Group Action Key Encapsulation and Non-Interactive Key Exchange in the QROM

Julien Duman, Dominik Hartmann, Eike Kiltz, Sabrina Kunzweiler, Jonas Lehmann, Doreen Riepel

Ruhr-Universität Bochum, Bochum, Germany
{julien.duman,dominik.hartmann,eike.kiltz,
sabrina.kunzweiler,jonas.lehmann-c6j,doreen.riepel}@rub.de

Abstract. In the context of quantum-resistant cryptography, cryptographic group actions offer an abstraction of isogeny-based cryptography in the Commutative Supersingular Isogeny Diffie-Hellman (CSIDH) setting. In this work, we revisit the security of two previously proposed natural protocols: the Group Action Hashed ElGamal key encapsulation mechanism (GA-HEG KEM) and the Group Action Hashed Diffie-Hellman non-interactive key-exchange (GA-HDH NIKE) protocol. The latter protocol has already been considered to be used in practical protocols such as Post-Quantum WireGuard (S&P '21) and OPTLS (CCS '20).

We prove that *active* security of the two protocols in the Quantum Random Oracle Model (QROM) inherently relies on very strong variants of the Group Action Strong CDH problem, where the adversary is given arbitrary *quantum access* to a DDH oracle. That is, quantum accessible Strong CDH assumptions are not only sufficient but also necessary to prove active security of the GA-HEG KEM and the GA-HDH NIKE protocols.

Furthermore, we propose variants of the protocols with QROM security from the classical Strong CDH assumption, i.e., CDH with classical access to the DDH oracle. Our first variant uses key confirmation and can therefore only be applied in the KEM setting. Our second but considerably less efficient variant is based on the twinning technique by Cash et al. (EUROCRYPT '08) and in particular yields the first actively secure isogeny-based NIKE with QROM security from the standard CDH assumption.

Keywords: Group actions, CSIDH, Hashed ElGamal, NIKE, QROM, twinning

1 Introduction

A non-interactive key exchange (NIKE) is a protocol that allows two parties to establish a common secret key in a non-interactive way. The first and most famous NIKE is the Diffie-Hellman key exchange [16] which forms the basis for a lot of other cryptographic protocols like ElGamal [19]. Most notably however, the existence of a secure NIKE implies secure key encapsulation mechanisms (KEM)

(and hence public-key encryption) and authenticated key exchange (AKE) [21]. A NIKE can therefore be seen as one of the most basic and important primitives in cryptography.

The emergence of quantum computing however continues to have an unprecedented impact on public key cryptography. When scaled to a suitable size, quantum computers pose a threat to almost all classical public-key primitives, including Diffie-Hellman and ElGamal [36]. To mitigate this threat, researchers started building quantum resisting public-key cryptography based on certain quantum-hard problems on codes, lattices and isogenies. Even though quantum-resistant public-key encryption from lattices seems to offer the favorable trade-off over codes and isogenies in terms of speed, ciphertext expansion, and security, building an efficient (even passively secure) NIKE from codes or lattices remains an unsolved research problem.

Isogeny-based Cryptography. A promising alternative approach to post-quantum security is based on isogenies. An isogeny is a non-constant homomorphism between elliptic curves. In an algebraic context, isogenies can be used to build a commutative group action that behaves similarly to exponentiation in finite fields. This was first observed by Couveignes [14] and independently by Rostovtsev and Stolbunov [34]. The first practical instantiation was obtained by Castryck et al. [12] which in contrast to previous work uses the group action on the set of *supersingular* elliptic curves. Throughout this paper, we will use the abstract framework of cryptographic group actions introduced by Alamati et al. [2] to model isogeny-based constructions. (See Section 2.3 for formal definitions.) At a syntactical level, cryptographic group actions allow for a simple Group Action Diffie-Hellman (GA-DH) key exchange and Group Action ElGamal (GA-EG) public-key encryption scheme. With this abstraction in mind, the famous Commutative Supersingular Isogeny Diffie-Hellman (CSIDH) key exchange protocol of [12] can be seen as a specific instantiation of GA-DH.

For cryptographic group actions, the analog of the traditional Computational Diffie-Hellman assumption (over prime-order groups) is the *Group Action Computational Diffie-Hellman assumption* (GA-CDH) [14,34,12,2], see also Definition 5. GA-CDH is sufficient to prove passive security of "hashed versions" of GA-DH and GA-EG in the random oracle model. In analogy to the prime-order group setting, for active security one requires a "strong" type of Computational Diffie-Hellman assumption [1]. Providing the adversary additional access to a Group Action Decisional Diffie-Hellman oracle GA-DDH$(\cdot,\cdot)$, i.e. an oracle which tells us whether a pair of elements forms a Diffie-Hellman tuple, defines the *Group Action Strong Computational Diffie-Hellman assumption* (GA-StCDH). The prefix *strong* refers to the fact that the first input to this oracle is fixed (as opposed to the stronger and non-falsifiable *gap* assumptions). This assumption is well-known in the standard prime-order group setting and has already been used in proving active security of several protocols [28,15,38] in the group action setting as well.[1]

[1] We stress that GA-StCDH over standard cryptographic group actions is well defined (and falsifiable), even though it is an interactive assumption. Furthermore, for some

Quantum Random Oracle Model. The random-oracle model (ROM) [7] is commonly used in modern cryptography to argue *practical security* of cryptographic schemes. Adversaries with access to quantum computers will be able to implement the hash function on those, and therefore can evaluate the hash function on arbitrary quantum superpositions. To account for this gain in capabilities, the *quantum(-accessible)* random-oracle model (QROM) has been introduced [9]. The QROM has become the accepted model for proving post-quantum security and it is generally believed that proofs in the classical ROM are not sufficient to claim post-quantum security.

Actively secure KEMs and NIKE Protocols. In this work we are interested in constructing actively (i.e. IND-CCA) secure KEMs and actively secure NIKE protocols over cryptographic group actions.

Let us first look at the simpler case of KEMs. Generally speaking, we know of two natural approaches to build efficient IND-CCA secure KEMs. The first approach is generic and applies the Fujisaki-Okamoto (FO) transform [22,24] to an IND-CPA secure PKE scheme (such as GA-EG) to obtain an IND-CCA secure KEM, with provable security in the QROM. The second, non-generic approach is to adapt the well-known (prime-order group) Hashed ElGamal encryption framework of [1] to group actions by "hashing the raw KEM key" to obtain the *Group Action Hashed ElGamal KEM* (GA-HEG). Indeed, [38] proved the security of GA-HEG (called CSIDH-ECIES in [38]) under the GA-StCDH assumption in the ROM.[2] GA-HEG was implicitly and explicitly used in [28,15,38] and its active (IND-CCA) security in the QROM was left as an open problem in [38].[3]

For building an actively secure NIKE, one cannot apply the FO transformation and hence has to resort to adapting the (prime-order group) Hashed Diffie-Hellman NIKE [21] to obtain the *Group Action Hashed Diffie-Hellman* NIKE protocol (GA-HDH). To the best of our knowledge, the active security of the GA-HDH NIKE has not been formally analyzed yet, not even in the ROM. This is in particular unsatisfactory since GA-HDH has already been considered to be used in practical protocols such as Post-Quantum WireGuard [26] and OPTLS [35].

In conclusion, while the IND-CCA security of GA-HEG in the ROM is known to be implied by the GA-StCDH assumption, it remains an open problem to prove its IND-CCA security in the QROM (under any assumption). Similarly,

---

groups actions (i.e., ones implied by cryptographic pairings over prime-order groups) the Decisional Diffie-Hellman oracle is publicly computable and hence GA-StCDH becomes non-interactive.

[2] The QROM proof of a variant called CSIDH-PSEC in [38] is severely flawed (see the full version [18] for details).

[3] There also exist IND-CCA secure PKE schemes constructed directly from CSIDH, using additional structure of the elliptic curves. [31] proposed the SimS scheme which is an extension of SiGamal [20] and relies on a non-standard knowledge-of-exponent assumption to achieve IND-CCA security in the standard model. These protocols and assumptions cannot be modeled in the abstract group action framework.

studying the active security of the GA-HDH NIKE in the QROM also remains an open problem.

1.1 Our Contributions

In this paper we study the active security of the Group Action Hashed Diffie-Hellman NIKE GA-HDH and the Group Action Hashed ElGamal KEM GA-HEG in the QROM, and derive variants thereof with improved security guarantees. We now discuss our results in detail. For an overview of our results obtained for KEMs we refer to Figure 1.

GA-HEG KEM and GA-HDH NIKE. It is easy to see that in the (non-quantum) ROM the active security of GA-HEG is implied by the GA-StCDH assumption. The first main contribution of this paper is to notice that in the QROM one requires a considerably stronger assumptions to prove security of GA-HEG. To this end we define the following two stronger variants of GA-StCDH which differ only in the access to the decision oracle (for implications see Figure 1):

- Partial Quantum access Strong Diffie-Hellman (GA-PQ-StCDH): the first input to the GA-DDH$(\cdot,\cdot)$ oracle is classical and the second is in quantum superposition.
- Full Quantum access Strong Diffie-Hellman (GA-FQ-StCDH): both inputs to the GA-DDH$(\cdot,\cdot)$ oracle are in quantum superposition.

Similar to the QROM, the answer of a quantum superposition query to the two quantum-accessible GA-DDH oracles is also in quantum superposition.

Our first main theorem states that under the GA-FQ-StCDH assumption (full quantum access to the DDH oracle), GA-HEG is IND-CCA secure in the QROM. Furthermore, IND-CCA security in the QROM of GA-HEG implies the GA-PQ-StCDH assumption (partial quantum access to the DDH oracle), hence GA-PQ-StCDH is necessary for GA-HEG's IND-CCA security. The situation for the GA-HDH NIKE is similar, with the difference that "double base" strong assumptions (called GA-DPQ-StCDH and GA-DFQ-StCDH) are required.

This leaves us in the alarming situation that active security of GA-HEG and GA-HDH inherently require a group action CDH assumption with quantum access to the DDH oracle. Due to the quantum access, the latter assumptions cannot be considered as standard assumptions and require further cryptanalysis before we can recommend using GA-HEG KEM and GA-HDH NIKE in practice.

We will now propose two modifications to get security without quantum access to the decision oracles. The first and more efficient modification is using "key confirmation" and only works for KEMs. The second and less efficient modification relies on the "twinning technique" and can be applied to NIKEs and KEMs.

