E')$ and norm $n(I) = d$.
- **HD Representation:** The isogeny φ is represented as the tuple (E', d, U', V') where $(U', V') = (\varphi(U), \varphi(V))$ for a deterministically chosen basis (U, V) of $E[2^a]$ such that $2^a > \sqrt{d}$.

Each representation comes with its own set of advantages and disadvantages. Most importantly, the kernel generator representation is only efficient for smooth d. On the other hand, if $\mathrm{End}(E)$ is unknown it is currently the only representation that allows randomly sampling an isogeny with domain E. In addition, *computing* an HD representation for φ currently requires knowledge of $\mathrm{End}(E)$, whereas *evaluating* φ (and thus also verifying the encoding) can be done efficiently without $\mathrm{End}(E)$ once the representation is known.

B.3 Isogeny Subroutines

In this section we introduce some subroutines that carry out isogeny computations required for SQIsignHD and POKÉ, many of which were introduced in earlier works [DLRW24, CSD+23, BM25]. We will only informally describe these subroutines and refer to [DLRW24, CSD+23, BM25] for further details on their implementation.

- $\mathsf{IsogenyFromKernel}_N(E, P, Q, d)$: Takes as input an elliptic curve E, two points $P, Q \in E[N]$ and an integer $d \in \mathbb{Z}_N^*$ and outputs an efficient representation of the isogeny $\varphi : E \to E/G$ with kernel $G = \langle P + [d]Q\rangle$. If P, Q correspond to a deterministic basis of $E[N]$ they may be omitted from the input.

Game IMP-KOA$_{\mathsf{ID}}(\mathcal{A})$	**Oracle** $\mathtt{Chal}(\mathsf{com})$ \\ max one query
00 $(\mathsf{sk},\mathsf{pk}) \xleftarrow{\$} \mathsf{Gen}$	04 $\mathsf{C} \leftarrow 1$
01 $\mathsf{rsp} \leftarrow \mathcal{A}^{\mathtt{Chal}}(\mathsf{pk})$	05 $\mathsf{chl} \xleftarrow{\$} \mathsf{ChlSet}$
02 **if** $\mathsf{C} \neq 1$ **return** 0	06 **return** chl
03 **return** $\mathsf{Ver}(\mathsf{pk},\mathsf{com},\mathsf{chl},\mathsf{rsp})$	

Fig. 14. Game defining IMP-KOA for an ID scheme $\mathsf{ID} = (\mathsf{Gen},\mathsf{Com},\mathsf{Rsp},\mathsf{Ver})$ with challenge set ChlSet.

- $\mathsf{RepresentIsogeny}(E_1,E_2,q,U,V,U',V')$: Takes as input two elliptic curves E_1,E_2, a positive integer q and four points $U,V \in E_1[2^a]$, $U',V' \in E_2$ where $(U',V') = (\varphi(U),\varphi(V))$ for a q-isogeny $\varphi : E_1 \to E_2$. It outputs a 4-dimensional 2^{2a}-isogeny $\Phi : E_1^2 \times E_2^2 \to E_2^2 \times E_1^2$ such that $\varphi = \pi \circ \Phi \circ \iota$, where $\iota : P \in E_1 \mapsto (P,\infty,\infty,\infty) \in E_1^2 \times E_2^2$ and π is the projection $(P_1,P_2,P_3,P_4) \in E_2^2 \times E_1^2 \mapsto P_1 \in E_2$. If the input is not a valid representation for such an isogeny, it outputs $\perp$. If U,V correspond to a deterministic basis of $E_1[2^a]$ they may be omitted from the input.
- $\mathsf{FastDoublePath}(p)$: Takes as input a prime p (thus defining the endomorphism ring $\mathrm{End}(E_0)$ of the curve $E_0/\mathbb{F}_{p^2}$ with $j(E_0) = 1728$) and outputs two cyclic isogenies $\varphi,\varphi' : E_0 \to E_1$ of degree dividing 2^{2a} and N^2 respectively as well as their corresponding ideals I,I'.
- $\mathsf{IsogenyToIdeal}(\varphi,I)$: Takes as input an isogeny $\varphi : E_1 \to E_2$ of smooth degree given in kernel representation as well as an ideal I with left order $\mathcal{O}_0$, right order $\mathcal{O}_1 \cong \mathrm{End}(E_1)$ and norm coprime to $\deg\varphi$. It outputs the ideal $I_\varphi \subseteq \mathcal{O}_1$ corresponding to φ. If $E_1 \cong E_0$ then the second input can be left empty.
- $\mathsf{RandomEquivalentIdeal}(I)$: Takes as input a $\mathcal{O}$-left ideal I and outputs an equivalent ideal $J \sim I$ that is uniformly distributed amongst all equivalent $\mathcal{O}$-left ideals of norm less than 2^a.
- $\mathsf{EvalTorsion}(I,\varphi,\psi,U,V)$: Takes as input an ideal I corresponding to an isogeny $\varphi_I : E_1 \to E_2$ of non-smooth degree, a basis (U,V) of $E_1[2^a]$ and two smooth isogenies $\varphi : E_0 \to E_1$, $\psi : E_0 \to E_2$ of degree coprime to 2 and outputs $(\varphi_I(U),\varphi_I(V))$.

C Further Security Notions for ID Schemes

For completeness, we introduce the following standard security notions for identification schemes.

Definition C.1 (Special Soundness (SS)). *Let* ID *be an identification scheme. We define the advantage of an adversary* $\mathcal{A}$ *as*

$$\mathsf{Adv}^{\mathsf{SS}}_{\mathsf{ID}}(\mathcal{A}) := \Pr\left[\begin{array}{c}\mathsf{Ver}(\mathsf{pk},\mathsf{com},\mathsf{chl},\mathsf{rsp}) = 1 \\ \wedge\, \mathsf{Ver}(\mathsf{pk},\mathsf{com},\mathsf{chl}',\mathsf{rsp}') = 1 \\ \wedge\, \mathsf{chl} \neq \mathsf{chl}'\end{array}\middle|\begin{array}{r}(\mathsf{pk},\mathsf{sk}) \xleftarrow{\$} \mathsf{Gen} \\ (\mathsf{com},\mathsf{chl},\mathsf{chl}',\mathsf{rsp},\mathsf{rsp}') \leftarrow \mathcal{A}(\mathsf{pk})\end{array}\right] .$$

Definition C.2 (Impersonation against Key Only Attack (IMP-KOA)). *Let* ID *be an identification scheme with challenge set* ChlSet*. Consider the game* **IMP-KOA**$_{\mathsf{ID}}$ *in Figure 14. We define the advantage of an adversary* $\mathcal{A}$ *as*

$$\mathsf{Adv}^{\mathsf{IMP\text{-}KOA}}_{\mathsf{ID}}(\mathcal{A}) := \Pr[\mathbf{IMP\text{-}KOA}_{\mathsf{ID}}(\mathcal{A}) \Rightarrow 1] .$$

Both security notions are related via the following theorem.

Theorem C.3 ($\mathsf{SS} \Rightarrow$ IMP-KOA [KMP16])**.** *Let* ID *be an identification scheme with challenge set* ChlSet*. For any adversary* $\mathcal{A}$ *against* IMP-KOA *there exists an adversary* $\mathcal{B}$ *against* SS *such that*

$$\mathsf{Adv}^{\mathsf{SS}}_{\mathsf{ID}}(\mathcal{B}) \geq \mathsf{Adv}^{\mathsf{IMP\text{-}KOA}}_{\mathsf{ID}}(\mathcal{A})\left(\mathsf{Adv}^{\mathsf{IMP\text{-}KOA}}_{\mathsf{ID}}(\mathcal{A}) - \frac{1}{|\mathsf{ChlSet}|}\right).$$

Remark C.4. Theorem C.3 implies a square-root loss when bounding the advantage of an adversary against IMP-KOA via the advantage of an adversary against SS.

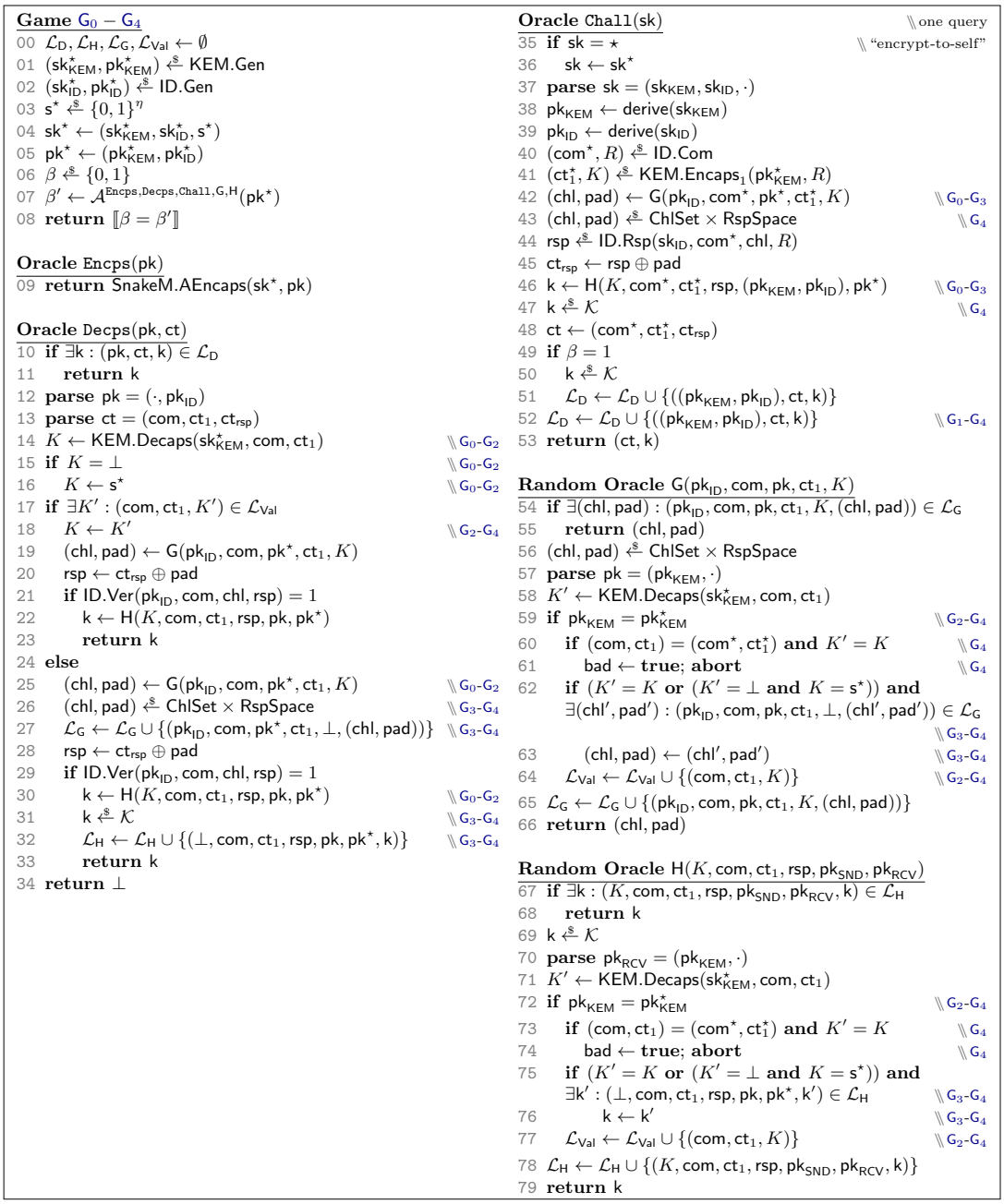

Fig. 15. Games for the proof of Theorem 3.7.

D Omitted Proofs

D.1 Proof of Theorem 3.7

Theorem 3.7 (Insider CCA). *For any adversary $\mathcal{A}$ against* Ins-CCA *security of* SnakeM[KEM, ID, G, H] *making at most q_G random oracle queries to* G *and at most q_H random oracle queries to* H*, there exists an adversary $\mathcal{B}$ against* OW-KCA *such that*

$$\mathsf{Adv}^{(q_{\mathtt{Enc}}, q_{\mathtt{Dec}})\text{-Ins-CCA}}_{\mathsf{SnakeM[KEM,ID,G,H]}}(\mathcal{A}) \leq \mathsf{Adv}^{(q_G+q_H)\text{-OW-KCA}}_{\mathsf{KEM}}(\mathcal{B}) + \delta_{\mathsf{SnakeM}}.$$

Proof. Let $\mathcal{A}$ be an adversary in the $(q_{\mathtt{Enc}}, q_{\mathtt{Dec}})$-Ins-CCA$_{\mathsf{SnakeM}}$ game. We consider the sequence of games in Figure 15.