GA-HEG-KC KEM: Key Confirmation. Our first method is to update GA-HEG the KEM with a key confirmation hash, i.e., every ciphertext additionally contains a hash of the "raw KEM key". This only increases the ciphertext size by one hash, but allows for a different IND-CCA proof technique in the QROM. To

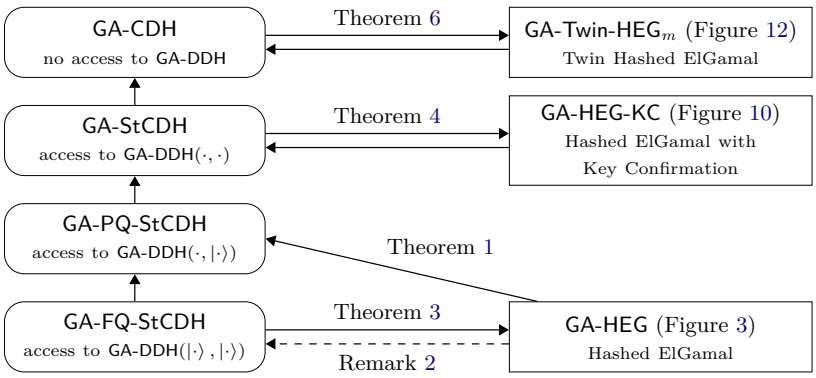

Fig. 1. Overview of our assumptions and results for different variants of hashed ElGamal. The assumptions (elements with rounded corners) are given in Definitions 5 and 6. Solid arrows without indication of a theorem correspond to trivial implications. For the assumptions the only difference is a more limited access to the decision oracle GA-DDH, where $|\cdot\rangle$ denotes quantum access. The dashed arrow holds for *quantum* security, where the adversary is allowed to issue decapsulation queries in superposition.

be more precise, in the classical ROM, one can use the additional hash to extract the secret information from a ciphertext. In the QROM, this is more involved, but we can use the extractable oracle simulator from [17] to use similar techniques and give a security proof only relying on the more standard GA-StCDH assumption. Specifically, we rely on the fact that decapsulation queries are classical, which allows us to partially measure the simulated random oracle and extract its queries without noticeably disturbing its quantum state.

Unfortunately, it is not possible to use key confirmation in a NIKE setting.

GA-Twin-HEG$_m$ KEM and GA-Twin-HDH$_m$ NIKE: Twinning. We show how to use the twinning technique [11] in the context of group actions to build an actively secure KEM and NIKE from the standard GA-CDH assumption (no DDH oracle access) in the QROM. Since group actions only have limited structure compared to prime-order groups, it seems unavoidable to pursue a bit-wise approach for the twinning technique. Our main leverage is a trapdoor test which allows us to check if several adversarial inputs form a Diffie-Hellman tuple with the challenge elements. The failure probability of this trapdoor test can be reduced to the generic quantum search problem, for which the quantum hardness is optimally bounded by the Grover algorithm. Although this approach does not achieve practical efficiency, it is interesting from a theoretical viewpoint. We specify the twinning parameter m for 128-bit security to instantiate the twinned versions of our GA-Twin-HEG$_m$ KEM and GA-Twin-HDH$_m$ NIKE. At this point we want to highlight that our GA-Twin-HDH$_m$ protocol is the only known NIKE with active security from a standard assumption (without quantum accessible DDH oracles).

Efficiency Comparison. In Table 1 in Section 6, we give an overview of the schemes analyzed in this work and compare them to the FO variant GA-EG-FO

of Group Action ElGamal. The KEM variants GA-HEG and GA-Twin-HEG$_m$ share the same minimal ciphertext size but we cannot recommend using them since GA-HEG's security inherently relies on the GA-FQ-StCDH assumption (with quantum accessible DDH oracle) and GA-Twin-HEG$_m$ is computationally very expensive. In comparison, the KEM variants GA-HEG-KC and GA-EG-FO only add one additional hash to the ciphertext but offer security from standard assumptions. Here GA-HEG-KC is preferable since decapsulation is about twice as efficient as in GA-EG-FO (due to FO's re-encryption).

As for the more important case of NIKEs, one either has to use the efficient GA-HDH variant with security under the GA-DFQ-StCDH assumption (with quantum accessible DDH oracles) or use the inefficient GA-Twin-HDH$_m$ NIKE. We leave it as an important open problem to construct a practically efficient actively secure NIKE under a standard hardness assumption.

QROM Proof Details. One of the standard tools to prove security in the QROM is the O2H [37] lemma, which unfortunately leads to quite loose bounds. Recently, there has been a lot of progress in developing new variants which give tighter bounds, such as the measure-rewind-measure O2H (MRM-O2H) [30] lemma. While these variants give usually tighter bounds, they can often only be applied in more limited scenarios due to additional constraints. In our work we show how to apply MRM to GA-HEG and GA-Twin-HEG$_m$ to obtain tighter bounds than the by applying the original O2H lemma. For proving GA-HEG-KC we need to extract the preimages of the key-confirmation hash. We use the extractable random-oracle simulator of [17], which allows use to to prove it from the GA-StCDH assumption.

For GA-Twin-HEG$_m$ and GA-Twin-HDH$_m$, the main tool to remove the need for the GA-StCDH is the trapdoor test. While it is easy to show its indistinguishability for regular groups in the standard model, it is unclear whether or not a quantum adversary has a significant advantage against the trapdoor test compared to a classical adversary. We solve the second problem by showing that the indistinguishability of the trapdoor test can be (tightly) reduced to the Generic Distinguishing Problem (GDP). This allows us to use well-known results on the hardness of quantum search to bound the advantage of such adversaries and apply the trapdoor test as a substitute for the decision oracle of the GA-StCDH assumption.

1.2 Further Applications

We believe our QROM analysis carries over to the following primitives and constructions.

Authenticated Key Exchange. Kawashima et al. as well as de Kock et al. [28,15] translated the Diffie-Hellman based AKE protocol of [13] to the CSIDH setting and proved security in the ROM assuming the GA-StCDH assumption. However, both works left it as an open question to prove security in the QROM. Our analysis demonstrates that this proof will only work assuming (at least partial) quantum access to the decision oracle. In this case, our proof techniques

carry over directly. Alternatively, we can also extend the AKE protocol by an additional round to include key confirmation. Using the same technique as in our result on hashed ElGamal with key confirmation will allow to prove security of this extended AKE protocol in the QROM based on the GA-StCDH assumption without quantum access to the decision oracle. However, the additional benefit here is that key confirmation enables explicit authentication, whereas the protocol without key confirmation only achieves implicit authentication.

Signcryption and Authenticated KEMs. The DH-AKEM which was analyzed in the context of the HPKE standard [3] can easily be translated to the group action setting. The scheme is syntactically a signcryption KEM and will be combined with a symmetric encryption scheme. This construction, also named the authenticated mode of HPKE, was proposed to be used in the Message Layer Security (MLS) secure group messaging protocol [6] and the Encrypted Server Name Indication (ESNI) extension for TLS 1.3 [32]. So far, a post-quantum secure instantiation was not proposed, but our results show how to prove security of a group action based construction in the QROM under GA-FQ-StCDH (full quantum access to the decision oracle). Alternatively, we can also extend the scheme by key confirmation and prove security under GA-StCDH.

Post-Quantum Secure TLS. Currently, there is a great effort in replacing the Diffie-Hellman based approach in the TLS handshake by a post-quantum secure alternative. In order to avoid signature schemes which are rather inefficient, a generic KEM-based approach was considered to allow for an easy instantiation [35], however at the cost of efficiency since it requires an additional round. Instead of signatures, it is also possible to use a NIKE directly, as considered for the case of long-term Diffie-Hellman keys in the OPTLS protocol by Krawczyk and Wee in [29] and in a subsequent IETF draft [33]. In this case, a security analysis of the group-action NIKE in the QROM is crucial and our work provides the first results in this direction, namely that a security proof for group action OPTLS will need to rely at least on the GA-PQ-StCDH assumption (partial quantum access to the decision oracles) and is implied by the GA-FQ-StCDH assumption (full quantum access).

More Applications. In the group setting, Hashed ElGamal can be used to build multi-recipient multi-message PKE (mmPKE) by using the same randomness for multiple messages. This reduces sender bandwidth and computation substantially and can be used in Continuous Group Key Agreement (CGKA), which underlies modern and scalable Secure Group Messaging (SGM) such as MLS [6] to significantly improve performance [4]. Since GA-HEG has an identical structure, reusing randomness can yield a similar construction with post-quantum security. This is a first step towards efficient, post-quantum secure SGM.

2 Preliminaries

For integers m, n where $m < n$, $[m, n]$ denotes the set $\{m, m+1, ..., n\}$. For $m = 1$, we simply write $[n]$. By $\log(x)$ we denote the logarithm over the reals

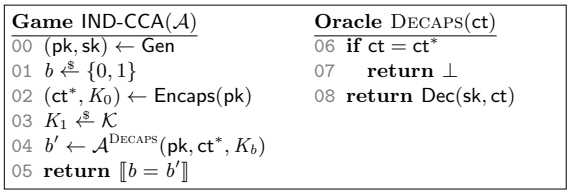

Fig. 2. The IND-CCA game for a key encapsulation mechanism KEM.

with base 2. For a (finite) set S, $s \xleftarrow{\$} S$ denotes that s is sampled uniformly and independently at random from S. $y \leftarrow \mathcal{A}(x_1, x_2, ...)$ denotes that on input $x_1, x_2, ...$ the probabilistic algorithm $\mathcal{A}$ returns y. $\mathcal{A}^{\mathrm{O}}$ denotes that algorithm $\mathcal{A}$ has access to oracle O. An adversary is a probabilistic algorithm. We will use code-based games, where $\Pr[\mathrm{G} \Rightarrow 1]$ denotes the probability that the final output of game G is 1. The notation $[\![B]\!]$, where B is a boolean statement, refers to a bit that is 1 if the statement is true and 0 otherwise. For all algorithms and oracles, we implicitly require that they check whether (adversarial) inputs are from the expected input space. If this is not the case, the algorithm (oracle) will simply return a failure symbol $\perp$.

2.1 Key Encapsulation Mechanisms

Syntax. Let $\mathcal{PK}$, $\mathcal{SK}$, $\mathcal{C}$, $\mathcal{K}$ be sets. A *key encapsulation mechanism* $\mathsf{KEM} = (\mathsf{Gen}, \mathsf{Encaps}, \mathsf{Decaps})$ consists of the following three algorithms

- Gen: The key generation algorithm outputs a public key $\mathsf{pk} \in \mathcal{PK}$ and a secret key $\mathsf{sk} \in \mathcal{SK}$.
- $\mathsf{Encaps}(\mathsf{pk})$: On input a public key pk, the encapsulation algorithm returns a ciphertext $\mathsf{ct} \in \mathcal{C}$ and a key $K \in \mathcal{K}$, where ct is an encapsulation of K.
- $\mathsf{Decaps}(\mathsf{sk}, \mathsf{ct})$: On input a secret key sk and a ciphertext ct, the decapsulation algorithm returns a key $K \in \mathcal{K}$ or a special failure symbol $\perp$.

We require perfect correctness, i.e. for all $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}$, $(\mathsf{ct}, K) \leftarrow \mathsf{Encaps}(\mathsf{pk})$, we have $\mathsf{Decaps}(\mathsf{sk}, \mathsf{ct}) = K$.