Game G_0. We start with the original Ins-CCA game for SnakeM.

$$\mathsf{Adv}^{(q_{\mathrm{Enc}},q_{\mathrm{Dec}})\text{-Ins-CCA}}_{\mathsf{SnakeM}[\mathsf{KEM},\mathsf{ID},\mathsf{G},\mathsf{H}]}(\mathcal{A}) = \left|\Pr[\mathsf{G}_0^{\mathcal{A}} \Rightarrow 1] - \frac{1}{2}\right| .$$

Game G_1. In the challenge oracle, we add an element to set $\mathcal{L}_{\mathsf{D}}$ not only in case $\beta = 1$ but also in case $\beta = 0$. This change is only distinguishable if the scheme is not correct, hence we have

$$\Pr\left[\mathsf{G}_0^{\mathcal{A}} \Rightarrow 1\right] = \Pr\left[\mathsf{G}_1^{\mathcal{A}} \Rightarrow 1\right] \leq \delta_{\mathsf{SnakeM}}.$$

Game G_2. We introduce list $\mathcal{L}_{\mathsf{Val}}$ which stores valid KEM tuples for $\mathsf{pk}^{\star}_{\mathsf{KEM}}$ that are queried to G and H. Note that this includes those tuples created during `Encps` queries. If there already exists a matching tuple in the list, we take the corresponding KEM key K in the execution of a decapsulation query. The changes are only conceptual:

$$\Pr\left[\mathsf{G}_1^{\mathcal{A}} \Rightarrow 1\right] = \Pr\left[\mathsf{G}_2^{\mathcal{A}} \Rightarrow 1\right] .$$

Game G_3. In this game we prepare for our final reduction and simulate `Decps` queries without using the secret key $\mathsf{sk}^{\star}_{\mathsf{KEM}}$. To this end, the game needs to handle decapsulation queries without an entry in $\mathcal{L}_{\mathsf{Val}}$. If there is no corresponding element in $\mathcal{L}_{\mathsf{Val}}$, random oracles G and H were not queried on the required inputs. Hence, the game can choose an arbitrary output $(\mathsf{chl}, \mathsf{pad})$ of G randomly and proceed with the decapsulation oracle without knowledge of K. The output is stored together with the known parts of the input, i.e. everything except K, to potentially patch the RO later (Line 27). If the ID tuple verifies, we also need to choose an output for RO H and store it for later use (Line 32). Note that this is sound since H was not queried as well as argued before. Later on, the game might have to patch the random oracles if the corresponding input is queried together with the correct K which can be checked by decapsulating the KEM ciphertext (Line 62 and 75). We also have to consider invalid KEM ciphertexts, i.e. ciphertexts where the decapsulation yields $\perp$. In these cases, the ROs check if the input K equals $\mathsf{s}^{\star}$ which represents the correct behavior in an honest decapsulation. Since the change can be perfectly simulated it holds that

$$\Pr\left[\mathsf{G}_2^{\mathcal{A}} \Rightarrow 1\right] = \Pr\left[\mathsf{G}_3^{\mathcal{A}} \Rightarrow 1\right] .$$

Game G_4. In the final game, we raise bad and abort if the random oracle is queried on the correct chellenge KEM key. Further, we sample the challenge key k at random without querying the random oracle H, independent of the challenge bit β. Also, we do not explicitly query random oracle G, but sample challenge and padding $(\mathsf{chl}, \mathsf{pad})$ at random. Note that adversary $\mathcal{A}$ never sees the random oracle outputs corresponding to the challenge due to the aborts which makes the simulation of the challenge oracle perfect. The two games G_3 and G_4 are identical until bad is set to true, hence

$$\left|\Pr\left[\mathsf{G}_3^{\mathcal{A}} \Rightarrow 1\right] - \Pr\left[\mathsf{G}_4^{\mathcal{A}} \Rightarrow 1\right]\right| \leq \Pr[\mathsf{bad}] .$$

Observe that

$$\Pr[\mathsf{G}_4^{\mathcal{A}} \Rightarrow 1] = \frac{1}{2} .$$

Final reduction. To bound event bad, we construct a reduction $\mathcal{B}$ to OW-KCA such that

$$\Pr[\mathsf{bad}] \leq \mathsf{Adv}^{(q_{\mathsf{G}}+q_{\mathsf{H}})\text{-OW-KCA}}_{\mathsf{KEM}}(\mathcal{B}).$$

Adversary $\mathcal{B}$ is formalized in Figure 16. The challenge oracle can be simulated by using inputs $\mathsf{com}^{\star}$ and $\mathsf{ct}^{\star}_1$ from $\mathcal{B}$ and $\mathsf{ct}_{\mathsf{rsp}}$ is chosen uniformly random which is a correct simulation because without having the output of G, pad is uniformly random and thus $\mathsf{ct}_{\mathsf{rsp}}$ is too. Note that the adversary never sees the output of G or H corresponding to the challenge because the game aborts in these cases and reduction $\mathcal{B}$ is winning the OW-KCA game. Finally, the abort condition is deployed via queries to the reduction's check oracle `Check`. □

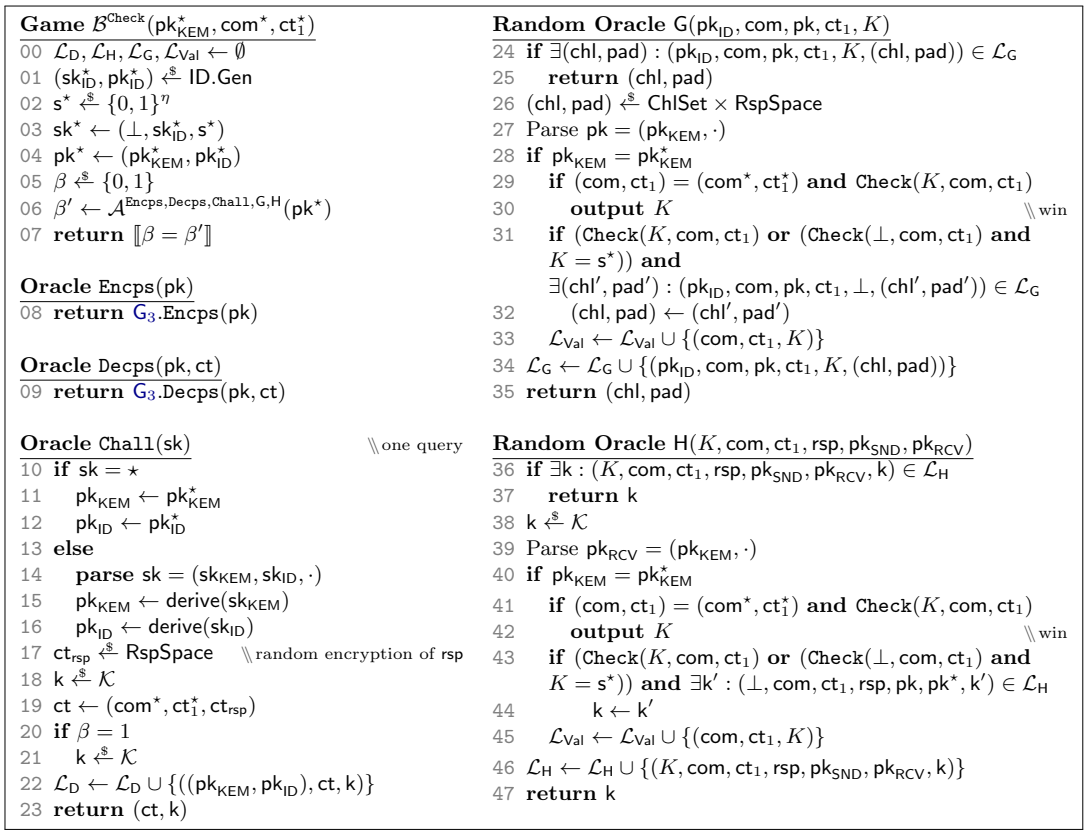

Fig. 16. Adversary $\mathcal{B}$ against OW-KCA of KEM having access to oracles `Check` and simulating $\mathsf{G}_3/\mathsf{G}_4$ for adversary $\mathcal{A}$.

D.2 Proof of Theorem 3.8

Theorem 3.8 (Insider Authenticity). *For any adversary $\mathcal{A}$ against* Ins-Aut *security of* SnakeM[KEM, ID, G, H] *making at most q_{G} random oracle queries to* G *and at most q_{H} random oracle queries to* H*, there exist an adversary $\mathcal{B}$ against* IMP-Enc*, an adversary $\mathcal{C}$ against* SS-Enc*, and an adversary $\mathcal{D}$ against* NM-Enc *such that*

$$\begin{aligned}\mathsf{Adv}^{(q_{\mathtt{Enc}},q_{\mathtt{Dec}})\text{-Ins-Aut}}_{\mathsf{SnakeM}[\mathsf{KEM},\mathsf{ID},\mathsf{G},\mathsf{H}]}(\mathcal{A}) &\le \mathsf{Adv}^{(q_{\mathtt{Enc}},q_{\mathsf{G}},q_{\mathsf{G}}+q_{\mathsf{H}})\text{-IMP-Enc}}_{\mathsf{KEM},\mathsf{ID}}(\mathcal{B}) + \mathsf{Adv}^{(q_{\mathtt{Enc}},q_{\mathsf{G}}+q_{\mathsf{H}})\text{-SS-Enc}}_{\mathsf{KEM},\mathsf{ID}}(\mathcal{C}) \\ &+ \mathsf{Adv}^{(q_{\mathtt{Enc}},q_{\mathsf{G}},q_{\mathsf{G}}+q_{\mathsf{H}})\text{-NM-Enc}}_{\mathsf{KEM},\mathsf{ID}}(\mathcal{D}) + \frac{q_{\mathtt{Enc}} \cdot q_{\mathsf{G}}}{|\mathsf{ChlSet}|} \\ &+ q_{\mathtt{Enc}} \cdot \delta_{\mathsf{ID}} + q_{\mathtt{Enc}}(q_{\mathsf{G}} + q_{\mathsf{H}}) \cdot \gamma_{\mathsf{ID}}\gamma_{\mathsf{KEM}},\end{aligned}$$

where ChlSet *is the challenge space of* ID.

Proof. We prove the theorem by a sequence of games depicted in Figure 17.

Game G_0. We start with the original Ins-Aut game for SnakeM.

$$\mathsf{Adv}^{(q_{\mathtt{Enc}},q_{\mathtt{Dec}})\text{-Ins-Aut}}_{\mathsf{SnakeM}[\mathsf{KEM},\mathsf{ID},\mathsf{G},\mathsf{H}]}(\mathcal{A}) = \left|\Pr[\mathsf{G}_0^{\mathcal{A}} \Rightarrow 1] - \frac{1}{2}\right|.$$

Game G_1. This is the same as the last game except that it aborts in the encapsulation oracle if RO G or H were queried before on the same commitment and ciphertext; com and ct_1. For at most q_{G} queries to G and q_{H} queries to H, by the spreadness of commitment and ciphertext it follows that

$$\left|\Pr\left[\mathsf{G}_0^{\mathcal{A}} \Rightarrow 1\right] - \Pr\left[\mathsf{G}_1^{\mathcal{A}} \Rightarrow 1\right]\right| \le q_{\mathtt{Enc}}(q_{\mathsf{G}} + q_{\mathsf{H}}) \cdot \gamma_{\mathsf{ID}}\gamma_{\mathsf{KEM}}.$$

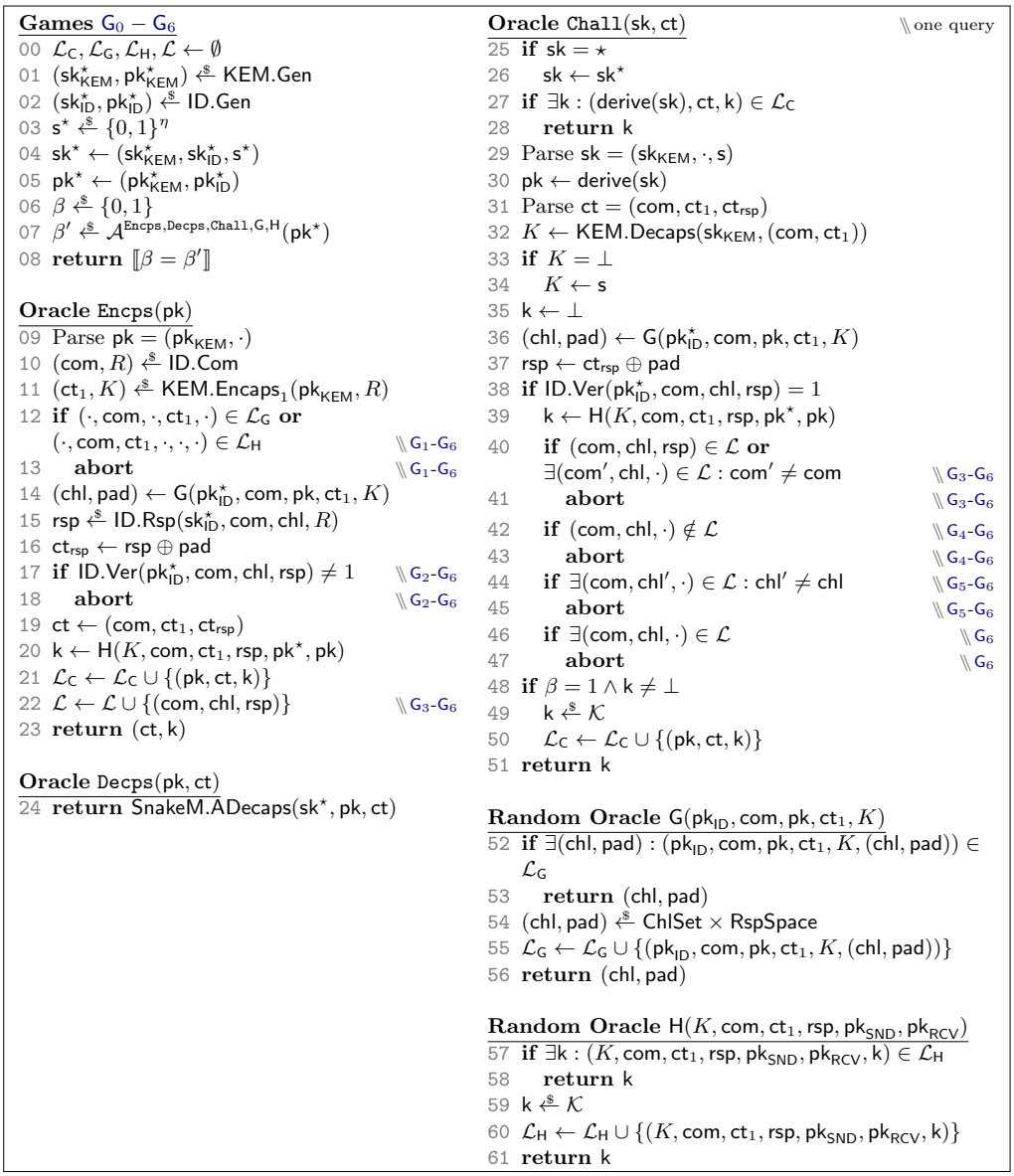

Fig. 17. Games $G_0 - G_6$ for the proof of Theorem 3.8.

Game G_2. This is the same as the last game except that it aborts if the ID transcript computed in the encapsulation oracle does not verify. The difference for one query is at most the correctness error of the scheme since the transcript is honestly computed. Hence it holds that

$$\left|\Pr\left[G_1^{\mathcal{A}} \Rightarrow 1\right] - \Pr\left[G_2^{\mathcal{A}} \Rightarrow 1\right]\right| \leq q_{\mathtt{Enc}} \cdot \delta_{\mathsf{ID}}.$$

Game G_3. This is identical to the last game except that we introduce list $\mathcal{L}$ which is filled in the encapsulation oracle with the transcript of ID, i.e. with tuples (com, rsp, chl). Further, the game aborts if the transcript computed in the challenge oracle verifies and already exists in list $\mathcal{L}$ or if there exists a transcript in $\mathcal{L}$ for which chl is the same but com is not (Line 41).

Claim 1:

$$\left|\Pr\left[G_2^{\mathcal{A}} \Rightarrow 1\right] - \Pr\left[G_3^{\mathcal{A}} \Rightarrow 1\right]\right| \leq \frac{q_{\mathtt{Enc}} \cdot q_G}{|ChlSet|}.$$

Proof (Claim). Assume the game aborts in Line 41. We first consider the case that there exists the same transcript (com, chl, rsp) in $\mathcal{L}$ already. This means that there must also exist an element

$(\mathsf{pk}', \mathsf{ct}', \mathsf{k}')$ in $\mathcal{C}$ such that $\mathsf{ct}' = (\mathsf{com}, \mathsf{ct}'_1, \mathsf{ct}'_{\mathsf{rsp}})$. It must hold $(\mathsf{pk}', \mathsf{ct}'_1, \mathsf{ct}'_{\mathsf{rsp}}) \neq (\mathsf{pk}, \mathsf{ct}_1, \mathsf{ct}_{\mathsf{rsp}})$, because otherwise the challenge oracle would return in Line 28.

Case: $\mathsf{ct}'_{\mathsf{rsp}} \neq \mathsf{ct}_{\mathsf{rsp}}$: This case implies $\mathsf{pad}' \neq \mathsf{pad}$ where pad' is the encryption padding from the encapsulation oracle because the response rsp is the same by assumption. This in turn means that the input to RO G must have been different. However, since chl is the same, we found a partial collision in G; in particular, a collision in the first output part which has size $|\mathsf{ChlSet}|$.

Case: $(\mathsf{pk}', \mathsf{ct}'_1) \neq (\mathsf{pk}, \mathsf{ct}_1)$: Since both, pk and ct_1, are input to RO G and the challenge chl is the same by assumption, we found a collision in the first output part of G again.

In the second abort condition, we have the same chl but a different com, we can apply the same argument as in the previous case directly since com is input to G which implies a collision again.

Overall, any case can be reduced to the probability of a collision between a chl component in $\mathcal{L}$ and the first part of a query to G. Further, we have $|\mathcal{L}| \leq q_{\mathtt{Enc}}$ and at most q_{G} queries to G which yields the claim. □

Game G_4. This is the same as the last game except that it aborts if for the commitment com and challenge chl computed in the challenge oracle there exists no element $(\mathsf{com}, \mathsf{chl}, \cdot)$ in $\mathcal{L}$ and the transcript verifies.

Claim 2: There exists and adversary $\mathcal{B}$ against IMP-Enc such that

$$\left|\Pr\left[\mathsf{G}_3^{\mathcal{A}} \Rightarrow 1\right] - \Pr\left[\mathsf{G}_4^{\mathcal{A}} \Rightarrow 1\right]\right| \leq \mathsf{Adv}^{(q_{\mathtt{Enc}}, q_{\mathsf{G}}, q_{\mathsf{G}}+q_{\mathsf{H}})\text{-IMP-Enc}}_{\mathsf{KEM},\mathsf{ID}}(\mathcal{B}).$$

Proof (Claim). We formally construct adversary $\mathcal{B}$ in Figure 18. The encapsulation oracle can be simulated by their own encapsulation oracle $\mathtt{Encps}_{\mathcal{B}}$. Receiving the response rsp, the reduction can now simulate an encrypted response by choosing the ciphertext uniformly random, computing the encryption padding by $\mathsf{pad} \leftarrow \mathsf{ct}_{\mathsf{rsp}} \oplus \mathsf{rsp}$ and store it together with the challenge to be able to patch the RO later. Due to the changes in G_1 this simulation is sound because G was not queried on the respective values. To answer RO queries to G and H, the outputs need to be consistent with $\mathtt{Encps}$ which means that the reduction has to recognize inputs corresponding to previous encapsulation queries. This is resolved by using the check oracle $\mathtt{Check}$ to recognize KEM shared keys corresponding to encapsulation queries such that the query can be answered using the stored values. If the input to G does not correspond to an $\mathtt{Encps}$ query, $\mathcal{B}$ uses their challenge oracle $\mathtt{Chal}_{\mathcal{B}}$ to sample a new challenge chl and samples a random padding pad on their own. Eventually, if the condition in Line 50 is fulfilled, adversary $\mathcal{B}$ can output the challenge transcript. The condition $(\mathsf{com}, \mathsf{chl}, \cdot) \notin \mathcal{L}$ implies that chl was output of a $\mathtt{Chal}_{\mathcal{B}}$ query (in G) and is thus a valid output for the IMP-Enc game.

□

Game G_5. This is the same as the last game except that it aborts if for the commitment com and challenge chl computed in the challenge oracle there already exists a tuple in $\mathcal{L}$ with the same com and a different chl and the complete challenge transcript verifies.

Claim 3: There exists and adversary $\mathcal{C}$ against SS-Enc such that

$$\left|\Pr\left[\mathsf{G}_4^{\mathcal{A}} \Rightarrow 1\right] - \Pr\left[\mathsf{G}_5^{\mathcal{A}} \Rightarrow 1\right]\right| \leq \mathsf{Adv}^{(q_{\mathtt{Enc}}, q_{\mathsf{G}}+q_{\mathsf{H}})\text{-SS-Enc}}_{\mathsf{KEM},\mathsf{ID}}(\mathcal{C}).$$

Proof (Claim). Adversary $\mathcal{C}$ is formally constructed in Figure 19. The simulation of encapsulation oracle and random oracles works similar to the previous reduction. The only difference is that there is no challenge oracle for adversary $\mathcal{C}$ and challenges are sampled directly from ChlSet. Outputting two transcripts in Line 52 yields a valid solution since the challenges are different and the transcripts in $\mathcal{L}$ are verifying due to the changes in Game G_2.

□

Adversary $\mathcal{B}^{\mathtt{Encps}_{\mathcal{B}},\mathtt{Chal}_{\mathcal{B}},\mathtt{Check}}(\mathsf{pk}^\star_{\mathsf{ID}})$
00 $\mathcal{L}_{\mathsf{C}}, \mathcal{L}_{\mathsf{G}}, \mathcal{L}_{\mathsf{H}}, \mathcal{L} \leftarrow \emptyset$
01 $(\mathsf{sk}^\star_{\mathsf{KEM}}, \mathsf{pk}^\star_{\mathsf{KEM}}) \xleftarrow{\$} \mathsf{KEM.Gen}$
02 $\mathsf{s}^\star \xleftarrow{\$} \{0,1\}^\eta$
03 $\mathsf{sk}^\star \leftarrow (\mathsf{sk}^\star_{\mathsf{KEM}}, \bot, \mathsf{s}^\star)$
04 $\mathsf{pk}^\star \leftarrow (\mathsf{pk}^\star_{\mathsf{KEM}}, \mathsf{pk}^\star_{\mathsf{ID}})$
05 $\beta \xleftarrow{\$} \{0,1\}$
06 $\beta' \xleftarrow{\$} \mathcal{A}^{\mathtt{Encps},\mathtt{Decps},\mathtt{Chall},\mathsf{G},\mathsf{H}}(\mathsf{pk}^\star)$
07 **return** $[\![\beta = \beta']\!]$

Oracle $\mathtt{Encps}(\mathsf{pk})$
08 Parse $\mathsf{pk} = (\mathsf{pk}_{\mathsf{KEM}}, \cdot)$
09 $((\mathsf{com}, \mathsf{ct}_1, \mathsf{rsp}), \mathsf{chl}) \xleftarrow{\$} \mathtt{Encps}_{\mathcal{B}}(\mathsf{pk}_{\mathsf{KEM}})$
10 $\mathsf{ct}_{\mathsf{rsp}} \xleftarrow{\$} \mathsf{RspSpace}$ ⟍ randomly choose ciphertext
11 $\mathsf{pad} \leftarrow \mathsf{ct}_{\mathsf{rsp}} \oplus \mathsf{rsp}$ ⟍ compute encryption padding
12 **if** $(\cdot, \mathsf{com}, \cdot, \mathsf{ct}_1, \cdot) \in \mathcal{L}_{\mathsf{G}}$ **or** $(\cdot, \mathsf{com}, \mathsf{ct}_1, \cdot, \cdot, \cdot) \in \mathcal{L}_{\mathsf{H}}$
13 **abort**
14 $\mathcal{L}_{\mathsf{G}} \leftarrow \mathcal{L}_{\mathsf{G}} \cup \{(\mathsf{pk}^\star_{\mathsf{ID}}, \mathsf{com}, \mathsf{pk}, \mathsf{ct}_1, \bot, (\mathsf{chl}, \bot))\}$ ⟍ store RO output
15 **if** $\mathsf{ID.Ver}(\mathsf{pk}^\star_{\mathsf{ID}}, \mathsf{com}, \mathsf{chl}, \mathsf{rsp}) \neq 1$
16 **abort**
17 $\mathsf{k} \xleftarrow{\$} \mathcal{K}$
18 $\mathcal{L}_{\mathsf{H}} \leftarrow \mathcal{L}_{\mathsf{H}} \cup \{(\bot, \mathsf{com}, \mathsf{ct}_1, \mathsf{rsp}, \mathsf{pk}^\star, \mathsf{pk}, \mathsf{k})\}$ ⟍ store RO output
19 $\mathsf{ct} \leftarrow (\mathsf{com}, \mathsf{ct}_1, \mathsf{ct}_{\mathsf{rsp}})$
20 $\mathcal{L}_{\mathsf{C}} \leftarrow \mathcal{L}_{\mathsf{C}} \cup \{(\mathsf{pk}, \mathsf{ct}, \mathsf{k})\}$
21 $\mathcal{L} \leftarrow \mathcal{L} \cup \{(\mathsf{com}, \mathsf{chl}, \mathsf{rsp})\}$
22 **return** $(\mathsf{ct}, \mathsf{k})$

Oracle $\mathtt{Decps}(\mathsf{pk}, \mathsf{ct})$
23 **return** $\mathsf{G}_3.\mathtt{Decps}(\mathsf{pk}, \mathsf{ct})$