Definition 1 (Security against Chosen Ciphertext Attacks (IND-CCA)). *Consider the* IND-CCA *security game in Figure 2. For a key encapsulation mechanism* KEM *we define the advantage of $\mathcal{A}$ winning the game as*

$$\mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathsf{KEM}}(\mathcal{A}) := |\Pr[\mathsf{IND\text{-}CCA}(\mathcal{A}) \Rightarrow 1] - 1/2|\,.$$

2.2 Non-Interactive Key Exchange

We recall syntax and the CKS security model of a Non-Interactive Key Exchange (NIKE) scheme, as defined in [11,21].

Syntax. A non-interactive key exchange scheme NIKE consists of three algorithms NIKE.Setup, NIKE.Gen and NIKE.SharedKey together with an identity

space $\mathcal{ID}$ and a shared key space $\mathcal{SHK}$, where identities in the scheme are only used to track which public key is associated to which user.

- $\mathsf{NIKE.Setup}$: The setup algorithm outputs a set of public parameters pp.
- $\mathsf{NIKE.Gen}(\mathsf{pp}, \mathsf{ID})$: On input pp and $\mathsf{ID} \in \mathcal{ID}$, the key generation algorithm outputs a public key pk and a secret key sk.
- $\mathsf{NIKE.SharedKey}(\mathsf{ID}_1, \mathsf{pk}_1, \mathsf{ID}_2, \mathsf{sk}_2)$: On input $\mathsf{ID}_1 \in \mathcal{ID}$ together with a public key pk_1 and $\mathsf{ID}_2 \in \mathcal{ID}$ together with a secret key sk_2, the shared key algorithm outputs a shared key K. In case $\mathsf{ID}_1 = \mathsf{ID}_2$, the algorithm outputs a failure symbol $\perp$.

Correctness. We require that for any pair of identities $\mathsf{ID}_1, \mathsf{ID}_2 \in \mathcal{ID}$ and any corresponding key pairs $(\mathsf{pk}_1, \mathsf{sk}_1)$ and $(\mathsf{pk}_2, \mathsf{sk}_2)$, it holds that

$$\mathsf{NIKE.SharedKey}(\mathsf{ID}_1, \mathsf{pk}_1, \mathsf{ID}_2, \mathsf{sk}_2) = \mathsf{NIKE.SharedKey}(\mathsf{ID}_2, \mathsf{pk}_2, \mathsf{ID}_1, \mathsf{sk}_1) \ .$$

CKS Security Model. The security of a NIKE protocol is modeled as a game between a challenger and an adversary $\mathcal{A}$. First, the challenger runs $\mathsf{NIKE.Setup}$ to generate the public parameter pp which it outputs to $\mathcal{A}$. The challenger also draws a random bit b and gives $\mathcal{A}$ access to the following oracles.

- RegisterHonest: $\mathcal{A}$ supplies an identity $\mathsf{ID} \in \mathcal{ID}$ and the challenger runs $\mathsf{NIKE.Gen}(\mathsf{pp}, \mathsf{ID})$ to generate a key pair $(\mathsf{pk}, \mathsf{sk})$. It records $(\mathit{honest}, \mathsf{ID}, \mathsf{pk}, \mathsf{sk})$ and returns the public key pk to $\mathcal{A}$.
- RegisterCorrupt: $\mathcal{A}$ supplies an identity $\mathsf{ID} \in \mathcal{ID}$ and a public key pk and the challenger records $(\mathit{corrupt}, \mathsf{ID}, \mathsf{pk}, \perp)$. If $\mathcal{A}$ issues a query with the same ID again later, only the most recent entry is kept. Note here that we do not require that $\mathcal{A}$ knows the corresponding secret key.
- CorruptReveal: $\mathcal{A}$ supplies two identities ID_1 and ID_2 with the restriction that one identity was registered as *honest* and the other one as *corrupt*, otherwise the oracle returns $\perp$. The challenger looks in its record to fetch the secret key of the honest party and the public key of the corrupted party. If ID_1 was honest, it computes and returns $\mathsf{NIKE.SharedKey}(\mathsf{ID}_2, \mathsf{pk}_2, \mathsf{ID}_1, \mathsf{sk}_1)$ and otherwise $\mathsf{NIKE.SharedKey}(\mathsf{ID}_1, \mathsf{pk}_1, \mathsf{ID}_2, \mathsf{sk}_2)$.
- Test: $\mathcal{A}$ supplies two identities ID_1 and ID_2 with the restriction that both were registered as *honest* and $\mathsf{ID}_1 \neq \mathsf{ID}_2$, otherwise the oracle returns $\perp$. The challenger fetches the public key of ID_1 and the secret key of ID_2 from its records and computes $K_0 = \mathsf{NIKE.SharedKey}(\mathsf{ID}_1, \mathsf{pk}_1, \mathsf{ID}_2, \mathsf{sk}_2)$. It also chooses a random key $K_1 \xleftarrow{\$} \mathcal{SHK}$ and records it for later. It outputs K_b, depending on the bit b chosen at the beginning. If $b = 1$ and $\mathcal{A}$ queries the same identities again, in either order, the recorded key is output again.

The oracles can be queried adaptively and an arbitrary number of times. We require that no identity that was registered as corrupt can be later registered as honest, and vice versa. Finally, the adversary outputs a bit b'.

Definition 2 (Security of NIKE). *Consider the* CKS *security game as described above. Then the advantage of adversary $\mathcal{A}$ against a non-interactive key exchange scheme* NIKE *is defined as*

$$\mathsf{Adv}^{\mathsf{CKS}}_{\mathsf{NIKE}}(\mathcal{A}) := |\Pr[b = b'] - 1/2| \ .$$

2.3 (Restricted) Effective Group Actions

We recall the definition of (restricted) effective group actions from [2], which provides an abstract framework to build cryptographic primitives relying on isogeny-based assumptions such as CSIDH.

Definition 3 (Group Action). *Let $(\mathcal{G}, \cdot)$ be a group with identity element $e \in \mathcal{G}$, and $\mathcal{X}$ a set. A map*

$$\star : \mathcal{G} \times \mathcal{X} \to \mathcal{X}$$

is a group action if it satisfies the following properties:

1. *Identity: $e \star x = x$ for all $x \in \mathcal{X}$.*
2. *Compatibility: $(g \cdot h) \star x = g \star (h \star x)$ for all $g, h \in \mathcal{G}$ and $x \in \mathcal{X}$.*

Remark 1. Throughout this paper, we only consider group actions, where $\mathcal{G}$ is commutative. Moreover we assume that the group action is regular. This means that for any $x, y \in \mathcal{X}$ there exists precisely one $g \in \mathcal{G}$ satisfying $y = g \star x$.

Definition 4 (Effective Group Action). *Let $(\mathcal{G}, \mathcal{X}, \star)$ be a group action satisfying the following properties:*

1. *$\mathcal{G}$ is finite and there exist efficient (PPT) algorithms for membership testing, equality testing, (random) sampling, group operation and inversion.*
2. *The set $\mathcal{X}$ is finite and there exist efficient algorithms for membership testing and to compute a unique representation.*
3. *There exists a distinguished element $\tilde{x} \in \mathcal{X}$ with known representation.*
4. *There exists an efficient algorithm to evaluate the group action, i.e. to compute $g \star x$ given g and x.*

Then we call $\tilde{x} \in \mathcal{X}$ the origin and $(\mathcal{G}, \mathcal{X}, \star, \tilde{x})$ an effective group action (EGA).

In practice, the requirements from the definition of EGA are often to strong. Therefore we will consider the weaker notion of restricted effective group actions which is defined in the full version [18].
Alamati et al. [2] introduced the definition of a weak unpredictable group action. We will use a different notation for that property which is syntactically closer to the prime-order group setting. Note that both definitions are equivalent. In particular, we will use the following assumption.

Definition 5 (Group Action Computational Diffie-Hellman Problem). *On input $(g \star \tilde{x}, h \star \tilde{x})$, the group action computational Diffie-Hellman problem* (GA-CDH) *requires to compute the set element $gh \star \tilde{x}$. To an effective group action* EGA*, we associate the advantage function of an adversary $\mathcal{A}$ as*

$$\mathsf{Adv}^{\mathsf{GA\text{-}CDH}}_{\mathsf{EGA}}(\mathcal{A}) := \Pr[\mathcal{A}(g \star \tilde{x}, h \star \tilde{x}) \Rightarrow gh \star \tilde{x}] \ ,$$

where $g, h \xleftarrow{\$} \mathcal{G}$.

The most promising post-quantum secure instantiation of REGAs is provided by CSIDH. We recall its properties in the full version [18].

2.4 QROM Preliminaries

We use different well-known results from post-quantum cryptography. Specifically, our proofs use the oneway-to-hiding [37] (O2H) lemma from [5] and its measure-rewind-measure (MRM) variant from [30] as well as the online extractable quantum random oracle framework from [17]. We recall the MRM O2H lemma below. Further definitions as well as some basic techniques such as random oracle simulation can be found in the full version [18].

Lemma 1 (Measure-Rewind-Measure O2H. Lemma 3.3 in [30]). *Let* $\mathsf{G}, \mathsf{H}\colon \mathcal{X} \to \mathcal{Y}$ *be random functions,* z *be a random value, and* $\mathcal{S} \subseteq \mathcal{X}$ *be a random set such that* $\mathsf{G}(x) = \mathsf{H}(x)$ *for every* $x \notin \mathcal{S}$*. The tuple* $(\mathsf{G}, \mathsf{H}, \mathcal{S}, z)$ *may have arbitrary joint distribution. Furthermore, let* $\mathcal{A}^{\mathrm{O}}$ *be a unitary/reversible quantum oracle algorithm which queries oracle* O *with query depth* d*. Then we can construct an algorithm* $\mathsf{Ext}^{\mathsf{G},\mathsf{H}}(z)$ *such that the running time of* Ext *is about at most three times the one of* $\mathcal{A}^{\mathrm{O}}$ *and*

$$\left|\Pr_{\mathsf{H},z}[\mathcal{A}^{\mathsf{H}}(z) \Rightarrow 1] - \Pr_{\mathsf{G},z}[\mathcal{A}^{\mathsf{G}}(z) \Rightarrow 1]\right| \leq 4d \Pr_{\mathsf{G},\mathsf{H},\mathcal{S},z}[\mathcal{S} \cap \mathcal{T} \neq \varnothing\colon \mathcal{T} \leftarrow \mathsf{Ext}^{\mathsf{G},\mathsf{H}}(z)]\,.$$

Some of our proofs rely on the hardness of the Generic Distinguishing Problem (GDP), a decisional variant of the Generic Search Problem (GSP) [39,27,25]. Intuitively, an adversary gets oracle access to a function from some domain $\mathcal{D}$ into $\{0,1\}$, which is either the all-zero function or a function where the probability that any given point maps to 1 is small (i.e. bounded by some $\lambda \in (0,1)$), and has to decide which is the case. While the complexity of this problem is clear in the classical case, it is somewhat more difficult in the quantum case. We recall and adapt the well-known bounds to the GDP problem in this section.