Random Oracle $\mathsf{G}(\mathsf{pk}_{\mathsf{ID}}, \mathsf{com}, \mathsf{pk}, \mathsf{ct}_1, K)$
24 **if** $\exists(\mathsf{chl}, \mathsf{pad}) : (\mathsf{pk}_{\mathsf{ID}}, \mathsf{com}, \mathsf{pk}, \mathsf{ct}_1, K, (\mathsf{chl}, \mathsf{pad})) \in \mathcal{L}_{\mathsf{G}}$
25 **return** $(\mathsf{chl}, \mathsf{pad})$
26 $\mathsf{chl} \xleftarrow{\$} \mathtt{Chal}_{\mathcal{B}}(\mathsf{com})$ ⟍ fresh challenge
27 $\mathsf{pad} \xleftarrow{\$} \mathsf{RspSpace}$
28 Parse $\mathsf{pk} = (\mathsf{pk}_{\mathsf{KEM}}, \cdot)$
29 **if** $\exists(\mathsf{chl}', \mathsf{pad}') : (\mathsf{pk}_{\mathsf{ID}}, \mathsf{com}, \mathsf{pk}, \mathsf{ct}_1, \bot, (\mathsf{chl}', \mathsf{pad}')) \in \mathcal{L}_{\mathsf{G}}$ **and** $\mathtt{Check}(K, \mathsf{pk}_{\mathsf{KEM}}, \mathsf{com}, \mathsf{ct}_1) = 1$
30 $(\mathsf{chl}, \mathsf{pad}) \leftarrow (\mathsf{chl}', \mathsf{pad}')$ ⟍ program consistently with Encps
31 $\mathcal{L}_{\mathsf{G}} \leftarrow \mathcal{L}_{\mathsf{G}} \cup \{(\mathsf{pk}_{\mathsf{ID}}, \mathsf{com}, \mathsf{pk}, \mathsf{ct}_1, K, (\mathsf{chl}, \mathsf{pad}))\}$
32 **return** $(\mathsf{chl}, \mathsf{pad})$

Oracle $\mathtt{Chall}(\mathsf{sk}, \mathsf{ct})$ ⟍ one query
33 **if** $\mathsf{sk} = \star$
34 $\mathsf{sk} \leftarrow \mathsf{sk}^\star$
35 **if** $\exists \mathsf{k} : (\mathsf{derive}(\mathsf{sk}), \mathsf{ct}, \mathsf{k}) \in \mathcal{L}_{\mathsf{C}}$
36 **return** k
37 Parse $\mathsf{sk} = (\mathsf{sk}_{\mathsf{KEM}}, \cdot, \mathsf{s})$
38 $\mathsf{pk} \leftarrow \mathsf{derive}(\mathsf{sk})$
39 Parse $\mathsf{ct} = (\mathsf{com}, \mathsf{ct}_1, \mathsf{ct}_{\mathsf{rsp}})$
40 $K \leftarrow \mathsf{KEM.Decaps}(\mathsf{sk}_{\mathsf{KEM}}, (\mathsf{com}, \mathsf{ct}_1))$
41 **if** $K = \bot$
42 $K \leftarrow \mathsf{s}$
43 $\mathsf{k} \leftarrow \bot$
44 $(\mathsf{chl}, \mathsf{pad}) \leftarrow \mathsf{G}(\mathsf{pk}^\star_{\mathsf{ID}}, \mathsf{com}, \mathsf{pk}, \mathsf{ct}_1, K)$
45 $\mathsf{rsp} \leftarrow \mathsf{ct}_{\mathsf{rsp}} \oplus \mathsf{pad}$
46 **if** $\mathsf{ID.Ver}(\mathsf{pk}^\star_{\mathsf{ID}}, \mathsf{com}, \mathsf{chl}, \mathsf{rsp}) = 1$
47 $\mathsf{k} \leftarrow \mathsf{H}(K, \mathsf{com}, \mathsf{ct}_1, \mathsf{rsp}, \mathsf{pk}^\star, \mathsf{pk})$
48 **if** $(\mathsf{com}, \mathsf{chl}, \mathsf{rsp}) \in \mathcal{L}$ **or** $\exists(\mathsf{com}', \mathsf{chl}, \cdot) \in \mathcal{L} : \mathsf{com}' \neq \mathsf{com}$
49 **abort**
50 **if** $(\mathsf{com}, \mathsf{chl}, \cdot) \notin \mathcal{L}$
51 **output** $(\mathsf{com}, \mathsf{chl}, \mathsf{rsp})$ ⟍ win
52 **if** $\beta = 1 \wedge \mathsf{k} \neq \bot$
53 $\mathsf{k} \xleftarrow{\$} \mathcal{K}$
54 $\mathcal{L}_{\mathsf{C}} \leftarrow \mathcal{L}_{\mathsf{C}} \cup \{(\mathsf{pk}, \mathsf{ct}, \mathsf{k})\}$
55 **return** k

Random Oracle $\mathsf{H}(K, \mathsf{com}, \mathsf{ct}_1, \mathsf{rsp}, \mathsf{pk}_{\mathsf{SND}}, \mathsf{pk}_{\mathsf{RCV}})$
56 **if** $\exists \mathsf{k} : (K, \mathsf{com}, \mathsf{ct}_1, \mathsf{rsp}, \mathsf{pk}_{\mathsf{SND}}, \mathsf{pk}_{\mathsf{RCV}}, \mathsf{k}) \in \mathcal{L}_{\mathsf{H}}$
57 **return** k
58 $\mathsf{k} \xleftarrow{\$} \mathcal{K}$
59 Parse $\mathsf{pk} = (\mathsf{pk}_{\mathsf{KEM}}, \cdot)$
60 **if** $\exists \mathsf{k}' : (\bot, \mathsf{com}, \mathsf{ct}_1, \mathsf{rsp}, \mathsf{pk}_{\mathsf{SND}}, \mathsf{pk}_{\mathsf{RCV}}, \mathsf{k}') \in \mathcal{L}_{\mathsf{H}}$ **and** $\mathtt{Check}(K, \mathsf{pk}_{\mathsf{KEM}}, \mathsf{com}, \mathsf{ct}_1) = 1$
61 $\mathsf{k} \leftarrow \mathsf{k}'$ ⟍ program consistently with Encps
62 $\mathcal{L}_{\mathsf{H}} \leftarrow \mathcal{L}_{\mathsf{H}} \cup \{(K, \mathsf{com}, \mathsf{ct}_1, \mathsf{rsp}, \mathsf{pk}_{\mathsf{SND}}, \mathsf{pk}_{\mathsf{RCV}}, \mathsf{k})\}$
63 **return** k

Fig. 18. Adversary $\mathcal{B}$ against IMP-Enc having access to oracles $\mathtt{Encps}_{\mathcal{B}}, \mathtt{Chal}_{\mathcal{B}}, \mathtt{Check}$ and simulating $\mathsf{G}_3/\mathsf{G}_4$ for $\mathcal{A}$.

Game G_6. This is the same as the last game except that it aborts if for the commitment com and challenge chl computed in the challenge oracle there already exists a tuple in $\mathcal{L}$ with the same com and chl and the complete challenge transcript verifies.

Claim 4: There exists and adversary $\mathcal{D}$ against NM-Enc such that

$$\left|\Pr\left[\mathsf{G}_5^{\mathcal{A}} \Rightarrow 1\right] - \Pr\left[\mathsf{G}_6^{\mathcal{A}} \Rightarrow 1\right]\right| \leq \mathsf{Adv}^{(q_{\mathtt{Enc}}, q_{\mathsf{G}}, q_{\mathsf{G}}+q_{\mathsf{H}})\text{-NM-Enc}}_{\mathsf{KEM},\mathsf{ID}}(\mathcal{D}).$$

Proof (Claim). Adversary $\mathcal{D}$ is formally constructed in Figure 20. The simulation of the game is similar to the reduction for IMP-Enc and the winning condition for adversary $\mathcal{D}$ is fulfilled by construction.

□

Claim 5:

$$\Pr[\mathsf{G}_6^{\mathcal{A}} \Rightarrow 1] = \frac{1}{2}.$$

Proof (Claim). Game G_6 aborts in the challenge oracle in any case if the ID transcript verifies: the tuple (com, chl) can either occur in $\mathcal{L}$, not appear, or appear partially, i.e. either com or chl, and all cases occur in the abort conditions. Hence, the game only continues for invalid

Adversary $\mathcal{C}^{\mathtt{Encps}_{\mathcal{C}},\mathtt{Check}}(\mathsf{pk}^\star_{\mathsf{ID}})$
00 $\mathcal{L}_{\mathsf{C}}, \mathcal{L}_{\mathsf{G}}, \mathcal{L}_{\mathsf{H}}, \mathcal{L} \leftarrow \emptyset$
01 $(\mathsf{sk}^\star_{\mathsf{KEM}}, \mathsf{pk}^\star_{\mathsf{KEM}}) \xleftarrow{\$} \mathsf{KEM.Gen}$
02 $\mathsf{s}^\star \xleftarrow{\$} \{0,1\}^\eta$
03 $\mathsf{sk}^\star \leftarrow (\mathsf{sk}^\star_{\mathsf{KEM}}, \bot, \mathsf{s}^\star)$
04 $\mathsf{pk}^\star \leftarrow (\mathsf{pk}^\star_{\mathsf{KEM}}, \mathsf{pk}^\star_{\mathsf{ID}})$
05 $\beta \xleftarrow{\$} \{0,1\}$
06 $\beta' \xleftarrow{\$} \mathcal{A}^{\mathtt{Encps},\mathtt{Decps},\mathtt{Chall},\mathsf{G},\mathsf{H}}(\mathsf{pk}^\star)$
07 **return** $[\![\beta = \beta']\!]$

Oracle $\mathtt{Encps}(\mathsf{pk})$
08 Parse $\mathsf{pk} = (\mathsf{pk}_{\mathsf{KEM}}, \cdot)$
09 $((\mathsf{com}, \mathsf{ct}_1, \mathsf{rsp}), \mathsf{chl}) \xleftarrow{\$} \mathtt{Encps}_{\mathcal{C}}(\mathsf{pk}_{\mathsf{KEM}})$
10 $\mathsf{ct}_{\mathsf{rsp}} \xleftarrow{\$} \mathsf{RspSpace}$ ⟍ randomly choose ciphertext
11 $\mathsf{pad} \leftarrow \mathsf{ct}_{\mathsf{rsp}} \oplus \mathsf{rsp}$ ⟍ compute encryption padding
12 **if** $(\cdot, \mathsf{com}, \cdot, \mathsf{ct}_1, \cdot) \in \mathcal{L}_{\mathsf{G}}$ **or** $(\cdot, \mathsf{com}, \mathsf{ct}_1, \cdot, \cdot, \cdot) \in \mathcal{L}_{\mathsf{H}}$
13 **abort**
14 $\mathcal{L}_{\mathsf{G}} \leftarrow \mathcal{L}_{\mathsf{G}} \cup \{(\mathsf{pk}^\star_{\mathsf{ID}}, \mathsf{com}, \mathsf{pk}, \mathsf{ct}_1, \bot, (\mathsf{chl}, \bot))\}$ ⟍ store RO output
15 **if** $\mathsf{ID.Ver}(\mathsf{pk}^\star_{\mathsf{ID}}, \mathsf{com}, \mathsf{chl}, \mathsf{rsp}) \neq 1$
16 **abort**
17 $\mathsf{k} \xleftarrow{\$} \mathcal{K}$
18 $\mathcal{L}_{\mathsf{H}} \leftarrow \mathcal{L}_{\mathsf{H}} \cup \{(\bot, \mathsf{com}, \mathsf{ct}_1, \mathsf{rsp}, \mathsf{pk}^\star, \mathsf{pk}, \mathsf{k})\}$ ⟍ store RO output
19 $\mathsf{ct} \leftarrow (\mathsf{com}, \mathsf{ct}_1, \mathsf{ct}_{\mathsf{rsp}})$
20 $\mathcal{L}_{\mathsf{C}} \leftarrow \mathcal{L}_{\mathsf{C}} \cup \{(\mathsf{pk}, \mathsf{ct}, \mathsf{k})\}$
21 $\mathcal{L} \leftarrow \mathcal{L} \cup \{(\mathsf{com}, \mathsf{chl}, \mathsf{rsp})\}$
22 **return** $(\mathsf{ct}, \mathsf{k})$

Oracle $\mathtt{Decps}(\mathsf{pk}, \mathsf{ct})$
23 **return** $\mathsf{G}_3.\mathtt{Decps}(\mathsf{pk}, \mathsf{ct})$

Random Oracle $\mathsf{G}(\mathsf{pk}_{\mathsf{ID}}, \mathsf{com}, \mathsf{pk}, \mathsf{ct}_1, K)$
24 **if** $\exists(\mathsf{chl}, \mathsf{pad}) : (\mathsf{pk}_{\mathsf{ID}}, \mathsf{com}, \mathsf{pk}, \mathsf{ct}_1, K, (\mathsf{chl}, \mathsf{pad})) \in \mathcal{L}_{\mathsf{G}}$
25 **return** $(\mathsf{chl}, \mathsf{pad})$
26 $(\mathsf{chl}, \mathsf{pad}) \xleftarrow{\$} \mathsf{ChlSet} \times \mathsf{RspSpace}$
27 Parse $\mathsf{pk} = (\mathsf{pk}_{\mathsf{KEM}}, \cdot)$
28 **if** $\exists(\mathsf{chl}', \mathsf{pad}') : (\mathsf{pk}_{\mathsf{ID}}, \mathsf{com}, \mathsf{pk}, \mathsf{ct}_1, \bot, (\mathsf{chl}', \mathsf{pad}')) \in \mathcal{L}_{\mathsf{G}}$ **and** $\mathtt{Check}(K, \mathsf{pk}_{\mathsf{KEM}}, \mathsf{com}, \mathsf{ct}_1) = 1$
29 $(\mathsf{chl}, \mathsf{pad}) \leftarrow (\mathsf{chl}', \mathsf{pad}')$ ⟍ program consistently with Encps
30 $\mathcal{L}_{\mathsf{G}} \leftarrow \mathcal{L}_{\mathsf{G}} \cup \{(\mathsf{pk}_{\mathsf{ID}}, \mathsf{com}, \mathsf{pk}, \mathsf{ct}_1, K, (\mathsf{chl}, \mathsf{pad}))\}$
31 **return** $(\mathsf{chl}, \mathsf{pad})$

Oracle $\mathtt{Chall}(\mathsf{sk}, \mathsf{ct})$ ⟍ one query
32 **if** $\mathsf{sk} = \star$
33 $\mathsf{sk} \leftarrow \mathsf{sk}^\star$
34 **if** $\exists \mathsf{k} : (\mathsf{derive}(\mathsf{sk}), \mathsf{ct}, \mathsf{k}) \in \mathcal{L}_{\mathsf{C}}$
35 **return** k
36 Parse $\mathsf{sk} = (\mathsf{sk}_{\mathsf{KEM}}, \cdot, \mathsf{s})$
37 $\mathsf{pk} \leftarrow \mathsf{derive}(\mathsf{sk})$
38 Parse $\mathsf{ct} = (\mathsf{com}, \mathsf{ct}_1, \mathsf{ct}_{\mathsf{rsp}})$
39 $K \leftarrow \mathsf{KEM.Decaps}(\mathsf{sk}_{\mathsf{KEM}}, (\mathsf{com}, \mathsf{ct}_1))$
40 **if** $K = \bot$
41 $K \leftarrow \mathsf{s}$
42 $\mathsf{k} \leftarrow \bot$
43 $(\mathsf{chl}, \mathsf{pad}) \leftarrow \mathsf{G}(\mathsf{pk}^\star_{\mathsf{ID}}, \mathsf{com}, \mathsf{pk}, \mathsf{ct}_1, K)$
44 $\mathsf{rsp} \leftarrow \mathsf{ct}_{\mathsf{rsp}} \oplus \mathsf{pad}$
45 **if** $\mathsf{ID.Ver}(\mathsf{pk}^\star_{\mathsf{ID}}, \mathsf{com}, \mathsf{chl}, \mathsf{rsp}) = 1$
46 $\mathsf{k} \leftarrow \mathsf{H}(K, \mathsf{com}, \mathsf{ct}_1, \mathsf{rsp}, \mathsf{pk}^\star, \mathsf{pk})$
47 **if** $(\mathsf{com}, \mathsf{chl}, \mathsf{rsp}) \in \mathcal{L}$ **or** $\exists(\mathsf{com}', \mathsf{chl}, \cdot) \in \mathcal{L} : \mathsf{com}' \neq \mathsf{com}$
48 **abort**
49 **if** $(\mathsf{com}, \mathsf{chl}, \cdot) \notin \mathcal{L}$
50 **abort**
51 **if** $\exists(\mathsf{com}, \mathsf{chl}', \mathsf{rsp}') \in \mathcal{L} : \mathsf{chl}' \neq \mathsf{chl}$
52 **output** $(\mathsf{com}, \mathsf{chl}, \mathsf{rsp}, \mathsf{chl}', \mathsf{rsp}')$ ⟍ win
53 **if** $\beta = 1 \wedge \mathsf{k} \neq \bot$
54 $\mathsf{k} \xleftarrow{\$} \mathcal{K}$
55 $\mathcal{L}_{\mathsf{C}} \leftarrow \mathcal{L}_{\mathsf{C}} \cup \{(\mathsf{pk}, \mathsf{ct}, \mathsf{k})\}$
56 **return** k

Random Oracle $\mathsf{H}(K, \mathsf{com}, \mathsf{ct}_1, \mathsf{rsp}, \mathsf{pk}_{\mathsf{SND}}, \mathsf{pk}_{\mathsf{RCV}})$
57 **if** $\exists \mathsf{k} : (K, \mathsf{com}, \mathsf{ct}_1, \mathsf{rsp}, \mathsf{pk}_{\mathsf{SND}}, \mathsf{pk}_{\mathsf{RCV}}, \mathsf{k}) \in \mathcal{L}_{\mathsf{H}}$
58 **return** k
59 $\mathsf{k} \xleftarrow{\$} \mathcal{K}$
60 Parse $\mathsf{pk} = (\mathsf{pk}_{\mathsf{KEM}}, \cdot)$
61 **if** $\exists \mathsf{k}' : (\bot, \mathsf{com}, \mathsf{ct}_1, \mathsf{rsp}, \mathsf{pk}_{\mathsf{SND}}, \mathsf{pk}_{\mathsf{RCV}}, \mathsf{k}') \in \mathcal{L}_{\mathsf{H}}$ **and** $\mathtt{Check}(K, \mathsf{pk}_{\mathsf{KEM}}, \mathsf{com}, \mathsf{ct}_1) = 1$
62 $\mathsf{k} \leftarrow \mathsf{k}'$ ⟍ program consistently with Encps
63 $\mathcal{L}_{\mathsf{H}} \leftarrow \mathcal{L}_{\mathsf{H}} \cup \{(K, \mathsf{com}, \mathsf{ct}_1, \mathsf{rsp}, \mathsf{pk}_{\mathsf{SND}}, \mathsf{pk}_{\mathsf{RCV}}, \mathsf{k})\}$
64 **return** k

Fig. 19. Adversary $\mathcal{C}$ against SS-Enc having access to oracles $\mathtt{Encps}_{\mathcal{C}}, \mathtt{Check}$ and simulating $\mathsf{G}_4/\mathsf{G}_5$ for $\mathcal{A}$.

transcripts and thus always outputs $\mathsf{k} = \bot$ in the challenge oracle which is independent of challenge bit β. □

□

D.3 Proof of Theorem 3.9

Theorem 3.9 (Honest Receiver Deniability). *There exists a* PPT *simulator* Sim *such that for any* HR-Den *adversary* $\mathcal{A}$ *against* $\mathsf{SnakeM}[\mathsf{KEM}, \mathsf{ID}, \mathsf{G}, \mathsf{H}]$ *making at most* q_{G} *random oracle queries to* G *and at most* q_{H} *random oracle queries to* H*, there exists an adversary* $\mathcal{B}$ *against* OW *security such that*

$$\mathsf{Adv}^{\mathsf{HR\text{-}Den}}_{\mathsf{SnakeM}[\mathsf{KEM},\mathsf{ID},\mathsf{G},\mathsf{H}],\mathsf{Sim}}(\mathcal{A}) \leq (q_{\mathsf{G}} + q_{\mathsf{H}}) \cdot \mathsf{Adv}^{\mathsf{OW}}_{\mathsf{KEM}}(\mathcal{B}).$$

Proof. We prove the theorem by a sequence of games depicted in Figure 21.

Game G_0. We start with the original HR-Den game for SnakeM and a simulator Sim as defined in Figure 21.

$$\mathsf{Adv}^{\mathsf{HR\text{-}Den}}_{\mathsf{SnakeM}[\mathsf{KEM},\mathsf{ID},\mathsf{G},\mathsf{H}],\mathsf{Sim}}(\mathcal{A}) = \left|\Pr[\mathsf{G}_0^{\mathcal{A}} \Rightarrow 1] - \frac{1}{2}\right|.$$

Adversary $\mathcal{D}^{\mathtt{Encps}_{\mathcal{D}},\mathtt{Chal}_{\mathcal{D}},\mathtt{Check}}(\mathsf{pk}^\star_{\mathsf{ID}})$
00 $\mathcal{L}_\mathsf{C}, \mathcal{L}_\mathsf{G}, \mathcal{L}_\mathsf{H}, \mathcal{L} \leftarrow \emptyset$
01 $(\mathsf{sk}^\star_{\mathsf{KEM}}, \mathsf{pk}^\star_{\mathsf{KEM}}) \xleftarrow{\$} \mathsf{KEM.Gen}$
02 $\mathsf{s}^\star \xleftarrow{\$} \{0,1\}^\eta$
03 $\mathsf{sk}^\star \leftarrow (\mathsf{sk}^\star_{\mathsf{KEM}}, \bot, \mathsf{s}^\star)$
04 $\mathsf{pk}^\star \leftarrow (\mathsf{pk}^\star_{\mathsf{KEM}}, \mathsf{pk}^\star_{\mathsf{ID}})$
05 $\beta \xleftarrow{\$} \{0,1\}$
06 $\beta' \xleftarrow{\$} \mathcal{A}^{\mathtt{Encps},\mathtt{Decps},\mathtt{Chall},\mathsf{G},\mathsf{H}}(\mathsf{pk}^\star)$
07 **return** $[\![\beta = \beta']\!]$

Oracle $\mathtt{Encps}(\mathsf{pk})$
08 Parse $\mathsf{pk} = (\mathsf{pk}_{\mathsf{KEM}}, \cdot)$
09 $((\mathsf{com}, \mathsf{ct}_1, \mathsf{rsp}), \mathsf{chl}) \xleftarrow{\$} \mathtt{Encps}_{\mathcal{D}}(\mathsf{pk}_{\mathsf{KEM}})$
10 $\mathsf{ct}_{\mathsf{rsp}} \xleftarrow{\$} \mathsf{RspSpace}$ \\randomly choose ciphertext
11 $\mathsf{pad} \leftarrow \mathsf{ct}_{\mathsf{rsp}} \oplus \mathsf{rsp}$ \\compute encryption padding
12 **if** $(\cdot, \mathsf{com}, \cdot, \mathsf{ct}_1, \cdot) \in \mathcal{L}_\mathsf{G}$ **or** $(\cdot, \mathsf{com}, \mathsf{ct}_1, \cdot, \cdot, \cdot) \in \mathcal{L}_\mathsf{H}$
13 **abort**
14 $\mathcal{L}_\mathsf{G} \leftarrow \mathcal{L}_\mathsf{G} \cup \{(\mathsf{pk}^\star_{\mathsf{ID}}, \mathsf{com}, \mathsf{pk}, \mathsf{ct}_1, \bot, (\mathsf{chl}, \bot))\}$ \\store RO output
15 **if** $\mathsf{ID.Ver}(\mathsf{pk}^\star_{\mathsf{ID}}, \mathsf{com}, \mathsf{chl}, \mathsf{rsp}) \neq 1$
16 **abort**
17 $\mathsf{k} \xleftarrow{\$} \mathcal{K}$
18 $\mathcal{L}_\mathsf{H} \leftarrow \mathcal{L}_\mathsf{H} \cup \{(\bot, \mathsf{com}, \mathsf{ct}_1, \mathsf{rsp}, \mathsf{pk}^\star, \mathsf{pk}, \mathsf{k})\}$ \\store RO output
19 $\mathsf{ct} \leftarrow (\mathsf{com}, \mathsf{ct}_1, \mathsf{ct}_{\mathsf{rsp}})$
20 $\mathcal{L}_\mathsf{C} \leftarrow \mathcal{L}_\mathsf{C} \cup \{(\mathsf{pk}, \mathsf{ct}, \mathsf{k})\}$
21 $\mathcal{L} \leftarrow \mathcal{L} \cup \{(\mathsf{com}, \mathsf{chl}, \mathsf{rsp})\}$
22 **return** $(\mathsf{ct}, \mathsf{k})$

Oracle $\mathtt{Decps}(\mathsf{pk}, \mathsf{ct})$
23 **return** $\mathsf{G}_3.\mathtt{Decps}(\mathsf{pk}, \mathsf{ct})$