Lemma 2 (Generic Distinguishing Problem, decision version of Lemma 2 in [5], Lemma 2.9 from [25]). *Let* $\mathsf{F}\colon \mathcal{X} \to \{0,1\}$ *be a random function drawn from a distribution such that* $\Pr[\mathsf{F}(x) = 1] \leq \lambda$ *for all* x *and* $\mathsf{K}\colon \mathcal{X} \to \{0\}$ *be the zero-function. Let* $\mathcal{A}$ *be a* q*-query algorithm with query depth* d *with quantum-access to its oracle. Then*

$$\mathsf{Adv}^{\mathsf{GDP}}_{\mathsf{F},q,d}(\mathcal{A}) \coloneqq \left|\Pr[\mathsf{GDP}^{\mathcal{A}}_{\mathsf{F},0} \Rightarrow 1] - \Pr[\mathsf{GDP}^{\mathcal{A}}_{\mathsf{F},1} \Rightarrow 1]\right| \leq 4\sqrt{(d+1)q\lambda}\,, \quad (1)$$

where $\mathsf{GDP}^{\mathcal{A}}_{\mathsf{F},0} \coloneqq \mathcal{A}^{\mathsf{K}}()$ *and* $\mathsf{GDP}^{\mathcal{A}}_{\mathsf{F},1} \coloneqq \mathcal{A}^{\mathsf{F}}()$*. Moreover, if the outputs of* F *are independent we have*

$$\mathsf{Adv}^{\mathsf{GDP}}_{\mathsf{F},q,d}(\mathcal{A}) \leq 8(q+1)^2\lambda\,. \quad (2)$$

We prove eq. (1) in the full version [18]. The bound in eq. (2) is a reformulation from Lemma 2.9 from [25].

3 Necessary Assumptions for Group Action KEM and NIKE in the QROM

In this section we will first recall the two schemes we are looking at: Group Action Hashed ElGamal and the Group Action Hashed Diffie-Hellman NIKE

$\underline{\mathsf{Gen}}$	$\underline{\mathsf{Encaps}(\mathsf{pk})}$	$\underline{\mathsf{Decaps}(\mathsf{sk}, \mathsf{ct})}$
00 $\mathsf{sk} \coloneqq g \xleftarrow{\$} \mathcal{G}$	03 $r \xleftarrow{\$} \mathcal{G}$	07 $z \coloneqq \mathsf{sk} \star \mathsf{ct}$
01 $\mathsf{pk} \coloneqq g \star \tilde{x}$	04 $\mathsf{ct} \coloneqq r \star \tilde{x}$	08 $K \coloneqq \mathsf{H}(\mathsf{ct}, z)$
02 **return** $(\mathsf{pk}, \mathsf{sk})$	05 $K \coloneqq \mathsf{H}(\mathsf{ct}, r \star \mathsf{pk})$	09 **return** K
	06 **return** (ct, K)	

Fig. 3. Key encapsulation mechanism GA-HEG for an effective group action $\mathsf{EGA} = (\mathcal{G}, \mathcal{X}, \star, \tilde{x})$, where $\mathsf{H} : \mathcal{X} \times \mathcal{X} \to \{0,1\}^\kappa$ is a hash function.

scheme. We denote the schemes by GA-HEG and GA-HDH, respectively.

Group Action Hashed ElGamal. The scheme is given in Figure 3. Note that this is the same scheme as the CSIDH-ECIES-KEM considered in [38]. The public parameters consist of an effective group action $\mathsf{EGA} = (\mathcal{G}, \mathcal{X}, \star, \tilde{x})$ and a hash function $\mathsf{H} : \mathcal{X}^2 \to \{0,1\}^\kappa$. Further we set $\mathcal{PK} = \mathcal{X}$, $\mathcal{SK} = \mathcal{G}$ and $\mathcal{K} = \{0,1\}^\kappa$. The key generation algorithm samples a random group element $g \xleftarrow{\$} \mathcal{G}$ as secret key. In order to compute the public key, g is applied to the origin element $\tilde{x}$ using the group action operation. The set element $\mathsf{pk} = g \star \tilde{x}$ is the public key. The encapsulation algorithm also first samples a random group element $r \xleftarrow{\$} \mathcal{G}$ and then calculates the ciphertext $\mathsf{ct} = r \star \tilde{x}$. The key is derived by first computing $r \star \mathsf{pk}$ (the shared DH value) and subsequently hashing $r \star \mathsf{pk}$ together with the ciphertext ct. Decapsulation first recomputes the shared DH value $g \star \mathsf{ct} = r \star \mathsf{pk}$ and then applies the hash function H. Correctness of the scheme holds due to the commutativity of the group action.

Group Action Hashed Diffie-Hellman. A schematic overview of the hashed Diffie-Hellman NIKE scheme GA-HDH is given in Figure 4. As in the hashed ElGamal scheme, the public parameters pp include the description of EGA together with a hash function $\mathsf{H} : \{0,1\}^* \to \{0,1\}^\kappa$ such that $\mathcal{PK} = \mathcal{X}$, $\mathcal{SK} = \mathcal{G}$ and $\mathcal{SHK} = \{0,1\}^\kappa$. We assume that $\mathcal{ID} = \{0,1\}^\mu$, which means that each identity is represented by a bitstring of length μ and there is a natural ordering $<$ on the space of identities. On input an $\mathsf{ID} \in \mathcal{ID}$, the key generation algorithm chooses a group element $g \xleftarrow{\$} \mathcal{G}$ which will be the secret key $\mathsf{sk}_{\mathsf{ID}}$. The public key is computed as $\mathsf{pk}_{\mathsf{ID}} = g \star \tilde{x} \in \mathcal{X}$. The shared key of an identity ID_1 with public key $\mathsf{pk}_{\mathsf{ID}_1} = x$ and an identity $\mathsf{ID}_2 \neq \mathsf{ID}_1$ with secret key $\mathsf{sk}_{\mathsf{ID}_2} = g$ is defined as

$$K = \begin{cases} \mathsf{H}(\mathsf{ID}_1, \mathsf{ID}_2, \mathsf{pk}_{\mathsf{ID}_1}, \mathsf{pk}_{\mathsf{ID}_2}, g \star x) & \textbf{if } \mathsf{ID}_1 < \mathsf{ID}_2 \\ \mathsf{H}(\mathsf{ID}_2, \mathsf{ID}_1, \mathsf{pk}_{\mathsf{ID}_2}, \mathsf{pk}_{\mathsf{ID}_1}, g \star x) & \textbf{if } \mathsf{ID}_2 < \mathsf{ID}_1 \end{cases}.$$

Correctness again holds because of the commutativity of the group action itself and the ordering of IDs.

One of the goals of this work is to prove these schemes secure in the QROM (cf. Section 4). However, as it turns out, we will need stronger assumptions for the proofs than those defined in the literature. In the next section we introduce the corresponding assumptions. Furthermore, we show that a (somewhat) stronger

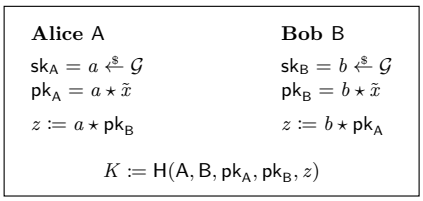

Fig. 4. Group Action Non-Interactive Key Exchange scheme GA-HDH for an effective group action $\mathsf{EGA} = (\mathcal{G}, \mathcal{X}, \star, \tilde{x})$, where $\mathsf{H} : \{0,1\}^* \to \{0,1\}^\kappa$ is a hash function.

assumption is indeed necessary by showing that it is implied by the security of the schemes themselves.

3.1 Computational Group Action Diffie-Hellman with Quantum Oracle Access

Our new assumptions are all variants of the group action strong computational Diffie-Hellman problem (GA-StCDH). The GA-StCDH assumption is basically the translation of the strong CDH problem to group actions (cf. also [28,15]), where the adversary is given access to a (fixed-base) decision oracle. What we need for our proofs is actually *quantum* access to the decision oracle, which is a considerably stronger assumption that was never considered before. For the NIKE proofs, we will also need a double-sided oracle definition, where the adversary gets access to two decision oracles, one for each of the challenge set elements, and its quantum variants. All variants are captured by Definition 6.

Definition 6 (Variants of GA-StCDH). *On input $(g\star\tilde{x}, h\star\tilde{x})$, the* GA-XXX-StCDH *requires to compute the set element $gh \star \tilde{x}$ with access to a decision oracle which is specified below. To an effective group action* EGA *and an adversary $\mathcal{A}$, we associate the advantage function*

$$\mathsf{Adv}^{\mathsf{GA\text{-}XXX\text{-}StCDH}}_{\mathsf{EGA}}(\mathcal{A}) := \Pr[\mathcal{A}^{\mathrm{O}}(g \star \tilde{x}, h \star \tilde{x}) \Rightarrow gh \star \tilde{x}] \ ,$$

where $g, h \xleftarrow{\$} \mathcal{G}$ and

$$\mathrm{O} := \begin{cases} \mathsf{GA\text{-}DDH}_g(\cdot,\cdot)\,, & \mathsf{XXX} = \{\} \quad \textit{(classical)} \\ \mathsf{GA\text{-}DDH}_g(\cdot,|\cdot\rangle)\,, & \mathsf{XXX} = \mathsf{PQ} \quad \textit{(partially quantum)} \\ \mathsf{GA\text{-}DDH}_g(|\cdot\rangle,|\cdot\rangle)\,, & \mathsf{XXX} = \mathsf{FQ} \quad \textit{(fully quantum)} \\ \{\mathsf{GA\text{-}DDH}_g(\cdot,\cdot), \mathsf{GA\text{-}DDH}_h(\cdot,\cdot)\}\,, & \mathsf{XXX} = \mathsf{D} \quad \textit{(double-sided classical)} \\ \{\mathsf{GA\text{-}DDH}_g(\cdot,|\cdot\rangle), \mathsf{GA\text{-}DDH}_h(\cdot,|\cdot\rangle)\}\,, & \mathsf{XXX} = \mathsf{DPQ} \quad \textit{(double-sided partially quantum)} \\ \{\mathsf{GA\text{-}DDH}_g(|\cdot\rangle,|\cdot\rangle), \mathsf{GA\text{-}DDH}_h(|\cdot\rangle,|\cdot\rangle)\}\,, & \mathsf{XXX} = \mathsf{DFQ} \quad \textit{(double-sided fully quantum)} \end{cases}$$

On basis-state inputs (y, z), $\mathsf{GA\text{-}DDH}_g$ *returns 1 if $g \star y = z$ and 0 otherwise.* $\mathsf{GA\text{-}DDH}_h$ *is defined equivalently. Note that superposition queries are implicitly then defined by linearity (i.e., $\mathrm{O}(\sum_x \alpha_x x) = \sum_x \alpha_x \mathrm{O}(x)$). We emphasize that the partially quantum variants of the oracle measure their corresponding first input implicitly.*

3.2 Necessity of the GA-(D)PQ-StCDH Assumption

We now show that partial quantum access to the decision oracle is indeed a *necessary* assumption to prove IND-CCA security of GA-HEG and CKS security of GA-HDH. We do that by showing the opposite direction, namely that the assumption is implied by the security of the corresponding scheme. This is captured by the following two theorems.