Random Oracle $\mathsf{G}(\mathsf{pk}_{\mathsf{ID}}, \mathsf{com}, \mathsf{pk}, \mathsf{ct}_1, K)$
24 **if** $\exists(\mathsf{chl}, \mathsf{pad}) : (\mathsf{pk}_{\mathsf{ID}}, \mathsf{com}, \mathsf{pk}, \mathsf{ct}_1, K, (\mathsf{chl}, \mathsf{pad})) \in \mathcal{L}_\mathsf{G}$
25 **return** $(\mathsf{chl}, \mathsf{pad})$
26 $\mathsf{chl} \xleftarrow{\$} \mathtt{Chal}_{\mathcal{D}}(\mathsf{com})$ \\fresh challenge
27 $\mathsf{pad} \xleftarrow{\$} \mathsf{RspSpace}$
28 Parse $\mathsf{pk} = (\mathsf{pk}_{\mathsf{KEM}}, \cdot)$
29 **if** $\exists(\mathsf{chl}', \mathsf{pad}') : (\mathsf{pk}_{\mathsf{ID}}, \mathsf{com}, \mathsf{pk}, \mathsf{ct}_1, \bot, (\mathsf{chl}', \mathsf{pad}')) \in \mathcal{L}_\mathsf{G}$ **and** $\mathtt{Check}(K, \mathsf{pk}_{\mathsf{KEM}}, \mathsf{com}, \mathsf{ct}_1) = 1$
30 $(\mathsf{chl}, \mathsf{pad}) \leftarrow (\mathsf{chl}', \mathsf{pad}')$ \\program consistently with Encps
31 $\mathcal{L}_\mathsf{G} \leftarrow \mathcal{L}_\mathsf{G} \cup \{(\mathsf{pk}_{\mathsf{ID}}, \mathsf{com}, \mathsf{pk}, \mathsf{ct}_1, K, (\mathsf{chl}, \mathsf{pad}))\}$
32 **return** $(\mathsf{chl}, \mathsf{pad})$

Oracle $\mathtt{Chall}(\mathsf{sk}, \mathsf{ct})$ \\one query
33 **if** $\mathsf{sk} = \star$
34 $\mathsf{sk} \leftarrow \mathsf{sk}^\star$
35 **if** $\exists \mathsf{k} : (\mathsf{derive}(\mathsf{sk}), \mathsf{ct}, \mathsf{k}) \in \mathcal{L}_\mathsf{C}$
36 **return** k
37 Parse $\mathsf{sk} = (\mathsf{sk}_{\mathsf{KEM}}, \cdot, \mathsf{s})$
38 $\mathsf{pk} \leftarrow \mathsf{derive}(\mathsf{sk})$
39 Parse $\mathsf{ct} = (\mathsf{com}, \mathsf{ct}_1, \mathsf{ct}_{\mathsf{rsp}})$
40 $K \leftarrow \mathsf{KEM.Decaps}(\mathsf{sk}_{\mathsf{KEM}}, (\mathsf{com}, \mathsf{ct}_1))$
41 **if** $K = \bot$
42 $K \leftarrow \mathsf{s}$
43 $\mathsf{k} \leftarrow \bot$
44 $(\mathsf{chl}, \mathsf{pad}) \leftarrow \mathsf{G}(\mathsf{pk}^\star_{\mathsf{ID}}, \mathsf{com}, \mathsf{pk}, \mathsf{ct}_1, K)$
45 $\mathsf{rsp} \leftarrow \mathsf{ct}_{\mathsf{rsp}} \oplus \mathsf{pad}$
46 **if** $\mathsf{ID.Ver}(\mathsf{pk}^\star_{\mathsf{ID}}, \mathsf{com}, \mathsf{chl}, \mathsf{rsp}) = 1$
47 $\mathsf{k} \leftarrow \mathsf{H}(K, \mathsf{com}, \mathsf{ct}_1, \mathsf{rsp}, \mathsf{pk}^\star, \mathsf{pk})$
48 **if** $(\mathsf{com}, \mathsf{chl}, \mathsf{rsp}) \in \mathcal{L}$ **or** $\exists(\mathsf{com}', \mathsf{chl}, \cdot) \in \mathcal{L} : \mathsf{com}' \neq \mathsf{com}$
49 **abort**
50 **if** $(\mathsf{com}, \mathsf{chl}, \cdot) \notin \mathcal{L}$
51 **abort**
52 **if** $\exists(\mathsf{com}, \mathsf{chl}', \cdot) \in \mathcal{L} : \mathsf{chl}' \neq \mathsf{chl}$
53 **abort**
54 **if** $\exists(\mathsf{com}, \mathsf{chl}, \mathsf{rsp}) \in \mathcal{L}$
55 **output** $(\mathsf{com}, \mathsf{chl}, \mathsf{rsp})$ \\win
56 **if** $\beta = 1 \wedge \mathsf{k} \neq \bot$
57 $\mathsf{k} \xleftarrow{\$} \mathcal{K}$
58 $\mathcal{L}_\mathsf{C} \leftarrow \mathcal{L}_\mathsf{C} \cup \{(\mathsf{pk}, \mathsf{ct}, \mathsf{k})\}$
59 **return** k

Random Oracle $\mathsf{H}(K, \mathsf{com}, \mathsf{ct}_1, \mathsf{rsp}, \mathsf{pk}_{\mathsf{SND}}, \mathsf{pk}_{\mathsf{RCV}})$
60 **if** $\exists \mathsf{k} : (K, \mathsf{com}, \mathsf{ct}_1, \mathsf{rsp}, \mathsf{pk}_{\mathsf{SND}}, \mathsf{pk}_{\mathsf{RCV}}, \mathsf{k}) \in \mathcal{L}_\mathsf{H}$
61 **return** k
62 $\mathsf{k} \xleftarrow{\$} \mathcal{K}$
63 Parse $\mathsf{pk} = (\mathsf{pk}_{\mathsf{KEM}}, \cdot)$
64 **if** $\exists \mathsf{k}' : (\bot, \mathsf{com}, \mathsf{ct}_1, \mathsf{rsp}, \mathsf{pk}_{\mathsf{SND}}, \mathsf{pk}_{\mathsf{RCV}}, \mathsf{k}') \in \mathcal{L}_\mathsf{H}$ **and** $\mathtt{Check}(K, \mathsf{pk}_{\mathsf{KEM}}, \mathsf{com}, \mathsf{ct}_1) = 1$
65 $\mathsf{k} \leftarrow \mathsf{k}'$ \\program consistently with Encps
66 $\mathcal{L}_\mathsf{H} \leftarrow \mathcal{L}_\mathsf{H} \cup \{(K, \mathsf{com}, \mathsf{ct}_1, \mathsf{rsp}, \mathsf{pk}_{\mathsf{SND}}, \mathsf{pk}_{\mathsf{RCV}}, \mathsf{k})\}$
67 **return** k

Fig. 20. Adversary $\mathcal{D}$ against NM-Enc having access to oracles $\mathtt{Encps}_{\mathcal{D}}, \mathtt{Chal}_{\mathcal{D}}, \mathtt{Check}$ and simulating $\mathsf{G}_5/\mathsf{G}_6$ for $\mathcal{A}$.

Game G_1. This game is the same as the previous one except that it aborts if the challenge KEM ciphertext and key, i.e. $\mathsf{com}^\star$, $\mathsf{ct}^\star_1$, and $K^\star$, are part of the query to random oracle G or H. Note that if the game does not abort and hence there does not exist a query with the corresponding components, this is equivalent to replacing the RO outputs for the challenges by uniformly random values (see Line 13 and 18).

Claim 6: There exists an adversary $\mathcal{B}$ against OW such that

$$\left|\Pr\left[\mathsf{G}_0^{\mathcal{A}} \Rightarrow 1\right] - \Pr\left[\mathsf{G}_1^{\mathcal{A}} \Rightarrow 1\right]\right| \leq (q_\mathsf{G} + q_\mathsf{H}) \cdot \mathsf{Adv}^{\mathsf{OW}}_{\mathsf{KEM}}(\mathcal{B}).$$

Proof (Claim). We define a sequence of intermediate games such that $\mathsf{G}_{0.i}$, $i \in [0, q_\mathsf{G} + q_\mathsf{H}]$, is the same as G_0 except that the game aborts if one of the random oracles is queried on the challenge ciphertext $(\mathsf{com}^\star, \mathsf{ct}^\star_1)$ together with the challenge KEM key $K^\star$ not later than in the i-th query to any of the random oracles. Note that we count the overall queries to both random oracles, G and H. Further it holds $\mathsf{G}_{0.0} = \mathsf{G}_0$ and $\mathsf{G}_{0.q_\mathsf{G}+q_\mathsf{H}} = \mathsf{G}_1$.

In Figure 22, we define adversary $\mathcal{B}_i$ against OW such that

$$|\Pr[\mathsf{G}^{\mathcal{A}}_{0.i-1} \Rightarrow 1] - \Pr[\mathsf{G}^{\mathcal{A}}_{0.i} \Rightarrow 1]| \leq \mathsf{Adv}^{\mathsf{OW}}_{\mathsf{KEM}}(\mathcal{B}_i).$$

Iterating over $i \in [1, q_\mathsf{G} + q_\mathsf{H}]$ yields the claim. □

Games $\mathsf{G}_0 - \mathsf{G}_2$

00 $(\mathsf{sk}_{\mathsf{KEM},1}, \mathsf{pk}_{\mathsf{KEM},1}) \xleftarrow{\$} \mathsf{KEM.Gen}$
01 $(\mathsf{sk}_{\mathsf{ID},1}, \mathsf{pk}_{\mathsf{ID},1}) \xleftarrow{\$} \mathsf{ID.Gen}$
02 $\mathsf{s}_1 \xleftarrow{\$} \{0,1\}^\eta$
03 $\mathsf{sk}_1 \leftarrow (\mathsf{sk}_{\mathsf{KEM},1}, \mathsf{sk}_{\mathsf{ID},1}, \mathsf{s}_1)$
04 $\mathsf{pk}_1 \leftarrow (\mathsf{pk}_{\mathsf{KEM},1}, \mathsf{pk}_{\mathsf{ID}}1))$
05 $(\mathsf{sk}_{\mathsf{KEM},2}, \mathsf{pk}_{\mathsf{KEM},2}) \xleftarrow{\$} \mathsf{KEM.Gen}$
06 $(\mathsf{sk}_{\mathsf{ID},2}, \mathsf{pk}_{\mathsf{ID},2}) \xleftarrow{\$} \mathsf{ID.Gen}$
07 $\mathsf{s}_2 \xleftarrow{\$} \{0,1\}^\eta$
08 $\mathsf{sk}_2 \leftarrow (\mathsf{sk}_{\mathsf{KEM},2}, \mathsf{sk}_{\mathsf{ID},2}, \mathsf{s}_2)$
09 $\mathsf{pk}_2 \leftarrow (\mathsf{pk}_{\mathsf{KEM},2}, \mathsf{pk}_{\mathsf{ID},2})$
10 $(\mathsf{com}, R) \xleftarrow{\$} \mathsf{ID.Com}$
11 $(\mathsf{ct}_1, K) \xleftarrow{\$} \mathsf{KEM.Encaps}_1(\mathsf{pk}_{\mathsf{KEM},2}, R)$
12 $(\mathsf{chl}, \mathsf{pad}) \leftarrow \mathsf{G}(\mathsf{pk}_{\mathsf{ID},1}, \mathsf{com}, \mathsf{pk}_2, \mathsf{ct}_1, K)$
13 $(\mathsf{chl}, \mathsf{pad}) \xleftarrow{\$} \mathsf{ChlSet} \times \mathsf{RspSpace}$ $\quad \setminus\setminus \mathsf{G}_1\text{-}\mathsf{G}_2$
14 $\mathsf{rsp} \xleftarrow{\$} \mathsf{ID.Rsp}(\mathsf{sk}_{\mathsf{ID},1}, \mathsf{com}, \mathsf{chl}, R)$
15 $\mathsf{ct}_{\mathsf{rsp}} \leftarrow \mathsf{rsp} \oplus \mathsf{pad}$
16 $\mathsf{ct}_{\mathsf{rsp}} \xleftarrow{\$} \mathsf{RspSpace}$ $\quad \setminus\setminus \mathsf{G}_2$
17 $\mathsf{k}_0 \leftarrow \mathsf{H}(K, \mathsf{com}, \mathsf{ct}_1, \mathsf{rsp}, \mathsf{pk}_1, \mathsf{pk}_2)$
18 $\mathsf{k}_0 \xleftarrow{\$} \mathcal{K}$ $\quad \setminus\setminus \mathsf{G}_1\text{-}\mathsf{G}_2$
19 $c_0 \leftarrow (\mathsf{com}, \mathsf{ct}_1, \mathsf{ct}_{\mathsf{rsp}})$
20 $(c_1, \mathsf{k}_1) \xleftarrow{\$} \mathsf{Sim}(\mathsf{pk}_1, \mathsf{pk}_2)$
21 $\beta \xleftarrow{\$} \{0,1\}$
22 $\beta' \leftarrow \mathcal{A}^{\mathsf{G},\mathsf{H}}(\mathsf{sk}_1, \mathsf{pk}_1, \mathsf{pk}_2, c_\beta, \mathsf{k}_\beta)$
23 **return** $[\![\beta = \beta']\!]$

Simulator $\mathsf{Sim}(\mathsf{pk}_1, \mathsf{pk}_2)$

24 Parse $\mathsf{pk}_2 = (\mathsf{pk}_{\mathsf{KEM}}, \mathsf{pk}_{\mathsf{ID}})$
25 $(\mathsf{com}, R) \xleftarrow{\$} \mathsf{KEM.Encaps}_0$
26 $(\mathsf{ct}_1, K) \xleftarrow{\$} \mathsf{KEM.Encaps}_1(\mathsf{pk}_{\mathsf{KEM}}, R)$
27 $\mathsf{ct}_{\mathsf{rsp}} \xleftarrow{\$} \mathsf{RspSpace}$
28 $\mathsf{k} \xleftarrow{\$} \mathcal{K}$
29 $\mathsf{ct} \leftarrow (\mathsf{com}, \mathsf{ct}_1, \mathsf{ct}_{\mathsf{rsp}})$
30 **return** $(\mathsf{ct}, \mathsf{k})$

Random Oracle $\mathsf{G}(\mathsf{pk}_{\mathsf{ID}}, \mathsf{com}, \mathsf{pk}, \mathsf{ct}_1, K)$

31 **if** $\exists(\mathsf{chl}, \mathsf{pad}) : (\mathsf{pk}_{\mathsf{ID}}, \mathsf{com}, \mathsf{pk}, \mathsf{ct}_1, K, (\mathsf{chl}, \mathsf{pad})) \in \mathcal{L}_\mathsf{G}$
32 **return** $(\mathsf{chl}, \mathsf{pad})$
33 $(\mathsf{chl}, \mathsf{pad}) \xleftarrow{\$} \mathsf{ChlSet} \times \mathsf{RspSpace}$
34 $\mathcal{L}_\mathsf{G} \leftarrow \mathcal{L}_\mathsf{G} \cup \{(\mathsf{pk}_{\mathsf{ID}}, \mathsf{com}, \mathsf{pk}, \mathsf{ct}_1, K, (\mathsf{chl}, \mathsf{pad}))\}$
35 **return** $(\mathsf{chl}, \mathsf{pad})$

Random Oracle $\mathsf{H}(K, \mathsf{com}, \mathsf{ct}_1, \mathsf{rsp}, \mathsf{pk}_{\mathsf{SND}}, \mathsf{pk}_{\mathsf{RCV}})$

36 **if** $\exists \mathsf{k} : (K, \mathsf{com}, \mathsf{ct}_1, \mathsf{rsp}, \mathsf{pk}_{\mathsf{SND}}, \mathsf{pk}_{\mathsf{RCV}}, \mathsf{k}) \in \mathcal{L}_\mathsf{H}$
37 **return** k
38 $\mathsf{k} \xleftarrow{\$} \mathcal{K}$
39 $\mathcal{L}_\mathsf{H} \leftarrow \mathcal{L}_\mathsf{H} \cup \{(K, \mathsf{com}, \mathsf{ct}_1, \mathsf{rsp}, \mathsf{pk}_{\mathsf{SND}}, \mathsf{pk}_{\mathsf{RCV}}, \mathsf{k})\}$
40 **return** k

Fig. 21. Games $\mathsf{G}_0 - \mathsf{G}_2$ for the proof of Theorem 3.9.

Game G_2. This game is the same as the previous one except that the encryption of the response, $\mathsf{ct}_{\mathsf{rsp}}$ is replaced by a uniformly random value from the response space $\mathsf{RspSpace}$ (Line 16). Since pad is uniformly random and not used anywhere else, $\mathsf{ct}_{\mathsf{rsp}}$ is information-theoretically indistinguishable from a uniformly random value. Hence it holds

$$\Pr[\mathsf{G}_1^{\mathcal{A}} \Rightarrow 1] = \Pr[\mathsf{G}_2^{\mathcal{A}} \Rightarrow 1].$$

Note that in G_2, the distribution of c_0 and k_0 is the same as for the output of the simulator. Hence, an adversary $\mathcal{A}$ does not have a better chance of winning the game than guessing the challenge bit and we have

$$\Pr[\mathsf{G}_2^{\mathcal{A}} \Rightarrow 1] = \frac{1}{2}.$$

□

E Detailed Security Analysis of SnakeM-Iso

In this section we discuss various attack strategies against SnakeM-Iso (translating to attacks against the underlying assumptions) and show that they do not apply, thus underpinning the overall security of SnakeM-Iso.

E.1 Attacking SnakeM-Iso Using Malicious POKÉ Public Keys

Attack Idea: Recover the commitment randomness via malformed POKÉ public keys.

Target: $\{\mathsf{IMP}, \mathsf{SS}, \mathsf{NM}\}$-Enc

Prevention: Choose $2^a \leq \sqrt{N}$.

One possible way to attack SnakeM-Iso is by using maliciously generated POKÉ public keys. If an honest party encrypts towards such a public key, the attacker is able to efficiently recover

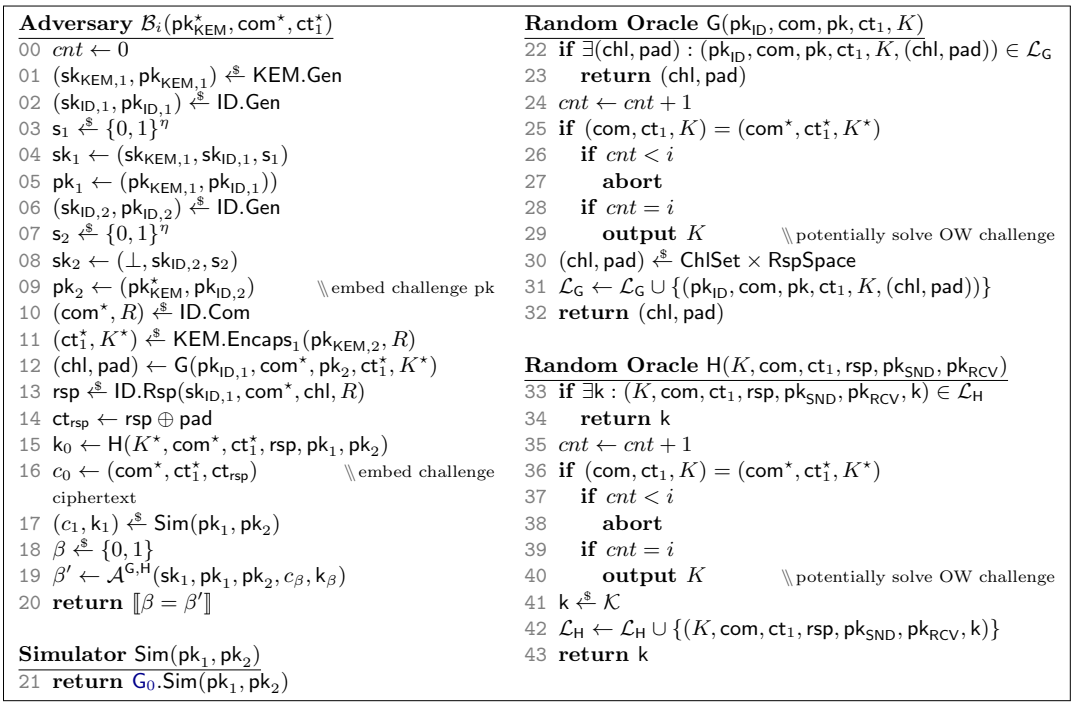

Fig. 22. Adversary $\mathcal{B}_i$ against OW simulating games $\mathsf{G}_{0.i-1}/\mathsf{G}_{0.i}$ for adversary $\mathcal{A}$ from the proof of Theorem 3.9.

the commitment randomness φ_{com}, which combined with the signature yields the $\mathsf{SQIsignHD}^+$ secret key of the honest party.

Concretely, the attack targets the $\{\mathsf{IMP}, \mathsf{SS}, \mathsf{NM}\}$-Enc assumption. In these assumptions, the adversary $\mathcal{A}$ has access to an `Encps` oracle that takes as input (essentially) a POKÉ public key pk and outputs a SnakeM-Iso ciphertext which was generated w.r.t. pk. In particular, the ciphertext contains a POKÉ ciphertext w.r.t. pk as well as a $\mathsf{SQIsignHD}^+$ signature using the encryption randomness as commitment randomness. Let in the following $(\mathsf{sk}^\star_{\mathsf{ID}}, \mathsf{pk}^\star_{\mathsf{ID}})$ be the $\mathsf{SQIsignHD}^+$ key pair in the (say) IMP-Enc game.

Instead of choosing an honest POKÉ public key as described in Section 4, adversary $\mathcal{A}$ chooses a random ℓ-isogeny $\varphi_{\mathsf{skEnc}} : E_0 \to E_{\mathsf{enc}}$ for some small enough $\ell \in \mathbb{N}$. Since ℓ is small, $\ker \varphi_{\mathsf{skEnc}}$ is defined over a small field extension of $\mathbb{F}_{p^2}$ and can hence be considered accessible. It then chooses a random basis $(\widetilde{P}_0, \widetilde{Q}_0)$ of $E_0[2^a]$ under the condition that $\widetilde{P}_0$, $\widetilde{Q}_0$, P_0 and Q_0 generate pairwise distinct subgroups. $\mathcal{A}$ then computes $(\widetilde{P}_1, \widetilde{Q}_1) = (\varphi_{\mathsf{skEnc}}(\widetilde{P}_0), \varphi_{\mathsf{skEnc}}(\widetilde{Q}_0))$ and $(R_1, S_1) = (\varphi_{\mathsf{skEnc}}(R_0), \varphi_{\mathsf{skEnc}}(S_0))$. Lastly, it chooses and a random $X_1, Y_1 \in E_{\mathsf{enc}}[D]$ and sets the POKÉ public key to $\mathsf{pk} = (E_{\mathsf{enc}}, \widetilde{P}_1, \widetilde{Q}_1, R_1, S_1, X_1, Y_1)$.

Next, the adversary queries the `Encps` oracle on pk, receiving a ciphertext $\mathsf{ct} = (\mathsf{com}, \mathsf{ct}_1, \mathsf{rsp})$ and a challenge chl. In the particular case of POKÉ and $\mathsf{SQIsignHD}^+$, the ciphertext ct and challenge chl consist of the following information:

- $\mathsf{com} = E_{\mathsf{com}}$, the codomain of some N-isogeny φ_{com},
- $\mathsf{ct}_1 = (E_{\mathsf{ct}}, P_2, Q_2, \widetilde{P}_3, \widetilde{Q}_3)$, where in particular E_{ct} is the codomain of the push-forward $\varphi'_{\mathsf{com}} = [\varphi_{\mathsf{skEnc}}]_* \varphi_{\mathsf{com}}$, $(P_2, Q_2) = ([\beta_P]\varphi_{\mathsf{com}}(P_0), [\beta_Q]\varphi_{\mathsf{com}}(Q_0))$ and $(\widetilde{P}_3, \widetilde{Q}_3) = ([\beta_P]\varphi'_{\mathsf{com}}(\widetilde{P}_1), [\beta_Q]\varphi'_{\mathsf{com}}(\widetilde{Q}_1))$,
- chl corresponds to an isogeny $\varphi_{\mathsf{chal}} : E_{\mathsf{sig}} \to E_{\mathsf{chal}}$, and
- rsp yields an HD-representation of an isogeny $\varphi_{\mathsf{rsp}} : E_{\mathsf{com}} \to E_{\mathsf{chal}}$.

From the commutativity of the POKÉ square, E_{ct} is the unique codomain of $\varphi'_{\mathsf{skEnc}} = [\varphi_{\mathsf{com}}]_* \varphi_{\mathsf{skEnc}}$. However, because φ_{skEnc} is a *rational* isogeny of small-ish degree ℓ, the adversary can simply

brute-force the push-forward $\varphi'_{\mathsf{skEnc}}$. Subsequently, $\mathcal{A}$ can compute

$$\widetilde{P}_2 = [\beta_P]\varphi_{\mathsf{com}}(\widetilde{P}_0) = [\ell^{-1}]\widehat{\varphi}_{\mathsf{skEnc}}(\widetilde{P}_3), \quad \text{and}$$
$$\widetilde{Q}_2 = [\beta_Q]\varphi_{\mathsf{com}}(\widetilde{Q}_0) = [\ell^{-1}]\widehat{\varphi}_{\mathsf{skEnc}}(\widetilde{Q}_3).$$

Since $P_0, Q_0, \widetilde{P}_0, \widetilde{Q}_0$ generate pairwise distinct subgroups, $\mathcal{A}$ can recover β_P and β_Q from the pairing equation

$$e(\widetilde{P}_0, P_0)^{N\beta_P^2} = e(\widetilde{P}_2, P_2) \;\text{ and }\; e(\widetilde{Q}_0, Q_0)^{N\beta_Q^2} = e(\widetilde{Q}_2, Q_2).$$

Hence $\mathcal{A}$ recovers the images $\varphi_{\mathsf{com}}(P_0)$ and $\varphi_{\mathsf{com}}(Q_0)$, that is, the image of the 2^a-torsion under the N-isogeny φ_{com}. If now $2^a \geq \sqrt{N}$ then $\mathcal{A}$ successfully recovers an efficient representation of φ_{com}, allowing him to recover $\mathsf{sk}^\star_{\mathsf{ID}}$ from the knowledge of $\varphi_{\mathsf{com}}, \varphi_{\mathsf{chal}}$ and φ_{rsp}. However, SnakeM-Iso uses $2^a \approx 2^{\lambda/2}$ and $N \approx 2^{3\lambda}$, hence the attack is not applicable.