Theorem 1. *Let* $\mathsf{H}\colon \mathcal{X} \times \mathcal{X} \to \{0,1\}^\kappa$ *be a random oracle. For any quantum adversary* $\mathcal{A}$ *against* GA-PQ-StCDH *making at most* q *queries to its decision oracle, there exists a quantum adversary* $\mathcal{B}$ *against* IND-CCA *security of* GA-HEG *making at most* q *decapsulation queries and* $q+1$ *quantum random oracle queries with*

$$\mathsf{Adv}^{\mathsf{GA\text{-}PQ\text{-}StCDH}}_{\mathsf{EGA}}(\mathcal{A}) \leq 2 \cdot \mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathsf{GA\text{-}HEG}}(\mathcal{B}) + \frac{8(q+1)^2+1}{2^\kappa}\,,$$

and the running time of $\mathcal{B}$ *is about that of* $\mathcal{A}$*.*

Theorem 2. *Let* $\mathsf{H}\colon \{0,1\}^* \to \{0,1\}^\kappa$ *be a random oracle. For any quantum adversary* $\mathcal{A}$ *against* GA-DPQ-StCDH *making at most* q *queries to its decision oracles, there exists a quantum adversary* $\mathcal{B}$ *against the* CKS *security of* GA-HDH *making* 2 *queries to the* RegisterHonest *oracle, at most* q *queries to the* RegisterCorrupt *oracle and* $q+1$ *quantum random oracle queries with*

$$\mathsf{Adv}^{\mathsf{GA\text{-}DPQ\text{-}StCDH}}_{\mathsf{EGA}}(\mathcal{A}) \leq 2 \cdot \mathsf{Adv}^{\mathsf{CKS}}_{\mathsf{GA\text{-}HDH}}(\mathcal{B}) + \frac{8(q+1)^2+1}{2^\kappa}\,,$$

and the running time of $\mathcal{B}$ *is about that of* $\mathcal{A}$*.*

We will prove Theorem 1 below. The proof of Theorem 2 is very similar and we refer to the full version [18] for more details.

Proof (of Theorem 1). The idea of the proof is to construct a reduction which implements the decision oracle using the decapsulation oracle by testing whether $\mathsf{Decaps}(x_1) = \mathsf{H}(x_1, x_2)$ on a decision oracle query $\mathrm{O}(x_1, x_2)$. Whenever $\mathrm{O}(x_1, x_2)$ returns 1, so will $\mathsf{Decaps}(x_1) = \mathsf{H}(x_1, x_2)$, except when x_1 is the challenge ciphertext. Therefore, whenever x_1 is the challenge ciphertext, the reduction is going to do the same test, except that it first "shifts" x_1 and x_2 by some other group element $\hat{g}$. After simulating all decision oracle queries, the reduction returns whether the challenge KEM key K does not equal $\mathsf{H}(c^*, z)$ where z is the group action CDH solution obtained by $\mathcal{A}$. We now proceed with the formal proof.

Let $\mathcal{A}$ be a quantum adversary as described in Theorem 1. Consider the sequence of games given in Figure 5.

Game G_1. This is the GA-PQ-StCDH game, where $\mathrm{O} = \mathsf{GA\text{-}DDH}_g$. By definition,

$$\Pr[\mathrm{G}_1^{\mathcal{A}} \Rightarrow 1] = \mathsf{Adv}^{\mathsf{GA\text{-}PQ\text{-}StCDH}}_{\mathsf{EGA}}(\mathcal{A})\,.$$

Game G_2. In this game, instead of returning whether $g \star x_1 = x_2$, the decision oracle returns whether $(x_1, g \star x_1) = (x_1, x_2)$. In order to prepare for the next

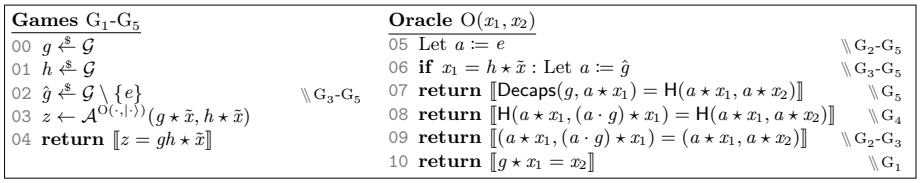

Fig. 5. Games G_1-G_5 for the proof of Theorem 1.

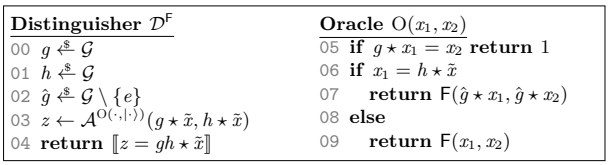

Fig. 6. Distinguisher $\mathcal{D}$ for the Generic Distinguishing Problem to bound G_4-G_5.

game hop, we additionally introduce a new variable a which denotes a group element. In G_2, a is always the neutral element e of $\mathcal{G}$, thus applying a on any set element does not have any effect. Since we always have $x_1 = x_1$, the check in line 09 is the same as in line 10. Hence we have $\Pr[G_1^{\mathcal{A}} \Rightarrow 1] = \Pr[G_2^{\mathcal{A}} \Rightarrow 1]$.

Game G_3. In this game we sample a group element $\hat{g} \xleftarrow{\$} \mathcal{G} \setminus \{e\}$ uniformly at random in line 02. For all queries (x_1, x_2) to O, where $x_1 = h \star \tilde{x}$, we now set a to $\hat{g}$. In this case, this will change the boolean test in line 09. However, since the group action operation is a bijection, this change is only conceptual. The reason for doing this, is that in the final reduction we are going to set $h \star \tilde{x}$ to be the challenge ciphertext c^* which we cannot query to the decapsulation oracle. Shifting by $\hat{g}$ in the case that $x_1 = h \star \tilde{x}$ will allow us to still simulate O. We get $\Pr[G_2^{\mathcal{A}} \Rightarrow 1] = \Pr[G_3^{\mathcal{A}} \Rightarrow 1]$.

Game G_4. In this game we perform the boolean test by first hashing both sides using a random oracle. In particular, we check if $\mathsf{H}(a \star x_1, (a \star g) \star x_1) = \mathsf{H}(a \star x_1, a \star x_2)$ in line 08. This introduces false positives into the decision oracle, when for any $\hat{x}_1 \in \mathcal{X}$ we have that $\mathsf{H}(\hat{x}_1, g \star \hat{x}_1)$ has preimages of the form $(\hat{x}_1, \hat{x}_2)$ with $\hat{x}_2 \neq g \star \hat{x}_1$. We can bound this change by reducing to the GDP problem, which we do in Figure 6. In particular, for every $(\hat{x}_1, \hat{x}_2)$ we have $\mathsf{F}(\hat{x}_1, \hat{x}_2)$ returns 1 with probability $\lambda := 1/2^\kappa$, which is the probability to find a second preimage for $\mathsf{H}(\hat{x}_1, g \star \hat{x}_1)$. If F is the zero function, the distinguisher $\mathcal{D}$ simulates G_3 and otherwise it simulates G_4. Thus by eq. (2) of Lemma 2 where we have set $\lambda := 1/2^\kappa$ we have

$$\begin{aligned} &\left|\Pr[G_3^{\mathcal{A}} \Rightarrow 1] - \Pr[G_4^{\mathcal{A}} \Rightarrow 1]\right| \\ =\ &\left|\Pr[\mathsf{GDP}_{\mathsf{F},0}^{\mathcal{D}} \Rightarrow 1] - \Pr[\mathsf{GDP}_{\mathsf{F},1}^{\mathcal{D}} \Rightarrow 1]\right| \leq 8(q+1)^2/2^\kappa\,. \end{aligned}$$

Game G_5. In this game we change the boolean test again and check whether $\mathsf{Decaps}(g, a \star x_1) = \mathsf{H}(a \star x_1, a \star x_2)$ in line 07. By definition of decapsulation, this change is again only conceptual. We have $\Pr[G_4^{\mathcal{A}} \Rightarrow 1] = \Pr[G_5^{\mathcal{A}} \Rightarrow 1]$.

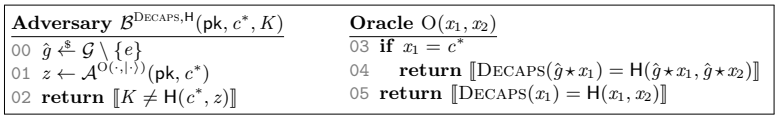

Fig. 7. Adversary $\mathcal{B}$ against IND-CCA security for bounding G_6.

It remains to bound G_5. We claim

$$\Pr[\mathrm{G}_5^{\mathcal{A}} \Rightarrow 1] \leq 2 \cdot \mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathsf{GA\text{-}HEG}}(\mathcal{B}) + 1/2^{\kappa}\,. \tag{3}$$

The adversary $\mathcal{B}$ in Figure 7 simulates G_5 as follows: it runs $\mathcal{A}$ on its own inputs (pk, c^*), thus defining $g \star \tilde{x} := \mathsf{pk}$ and $h \star \tilde{x} := c^*$. Note that it can simulate oracle O as in G_5 using its own DECAPS oracle and random oracle H provided by the IND-CCA challenger. If $\mathcal{A}$ queries O on the challenge ciphertext c^*, we make use of the additional element $\hat{g}$, thus $\mathcal{B}$ never queries DECAPS on the challenge ciphertext. Finally $\mathcal{A}$ outputs z. If $\mathsf{H}(c^*, z) = K^*$, where K^* is the challenge key $\mathcal{B}$ received at the beginning, it returns 0 (real), otherwise it returns $b' := 1$ (random). Clearly, if $\mathcal{A}$ computes z as $gh \star \tilde{x}$, $\mathcal{B}$ always wins the IND-CCA game when it is in the real world. In the random world, it will win only with probability $1-1/2^{\kappa}$ since the challenge key might be the same as the real key with probability $1/2^{\kappa}$. When z is not the correct solution and K is the real key, then $\mathcal{B}$ will only win if the output of H still coincides with K, i.e. with probability $1/2^{\kappa}$. However, if K is a random key, $\mathcal{B}$ will win again with probability $1 - 1/2^{\kappa}$. Collecting the conditional probabilities yields the bound claimed in eq. (3).

It remains to analyze the running time of $\mathcal{B}$ and its additional oracle calls. $\mathcal{B}$ runs $\mathcal{A}$ once and for every query to O, $\mathcal{B}$ makes one call to the decapsulation oracle and random oracle. After running $\mathcal{A}$ it makes one additional call to the random oracle, which yields the claimed number of additional oracle calls, which concludes our proof. □

Remark 2. Quantum-secure signatures and public-key encryption schemes have been studied in [10], where the adversary gets quantum access to the signing and decryption oracle, respectively. One can show that the *Quantum* IND-CCA (IND-qCCA) security of GA-HEG is equivalent to the GA-FQ-StCDH assumption, that is the assumption is necessary and sufficient. The proof that IND-qCCA implies the GA-FQ-StCDH assumption is the same as the proof of Theorem 1. Therefore, observe that since the first input of the decision oracle is not measured, the reduction needs a quantum-accessible decapsulation oracle, which is provided by the IND-qCCA game. The sufficiency follows by observing that the reduction in the proof of Theorem 3 can actually simulate quantum decapsulation queries. We leave it as an open problem whether the GA-PQ-StCDH assumption is sufficient for IND-CCA security GA-HEG.

4 Security of Group Action Hashed ElGamal and NIKE

We now prove security of the two schemes in the quantum random oracle model. In particular, we prove IND-CCA security of GA-HEG under the GA-FQ-StCDH assumption and CKS security of GA-HDH under the GA-DFQ-StCDH assumption, i.e., with full quantum access to the decision oracle.

Due to our results in Section 3.2, we cannot hope to prove security of the (un-modified) schemes based on assumptions without quantum access. However, adding key confirmation to GA-HEG allows us to do so. We elaborate in more detail in Section 4.2. Unfortunately, key confirmation cannot be applied in the context of non-interactive schemes such as GA-HDH.

4.1 Security of GA-HEG

The following theorem states security of GA-HEG based on the GA-FQ-StCDH assumption. For the proof we will use the MRM O2H lemma (Lemma 1).

Remark 3. Alternatively, we could use the O2H variant of [8] (also for proving $\mathsf{GA\text{-}Twin\text{-}HEG}_m$) by using its extractor in the proof, yielding a bound of $\sqrt{\mathsf{Adv}}$. Since both versions are applicable, one can essentially choose between a quadratic loss independent of the adversary's query depth or a linear loss in the query depth. To keep proofs and theorems simple, we only prove the bound using MRM.

Theorem 3. *For any quantum adversary $\mathcal{A}$ against* IND-CCA *security of* GA-HEG *that issues at most q queries to the quantum-accessible random oracle* H *of query depth d with query parallelism $p := q/d$, there exists an adversary $\mathcal{B}$ against* GA-FQ-StCDH *such that*

$$\mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathsf{GA\text{-}HEG}}(\mathcal{A}) \leq 4d\mathsf{Adv}^{\mathsf{GA\text{-}FQ\text{-}StCDH}}_{\mathsf{EGA}}(\mathcal{B})\,,$$

and the running time of $\mathcal{B}$ is about three times that of $\mathcal{A}$ plus at most $\mathcal{O}(q+p)$ queries to the decision oracle and the time to simulate up to $\mathcal{O}(\max\{q_D, q\})$ random oracle queries, where q_D is the number of decapsulation queries.

Proof. Let $\mathcal{A}$ be a quantum adversary as described in Theorem 3. Consider the games given in Figure 8. We proceed by analyzing the different games.

Game G_1. This is the IND-CCA game where we unfolded the definition of GA-HEG. By definition,

$$\left|\Pr[\mathrm{G}_1^{\mathcal{A}} \Rightarrow 1] - 1/2\right| = \mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathsf{GA\text{-}HEG}}(\mathcal{A})\,.$$

Game G_2. Here we introduce the following conceptual change: the random oracle H is simulated using two *internal* random oracles H_1 and H_2, where the first one is used on valid DH tuples, and the second on invalid ones. For this change to be meaningful (i.e., simulatable) later on, we need a quantum-accessible decision

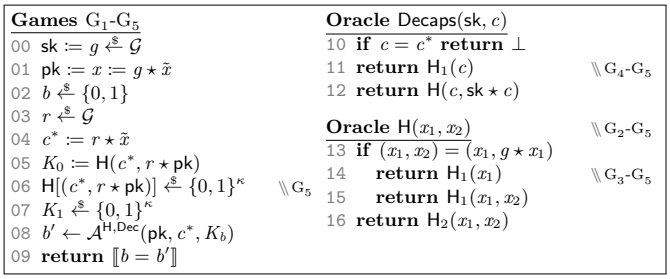

Fig. 8. Games G_1-G_5 for the proof of Theorem 3, where H_1 and H_2 are internal random oracles.

oracle, which is provided by the GA-FQ-StCDH assumption. Clearly, the change is only conceptual and we have $\Pr[G_1^{\mathcal{A}} \Rightarrow 1] = \Pr[G_2^{\mathcal{A}} \Rightarrow 1]$.

Game G_3. Next, we drop the input x_2 in the case where the random oracle H_1 is used, that is we return $\mathsf{H}_1(x_1)$ instead of $\mathsf{H}_1(x_1, x_2)$. Since relative to pk and x_1 there exists a *unique* x_2 s.t. $(x_1, x_2) = (x_1, g \star x_1)$, due to the regularity property of EGA, this change is again only conceptual and we have $\Pr[G_2^{\mathcal{A}} \Rightarrow 1] = \Pr[G_3^{\mathcal{A}} \Rightarrow 1]$.

Game G_4. In this game we remove the usage of the secret key in the random oracle calls of the decapsulation oracle by returning $\mathsf{H}_1(c)$ instead of $\mathsf{H}(c, g \star c)$. Note that the secret key is only used to check for the DDH condition, which can be simulated with access to $\mathsf{GA\text{-}DDH}_g(|\cdot\rangle, |\cdot\rangle)$. Due to the previous conceptual change $\mathsf{H}_1(c) = \mathsf{H}(c, g \star c)$ holds by definition and therefore this change is again only conceptual, thus $\Pr[G_3^{\mathcal{A}} \Rightarrow 1] = \Pr[G_4^{\mathcal{A}} \Rightarrow 1]$.

Game G_5. In this game we reprogram the random oracle on the challenge input $(c^*, r \star \mathsf{pk})$, after querying $\mathsf{H}(c^*, r \star \mathsf{pk})$ in line 06. Now K_0 is identicially distributed as K_1, therefore the key is now independent of the challenge bit b and we have $\Pr[G_5^{\mathcal{A}} \Rightarrow 1] = 1/2$. Due to Lemma 1 (MRM-O2H) we have

$$\left|\Pr[G_4^{\mathcal{A}} \Rightarrow 1] - \Pr[G_5^{\mathcal{A}} \Rightarrow 1]\right| \leq 4d \Pr[G_6^{\mathsf{Ext}} \Rightarrow 1]\,,$$

where G_6^{Ext} is like $G_4^{\mathcal{A}}$, except that instead of running $\mathcal{A}$, it runs the extraction algorithm $\mathsf{Ext}^{\mathsf{Decaps},\mathsf{H},\mathsf{H}'}$ from the MRM-O2H lemma to obtain a set $\mathcal{T}$ and the winning condition is changed to $[\![\mathcal{S} \cap \mathcal{T} \neq \varnothing]\!]$, where $\mathcal{S} \coloneqq \{(c^*, r \star \mathsf{pk})\}$ and H' is the reprogrammed random oracle.

We bound the right-hand probability by the adversary $\mathcal{B}$ given in Figure 9, which runs the extraction algorithm simulating Decaps and H as in G_4 and H' (the reprogrammed H) as in G_5. Observe that $\mathcal{B}$ can simulate quantum decapsulation queries, since it has quantum access to H_1, which is why we can apply the MRM-O2H lemma. Since $\mathcal{B}$ wins if $\mathcal{S} \cap \mathcal{T} \neq \varnothing$, we have

$$\Pr[G_6^{\mathsf{Ext}} \Rightarrow 1] \leq \mathsf{Adv}_{\mathsf{EGA}}^{\mathsf{GA\text{-}FQ\text{-}StCDH}}(\mathcal{B})\,.$$

Combining all inequalities yields the claimed bound. We conclude our proof by analyzing the running time of $\mathcal{B}$. $\mathcal{B}$ runs the extraction algorithm Ext, whose

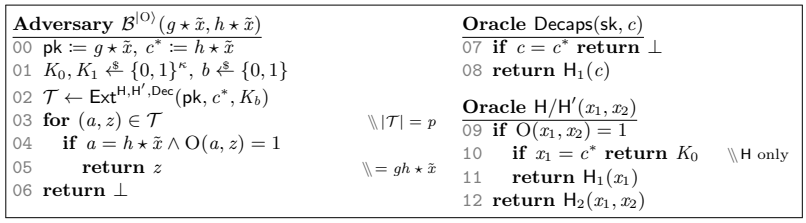

Fig. 9. Adversary $\mathcal{B}$ for the game-hop G_4-G_5 for the proof of Theorem 3. H_1 and H_2 are internal random oracles. The oracle O is the $\mathsf{GA\text{-}DDH}_g$ oracle.

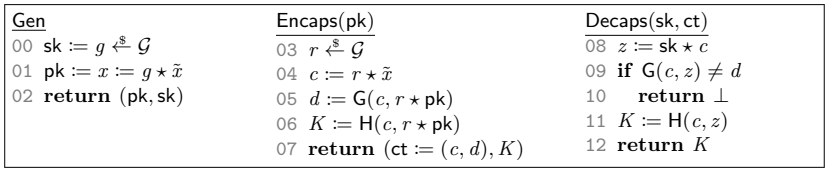

Fig. 10. Key encapsulation mechanism GA-HEG-KC for an effective group action $\mathsf{EGA} = (\mathcal{G}, \mathcal{X}, \star, \tilde{x})$, where $\mathsf{G} : \mathcal{X} \times \mathcal{X} \to \{0,1\}^n$ and $\mathsf{H} : \mathcal{X} \times \mathcal{X} \to \{0,1\}^\kappa$ are hash functions.

running time is at most three times that of $\mathcal{A}$. For every run of $\mathcal{A}$, it has to simulate at most $\max\{q_D, q\}$ calls to H_1 and q calls to H_2 (through H, H'), where it calls O on every query. Then, after obtaining $\mathcal{T}$, it makes at most p queries to O, thus $q + p$ total queries to O. Multiplying the parts of simulating $\mathcal{A}$ by 3, adding up and applying $\mathcal{O}$ notation yields the claimed running time and additional oracle calls, which concludes our proof. □

4.2 Security of GA-HEG via Key Confirmation

We recall the Hashed ElGamal scheme with key confirmation in Figure 10. We denote this scheme by GA-HEG-KC. Compared to the original scheme in Figure 3, we now have a second hash function $\mathsf{G}\colon \mathcal{X} \times \mathcal{X} \to \{0,1\}^n$ which is used to compute an additional ciphertext element d. The input to this hash function is the same as for the final key. The decapsulation algorithm now first checks if d is valid by recomputing it. If this check passes, the actual key is computed and returned, otherwise the algorithm outputs a failure symbol $\bot$.