E.2 Attacking SnakeM-Iso via a Guessing Strategy

Attack Idea: Recover the commitment randomness by guessing 1/3 of φ_{com} and recover the remaining 2/3 via meet-in-the-middle.

Target: $\{\mathsf{IMP}, \mathsf{SS}, \mathsf{NM}\}$-Enc

Prevention: Choose $N \approx 2^{3\lambda}$.

The next attack again targets the $\{\mathsf{IMP}, \mathsf{SS}, \mathsf{NM}\}$-Enc assumption and proceeds similarly to the adaptive attack outlined in [GPST16]. To this end, the adversary $\mathcal{A}$ guesses a constant fraction $1/\kappa$ of φ'_{com}. Subsequently, $\mathcal{A}$ chooses the points (R_1, S_1) in his public key pk in such a way that if an honest party encrypts towards pk, then the resulting ciphertext is well-formed (i.e. decryptable) if and only if the guess was correct. $\mathcal{A}$ can then brute-force the remaining $(\kappa-1)/\kappa$ fraction of φ_{com} via meet-in-the-middle, resulting in an overall complexity of $\max\{N^{1/\kappa}, N^{(\kappa-1)/2\kappa}\}$. Here, the optimal trade-off is for $\kappa = 3$.

Concretely, let $(\mathsf{pk}, \mathsf{sk})$ be a POKÉ key pair with $\mathsf{pk} = (E_{\mathsf{enc}}, P_1, Q_1, R_1, S_1, X_1, Y_1)$ and $\varphi_{\mathsf{skEnc}} : E_0 \to E_{\mathsf{enc}}$ the corresponding secret key isogeny. Furthermore, let $N = \prod_i \ell_i^{e_i}$ and recall that $\ker \varphi'_{\mathsf{com}} = \langle R_1 + [r_{\mathsf{com}}]S_1\rangle$ for some $r_{\mathsf{com}} \in \mathbb{Z}_N^*$. Furthermore, let $j \in \mathbb{N}$ be such that $n = \prod_{i \leq j} \ell_i^{e_i} \approx N^{1/\kappa}$ and let $\rho \in \mathbb{Z}_n^*$ be a guess for $r_{\mathsf{com}} \bmod n$. By extending the idea in [GPST16, Remark 2] it is possible to choose $(\tilde{R}_1, \tilde{S}_1) \in E_{\mathsf{enc}}[N]$ such that

1. $\langle R_1 + [r_{\mathsf{com}}]S_1\rangle = \langle \tilde{R}_1 + [r_{\mathsf{com}}]\tilde{S}_1\rangle$ if $\rho \equiv r_{\mathsf{com}} \bmod n$,
2. $\langle R_1 + [r_{\mathsf{com}}]S_1\rangle \neq \langle \tilde{R}_1 + [r_{\mathsf{com}}]\tilde{S}_1\rangle$ if $\rho \not\equiv r_{\mathsf{com}} \bmod n$, and
3. $(\tilde{R}_1, \tilde{S}_1)$ is a basis of $E_{\mathsf{enc}}[N]$, implying that both points have order N.

Therefore, if the guess ρ was correct, we have that φ'_{com} stays invariant under the change $(R_1, S_1) \to (\tilde{R}_1, \tilde{S}_1)$. In particular, we still have $\varphi'_{\mathsf{com}} = [\varphi_{\mathsf{skEnc}}]_*\varphi_{\mathsf{com}}$ and the POKÉ diagram commutes. Therefore, it is still possible to compute $\varphi'_{\mathsf{skEnc}} = [\varphi_{\mathsf{com}}]_*\varphi_{\mathsf{skEnc}}$.

However, if the guess ρ was incorrect, the isogeny with kernel $\langle \tilde{R}_1 + [r_{\mathsf{com}}]\tilde{S}_1\rangle$ is distinct from $[\varphi_{\mathsf{skEnc}}]_*\varphi_{\mathsf{com}}$. Furthermore, the POKÉ diagram does not commute anymore, making it impossible to compute $\varphi'_{\mathsf{skEnc}} = [\varphi_{\mathsf{com}}]_*\varphi_{\mathsf{skEnc}}$.

To summarize, the adversary $\mathcal{A}$ first chooses an honest POKÉ key pair $(\mathsf{pk}, \mathsf{sk})$ with $\mathsf{pk} = (E_{\mathsf{enc}}, P_1, Q_1, R_1, S_1, X_1, Y_1)$. He then makes a guess ρ and replaces (R_1, S_1) with the corresponding $(\tilde{R}_1, \tilde{S}_1)$, yielding the public key $\tilde{\mathsf{pk}}$. He then queries `Encps` on $\tilde{\mathsf{pk}}$, receiving (among other things) a POKÉ ciphertext ct that was created using some secret randomness r_{com}. Now $\mathcal{A}$ is able to decrypt ct if and only if the guess ρ for $r_{\mathsf{com}} \bmod n$ was correct. If the guess was incorrect, $\mathcal{A}$ proceeds to query `Encps` on $\tilde{\mathsf{pk}}$ until the chosen r_{com} coincides with his guess ρ. After an expected $N^{1/\kappa}$ queries, he will succeed and thus knows a $1/\kappa$ fraction of r_{com} (or equivalently φ_{com}). The remaining $(\kappa-1)/\kappa$ fraction of φ_{com} can be recovered via meet-in-the-middle, resulting in a runtime of $N^{(\kappa-1)/2\kappa}$. Once φ_{com} is recovered, the adversary can recover

the $\mathsf{SQIsignHD}^+$ secret key as outlined in the previous attack, thus successfully breaking the assumption.

The attack is prevented by choosing N large enough such that $N^{1/3} \in \Theta(2^\lambda)$, which matches our proposed parameters since there $N \geq 2^{3\lambda}$.

Remark E.1. We highlight that the above attack corresponds to a "one-shot" version of [GPST16]. Indeed, [GPST16] uses the above idea to recover a secret isogeny step-by-step, yielding a polynomial-time attack. However, in their setting the secret isogeny is *static*, which is not the case in $\mathsf{SnakeM\text{-}Iso}$ since φ_{com} changes in each encapsulation. Hence, the exponential-time "one-shot" version above seems to be optimal.

E.3 Attacking OW-KCA

One may wonder if a GPST/MOXZ-style attack [GPST16, MOXZ24] can be mounted against φ_{skEnc} instead of φ_{com}, since the former is indeed static in the $\mathsf{OW\text{-}KCA}$ game. However, such an attack seems out of reach for two reasons:

- The degree of φ_{skEnc} is unknown and prime, hence there does not seem to be a natural way to split φ_{skEnc} into a $1/\kappa$ fraction.
- We include an additional check in Figure 10, Line 28 that checks whether the provided torsion points are actual faithful images under $\varphi'_{\mathsf{skEnc}}$.

F ElGamal-Schnorr AKEM

We first recall the AKEM constructed from ElGamal and Schnorr as described in the technical overview (cf. Section 1.1). For a group $(\mathbb{G}, p, g)$ of prime order p and generator g, an ElGamal public key $\mathsf{pk}_{\mathsf{KEM}}$ and a Schnorr secret key $\mathsf{sk}_{\mathsf{SIG}}$, we write the encapsulation algorithm as

$$(\mathsf{ct} = (g^r, \mathsf{pk}_{\mathsf{KEM}}^r \cdot \mathsf{k}), \mathsf{k}) \leftarrow \mathsf{Encaps}(\mathsf{pk}_{\mathsf{KEM}}; r),$$

and the signing algorithm as

$$(\mathsf{com} = g^r, \mathsf{rsp} = r + \mathsf{sk}_{\mathsf{SIG}} \cdot \mathsf{chl}) \xleftarrow{\$} \mathsf{Sign}(\mathsf{sk}_{\mathsf{SIG}}, m; r),$$

where $\mathsf{chl} = \mathsf{G}(\mathsf{com}, m)$ and G is modeled as a random oracle. In the SnakeM construction we combine the two by signing $m = (\mathsf{k}, \mathsf{ctxt})$ and deriving the AKEM key has $\mathsf{H}(\mathsf{k}, \mathsf{ctxt})$, where H is another random oracle and ctxt contains (parts of) the transcript and public keys.

Compatibility, Correctness, and Spreadness. When viewing the generation of the first ElGamal ciphertext component as Encaps_0, it is easy to see that it is compatible with the commitment generation of the underlying Schnorr ID scheme. Further, both schemes are perfectly correct. The commitment spreadness is $\gamma_{\mathsf{ID}} = 1/p$ and the KEM spreadness is $\gamma_{\mathsf{KEM}} = 1$ since the second ciphertext component does not introduce any new randomness.

Security. For our composition theorems Theorems 3.7 to 3.9, we need to show that ElGamal is $\mathsf{OW\text{-}KCA}$ secure and that Schnorr is $\mathsf{IMP\text{-}Enc}, \mathsf{NM\text{-}Enc}, \mathsf{SS\text{-}Enc}$ secure, that is, even in the presence of a transcript oracle that also outputs the second part of the ElGamal ciphertext. The former holds under the Strong Computational Diffie-Hellman assumption. The latter notions are more interesting.

Why (Almost) Standard ID Notions are Sufficient Here. We first look at $\mathsf{IMP\text{-}Enc}$ security. We would like to show that $\mathsf{IMP\text{-}KOA}$ security of Schnorr implies $\mathsf{IMP\text{-}Enc}$ for Schnorr and ElGamal. For this we need to show that we can simulate the `Encps` and `Check` oracle. We first look at the former. We want to construct a reduction $\mathcal{B}$ against $\mathsf{IMP\text{-}KOA}$ that gets an (honestly generated) Schnorr public key $\mathsf{pk}_{\mathsf{SIG}} = g^x$ as input. It simulates `Encps` queries for a (possibly adversarially generated) ElGamal public key $\mathsf{pk}_{\mathsf{KEM}} = g^y$ using perfect honest-verifier zero-knowledge, i.e., it computes $\mathsf{com}, \mathsf{chl}, \mathsf{rsp}$ by first picking $\mathsf{rsp}, \mathsf{chl} \xleftarrow{\$} \mathbb{Z}_p$ and then

computing $\mathsf{com} = g^{\mathsf{rsp}} \cdot \mathsf{pk}_{\mathsf{SIG}}^{-\mathsf{chl}}$. It then picks $\mathsf{ct}_1 \xleftarrow{\$} \mathbb{G}$. Since k is randomly sampled, ct_1 is correctly distributed. More concretely, we can express it as $\mathsf{ct}_1 = \mathsf{com}^y \cdot \mathsf{k} = g^{(\mathsf{rsp}-x\cdot\mathsf{chl})y} \cdot \mathsf{k}$. However, $\mathcal{B}$ also need to simulate $\texttt{Check}$. Since the adversary can choose $\mathsf{pk}_{\mathsf{KEM}}$ such that it knows y, it can decrypt $\mathsf{com}, \mathsf{ct}_1$ to obtain k. Hence, $\mathcal{B}$ needs to recognize $\texttt{Check}$ queries of the correct form $(\mathsf{k}, \mathsf{pk}_{\mathsf{KEM}}, \mathsf{com}, \mathsf{ct}_1)$. Therefore, standard IMP-KOA security is not sufficient, but we need to provide the reduction with an additional $\texttt{DDH}_x(\cdot,\cdot)$ oracle which on input Y, Z outputs whether $Z = Y^x$. While this is stronger than standard IMP-KOA security, it does not rely on the ElGamal KEM directly. A similar argument holds for special soundness (cf. also Theorem C.3). Further, we achieve non-malleability since Schnorr has unique responses and therefore perfect non-malleability.

Necessity of Enc-Oracle for our Construction. The above implication highly relies on the structure of prime-order groups and perfect HVZK. For non-oriented isogenies, and also in general, an analogue to the $\texttt{DDH}$ oracle may not exist and one rather needs a "key-checking" oracle for adversarially generated public keys. Similarly, HVZK may not be perfect, as is the case for SQIsignHD$^+$ due to current limitations regarding randomly sampling HD isogenies. For the same reason, it may not be possibly to sample the second ciphertext component perfectly, as is the case for POKÉ. Hence, it is hard to separate the two primitives from each other and our notions are more involved.