Theorem 4 establishes security of GA-HEG-KC based on the GA-StCDH assumption, that is without quantum access to the decision oracle. One reason for the looser bound is that the classical decision oracle does not enable us to apply the more recent O2H lemmata. The other is that we have to first apply O2H, before applying the extractable RO simulator.

Theorem 4. *Let* $\mathsf{G}\colon \mathcal{X} \times \mathcal{X} \to \{0,1\}^n$ *be a random oracle. For any quantum adversary* $\mathcal{A}$ *against* IND-CCA *security of* GA-HEG-KC *that issues at most* d *parallel queries each of size* p *(in total* $q := dp$ *queries) to the quantum-accessible*

random oracles H *and* G *and* q_D *decapsulation queries, there exists an adversary* $\mathcal{B}$ *against the* GA-StCDH *such that*

$$\mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathsf{GA\text{-}HEG\text{-}KC}}(\mathcal{A}) \leq 2d\sqrt{\mathsf{Adv}^{\mathsf{GA\text{-}StCDH}}_{\mathsf{EGA}}(\mathcal{B})} + \frac{8(q+1)^2}{2^n} + \sqrt{\frac{32q_D(q_D+q)}{\sqrt{2^n}}} + \sqrt{\frac{4q_D}{2^n}} + \sqrt{\frac{40e^2(q+2q_D+2)^3}{2^n}},$$

and the running time of $\mathcal{B}$ *is about that of* $\mathcal{A}$ *plus the running time for using extractable random-oracle simulator for* q_D *extraction queries and* q *hash queries, which is about* $\mathcal{O}(q \cdot q_D + q^2)$ *and simulating* H *for* q *queries, additionally* $\mathcal{B}$ *makes at most* $q_D + p$ *queries to its decision oracle.*

Note that n depends on the desired security level. Due to the fourth root term, n needs to be around four times the security parameter in bits. We discuss this in more detail in Section 6. We will now sketch the proof of Theorem 4. The full proof can be found in the full version [18].

Proof (Sketch). After some simple changes we first reprogram the random oracle H and G on the challenge inputs using O2H. Then the main idea of the proof is to simulate the random oracle G using the extractable random-oracle simulator. The reduction can then simulate decapsulation queries by extracting the inputs from the key-confirmation hash and verify the validity using the decision oracle $\mathsf{GA\text{-}DDH}(g \star \tilde{x}, \cdot, \cdot)$. Note that since the decapsulation oracle is classical, the extracted values are also classical and we only need classical access to $\mathsf{GA\text{-}DDH}(g \star \tilde{x}, \cdot, \cdot)$. Once we can simulate decapsulation without the secret key using the classical decision oracle, we can reduce the game to the GA-StCDH problem. □

4.3 Security of GA-HDH

The following theorem establishes security of GA-HDH based on the GA-DFQ-StCDH assumption. As opposed to the proof of GA-HEG, we have to use the semi-classical variant of the O2H lemma which yields a worse bound. We explain the reason in the full version [18].

Theorem 5. *For any quantum adversary* $\mathcal{A}$ *against the* CKS *security of* GA-HDH *that issues at most* d *parallel queries, each of size* p*, to the quantum-accessible random oracle* H*, there exists an adversary* $\mathcal{B}$ *against* GA-DFQ-StCDH *such that*

$$\mathsf{Adv}^{\mathsf{CKS}}_{\mathsf{GA\text{-}HDH}}(\mathcal{A}) \leq \sqrt{8(d+1)\mathsf{Adv}^{\mathsf{GA\text{-}DFQ\text{-}StCDH}}_{\mathsf{EGA}}(\mathcal{B})},$$

and the running time of $\mathcal{B}$ *is about three times that of* $\mathcal{A}$ *plus* $\mathcal{O}(q+p)$ *queries to the decision oracle and the running time for simulating* $\mathcal{O}(\max\{d \cdot p, q_R, q_T\})$ *queries to the random oracle and* $\mathcal{O}(q_O)$ *rerandomizations on the set elements, where* q_O*,* q_R *and* q_T *are the number of register-honest, reveal and test queries.*

We will only sketch the proof here. The full proof can be found in the full version [18].

Proof (Sketch). As in the proof of Theorem 3, our goal is to use a variant of the O2H lemma in order to randomize all challenge keys and bound the advantage of the O2H extractor using the GA-DFQ-StCDH assumption. However, instead of just a decapsulation oracle, we have to simulate the CorruptReveal oracle and the Test oracle. Although the adversary is allowed to choose identities for honest keys, we can compute all honest keys before the adversary can make any queries, so we can vary the behavior of the random oracle when it interacts with honest or corrupted keys. Note that this technique is not generally possible as the key generation could depend on the provided ID in other schemes. This allows to only hash $(\mathsf{ID}_1, \mathsf{ID}_1, \mathsf{pk}_1, \mathsf{pk}_2)$ without the shared DH value between pk_1 and pk_2, when at least one key is honest. Additionally, we can use a different internal random oracle, when both keys are honest. In the final reduction on GA-DFQ-StCDH, we embed the challenge set elements into the public keys using rerandomization. For each public key, we randomly choose which challenge element we use such that the adversary will issue a test query at least for one pair of identities containing both challenge elements. We can check whether quantum random oracle queries contain valid DH tuples using quantum access to the decision oracles. Then we can use the O2H lemma in its semi-classical variant and bound the success probability of its extractor with the GA-DFQ-StCDH assumption. □

5 Twinning for Group Actions

In this section, we adapt the twinning technique from [11] to the group actions setting. Due to the limited structure that group actions offer, we need a novel approach to develop and analyze the underlying trapdoor test. The trapdoor test will allow us to effectively simulate a decision oracle, apart from a small error probability. In contrast to the original twinning approach, the analysis of the error term is more involved and depends on an additional parameter m, which affects the "twinning factor". To illustrate this in an example: whereas in the traditional prime-order group setting, twinning doubles the size of public keys, the group action twinning technique will result in a public key of length m.

Using this technique we get two new schemes GA-Twin-HEG$_m$ and GA-Twin-HDH$_m$, the twinned versions of GA-HEG and GA-HDH, which will be presented and analyzed in Sections 5.2 and 5.3. It allows us to remove the strong variants of GA-CDH including quantum access to decision oracles in the security proofs. Consequently we obtain a proof based on the standard GA-CDH assumption, albeit in exchange for larger keys and overall increased computation cost. Nevertheless, using our new twinning technique is thus far the only known method that allows for a security proof of a NIKE scheme from standard assumptions in the QROM. In Section 6 we discuss different parameter choices for m.

5.1 A Trapdoor Test

In order to replace the GA-(FQ-)StCDH assumption, an algorithm must be able to simulate the decision oracle $\mathsf{GA\text{-}DDH}_g$ without knowing g explicitly. The following trapdoor test will be our basic tool to achieve this task.

Lemma 3 (Trapdoor Test). *Let* $\mathsf{EGA} = (\mathcal{G}, \mathcal{X}, \star, \tilde{x})$, $\ell, m \in \mathbb{N}$ *such that* $1 < \ell < m/2$. *Suppose* $x_0, x_1, ..., x_{\ell-1}$, $s_\ell, ..., s_m$, $h_\ell, ..., h_m$ *are mutually independent random variables, where* $x_0, x_1, ..., x_{\ell-1}$ *take values in* $\mathcal{X}$, *and for all* $i \in [\ell, m]$ s_i *are uniformly distributed over* $[0, \ell-1]$ *with the additional condition that each value in* $[0, \ell-1]$ *is taken at least once. Further, for all* $i \in [\ell, m]$ h_i *are uniformly distributed over* $\mathcal{G}$. *Define random variables* $x_\ell, ..., x_m$, *where* $x_i = h_i \star x_{s_i}$ *for* $i \in [\ell, m]$. *Further, let* $g_i \in \mathcal{G}$ *such that* $x_i = g_i \star \tilde{x}$ *for every* $i \in [m]$. *In addition, suppose that* $\bar{z}_0, \bar{z}_1, ..., \bar{z}_m$ *are random variables taking values in* $\mathcal{X}$.

We define

$$F_0(\bar{z}_0, \ldots, \bar{z}_m) \coloneqq \begin{cases} 1 & \text{if } \bar{z}_i = h_i \star \bar{z}_{s_i} \quad \forall i \in [\ell, m] \\ 0 & \text{else} \end{cases} \tag{4}$$

and

$$F_1(\bar{z}_0, \ldots, \bar{z}_m) \coloneqq \begin{cases} 1 & \text{if } \bar{z}_i = g_i \star \bar{z}_0 \quad \forall i \in [m] \\ 0 & \text{else} \end{cases} \tag{5}$$

and the advantage of an adversary $\mathcal{A}$ *in distinguishing* F_0 *from* F_1 *with oracle access to one of the two functions and making at most* q *queries of depth* d *as*

$$\mathsf{Adv}^{\mathsf{TDT}}_{\mathsf{EGA},q,d,\ell,m}(\mathcal{A}) \coloneqq \left|\Pr[\mathcal{A}^{F_0} \Rightarrow 1] - \Pr[\mathcal{A}^{F_1} \Rightarrow 1]\right|$$

We call eq. (4) *the* Trapdoor Test. *The following properties hold:*

1. $x_\ell, ..., x_m$ *are uniformly distributed over* $\mathcal{X}$;
2. x_i *and* x_j *are independent for all* $i \in [0, \ell-1], j \in [\ell, m]$;
3. *if* $F_1(z) = 1$, *then also* $F_0(z) = 1$ *for any input vector* $\vec{z}$;
4. *for any classical (quantum) adversary* $\mathcal{A}$ *with oracle access to* F_b *for* $b \in \{0,1\}$, *the probability that* $\mathcal{A}$ *outputs 1 after at most* q *queries to* F_b *with query depth* d *is upper-bounded by the advantage of a classical (quantum) adversary* $\mathcal{B}$ *against the* GDP *problem for a function* $T : \mathcal{Y} \to \{0,1\}$ *with* $\Pr[T(x) = 1 \colon x \xleftarrow{\$} \mathcal{Y}] \leq \frac{1}{|\mathcal{Y}|}$ *and* $|\mathcal{Y}| = \ell!\ell^{m-2\ell+1}$ *(see Remark 4). Specifically,*

$$\mathsf{Adv}^{\mathsf{TDT}}_{\mathsf{EGA},q,d,\ell,m}(\mathcal{A}) \leq \mathsf{Adv}^{\mathsf{GDP}}_{T,q,d}(\mathcal{B}) \leq \begin{cases} \frac{2q}{|\mathcal{Y}|} & \textit{(classical)} \\ 4\sqrt{\frac{(d+1)q}{|\mathcal{Y}|}} & \textit{(quantum)} \end{cases}.$$

Proof. Properties 1. to 3. hold by inspection. For property 4., we build an adversary $\mathcal{B}$ on the GDP problem from a successful distinguisher $\mathcal{A}$ of the trapdoor test. The proofs are identical for the classical and quantum case as the oracles that $\mathcal{B}$ has to implement can all be defined as classical functions which make

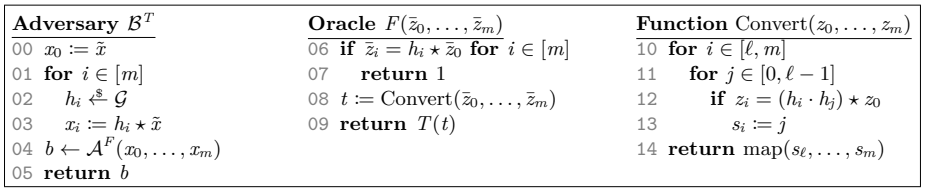

Fig. 11. Adversary $\mathcal{B}^{|T\rangle}$ against the GDP problem for the function T. The function "map" is the selected bijection from the set of possible s_i into $\mathcal{Y}$.

classical queries to other oracles, so by making all oracles quantum, the proof does not change.

First note that if $\mathcal{A}$ only queries tuples $z_0, \ldots, z_m$ to its function F_b for which x_i, z_0, z_i form a DH tuple, then both oracles always behave identically, so we assume that it will not make such queries. Since the s_i take all values in $[0, \ell - 1]$, for non-DH queries, the oracles differ only if $\mathcal{A}$ guesses *all* s_i used to generate the x_i correctly. In that case it could choose the first ℓ elements at random and set the last $m - \ell + 1$ elements to $g_i \star x_{s_i}$, where the g_i are the discrete logarithms of the i-th randomly chosen element. If the s_i do not cover all values in $[0, \ell - 1]$, this argument does not hold (see Remark 5).

We will construct an adversary $\mathcal{B}$ on the GDP problem for a function T, which will simulate the function F_1 if T is the all-zero function and F_0, i.e. the trapdoor test, if not. Specifically, let $T : \mathcal{Y} \to \{0, 1\}$ such that there is a bijective mapping from $\mathcal{Y}$ into the set of all possible combinations of s_i.

We describe $\mathcal{B}$ in Figure 11. First, $\mathcal{B}$ sets x_0 to the origin element $\tilde{x}$ and chooses m random elements $x_1, \ldots x_m$ and runs $\mathcal{A}$ on them as input. When $\mathcal{A}$ makes a query to F, $\mathcal{B}$ first checks if $\mathcal{A}$ provided a valid DH tuple and if so, returns 1. Otherwise, it computes which s_i were (implicitly) chosen to generate the query and maps them to the unique element they correspond to in $\mathcal{Y}$. Then it queries this element to its own function T and returns the result.

If T is the all-zero function, then F only returns 1 if the first check succeeds, i.e., F is equal to F_1 from eq. (5). Otherwise, there is exactly one entry in T for which it returns 1. Therefore, by returning the result of the query to T, $\mathcal{B}$ implicitly chooses its s_i as the ones corresponding to said entry in T and therefore simulates F_0 from eq. (4). So by outputting the same result as $\mathcal{A}$, $\mathcal{B}$ wins if and only if $\mathcal{A}$ wins and the claim follows. The quantum bound then follows directly from Lemma 2. □

Remark 4 (Sampling s_i). Let $\ell, m \in \mathbb{N}$ as in Lemma 3 and $k = m - \ell + 1$. Define

$$\mathcal{Y}^* = \{(s_\ell, \ldots, s_m) \in [0, \ell - 1]^k \mid \forall i \in [0, \ell - 1] \; \exists j : s_j = i\}.$$

In principal this is the set of possible values for the $(s_\ell, \ldots, s_m)$ from the lemma. The cardinality of $\mathcal{Y}^*$ may be described by the *Stirling partition number* multi-

Gen	Encaps(pk)	Decaps(sk, ct)
00 $\mathsf{sk} := (h_1, ..., h_m) \xleftarrow{\$} \mathcal{G}^m$	03 $r \xleftarrow{\$} \mathcal{G}$	07 $K := \mathsf{H}(\mathsf{ct}, h_1 \star \mathsf{ct}, ..., h_m \star \mathsf{ct})$
01 $\mathsf{pk} := (y_1, ..., y_m) := (h_1 \star \bar{x}, ...h_m \star \bar{x})$	04 $\mathsf{ct} := r \star \bar{x}$	08 **return** K
02 **return** $(\mathsf{pk}, \mathsf{sk})$	05 $K := \mathsf{H}(\mathsf{ct}, r \star y_1, ..., r \star y_m)$	
	06 **return** (ct, K)	

Fig. 12. Twin Hashed ElGamal KEM GA-Twin-HEG$_m$ with twinning parameter m. $\mathsf{H} : \mathcal{X}^{m+1} \to \{0,1\}^\kappa$ is a hash function.

plied by $\ell!$, more precisely

$$|\mathcal{Y}^*| = \ell! \cdot \left\{ {k \atop \ell} \right\} = \sum_{i=0}^{d} (-1)^i \binom{\ell}{i} (\ell - i)^k.$$

One possibility to sample randomly from the entire set $\mathcal{Y}^*$ is rejection sampling from $[0, \ell-1]^k$. Since this is not very practical, we suggest the following sampling method which samples from the strictly smaller subset $\mathcal{Y}$ of size $\ell!\ell^{k-\ell}$.

In order to ensure that the s_i take each value in $[0, \ell - 1]$, we first sample exactly these ℓ elements and then sample the remaining $k-\ell$ elements uniformly at random from $[0, \ell - 1]$.

Remark 5 (Necessity of the condition on s_i). The assumption that each value in $[0, \ell-1]$ is taken at least once by the s_i is a necessary assumption. Otherwise, an adversary can simply guess a value $\alpha \in [0, \ell - 1]$ that is not taken by the s_i and subsequently choose $\bar{z}_\alpha$ randomly while computing all other $\bar{z}_i$ honestly. This would lead to

$$1 = F_0(\bar{z}_0, ..., \bar{z}_\alpha, ..., \bar{z}_m) \neq F_1(\bar{z}_0, ..., \bar{z}_\alpha, ..., \bar{z}_m) = 0$$

because $\bar{z}_\alpha$ is never used on the right side of $\bar{z}_i = h_i \star \bar{z}_{s_i}$ during the trapdoor test in (4). Therefore, the adversary is able to distinguish both functions without guessing all s_i which prevents the aforementioned reduction.

In order to use the trapdoor test in security proofs, we need to choose m and ℓ such that the advantage defined above becomes a small statistical factor. In Section 6, we compute these values for a security level of 128 bits.

5.2 Twin Hashed ElGamal

Applying the twinning technique to Hashed ElGamal yields the Twin Hashed ElGamal encryption scheme GA-Twin-HEG$_m$ for an integer $m \in \mathbb{N}$, which is formally described in Figure 12. While twinning significantly increases the public key size and computation for both encapsulation and decapsulation, it allows us to prove its IND-CCA security without the use of strong variants of the GA-CDH problem. Furthermore, the ciphertext still consists of only one element.

Theorem 6. *Let $\ell, m \in \mathbb{N}$ such that $1 < \ell < m/2$. For any quantum adversary $\mathcal{A}$ against* IND-CCA *security of* GA-Twin-HEG$_m$ *that issues at most q queries*

to the quantum-accessible random oracle H *with query depth* d*, there exists a quantum adversary* $\mathcal{B}$ *against* GA-CDH *such that*

$$\mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathsf{GA\text{-}Twin\text{-}HEG}_m}(\mathcal{A}) \leq 4d\mathsf{Adv}^{\mathsf{GA\text{-}CDH}}_{\mathsf{EGA}}(\mathcal{B}) + 4\sqrt{\frac{(d+1)q}{\ell!\ell^{m-2\ell+1}}} \ ,$$

and the running time of $\mathcal{B}$ *is about three times that of* $\mathcal{A}$ *plus the time to simulate* $\mathcal{O}(\max\{q, q_D\})$ *queries to* H*, where* q_D *is the number of decapsulation queries.*

We will only sketch the proof here and refer to the full version [18] for the full proof. In fact, it is similar to the one of Theorem 3, only that we use the trapdoor test whenever the other proof uses the decision oracle.

Proof (Sketch). Let $\mathcal{A}$ be a quantum adversary in the IND-CCA game. Our goal is to construct an adversary $\mathcal{B}$ against GA-CDH. The main question is how $\mathcal{B}$ simulates decapsulation queries. Therefore, let H_1 and H_2 be *internal* random oracles, the first is used for valid DH tuples and the second for invalid ones. Since for every ciphertext element x_1 there exists a unique vector of m set elements s.t. these form a DH tuple with the public key set elements, the output of H_1 only depends on x_1. We can check if a query consists of valid DH tuples using the trapdoor test. After this change, $\mathcal{B}$ can simulate decapsulation queries by just returning $\mathsf{H}_1(x_1)$. Next, we can apply the MRM-O2H lemma to reprogram H on the challenge ciphertext c^* and the corresponding DH tuples $(\mathsf{sk}[i] \star c^*)_{i\in[m]}$. For this the adversary $\mathcal{B}$ needs to be able to simulate H and H' (the reprogrammed H), which it can do using the trapdoor test. Note that since we applied the variant which considers parallel random oracle queries, the measured inputs are a set of size p. Due to the trapdoor test $\mathcal{B}$ can find the correct solution. In the final game, since the key K^* is now independent of the bit b, the adversary wins the game with probability $1/2$ and the claimed bound follows. □

5.3 Twin NIKE

We construct a NIKE scheme $\mathsf{GA\text{-}Twin\text{-}HDH}_m$ from an effective group action $\mathsf{EGA} = (\mathcal{G}, \mathcal{X}, \star, \tilde{x})$, which defines the public parameters pp together with an integer $m \in \mathbb{N}$ and a hash function $\mathsf{H} : \{0,1\}^* \to \{0,1\}^\kappa$, thus defining $\mathcal{SHK} = \{0,1\}^\kappa$. As in Section 3, we assume that the identities can be represented by bitstrings of fixed length μ. On input an ID, the key generation algorithm chooses m group elements $(g_1, ..., g_m) \xleftarrow{\$} \mathcal{G}^m$ which form the secret key $\mathsf{sk}_{\mathsf{ID}}$. The public key is computed as $\mathsf{pk}_{\mathsf{ID}} = (g_1 \star \tilde{x}, ..., g_m \star \tilde{x}) \in \mathcal{X}^m$. The shared key of an identity ID_1 with public key $\mathsf{pk}_{\mathsf{ID}_1} = (x_1, ..., x_m)$ and an identity ID_2 with secret key $\mathsf{sk}_{\mathsf{ID}_2} = (g_1, ..., g_m)$ is defined as

$$K = \begin{cases} \mathsf{H}(\mathsf{ID}_1, \mathsf{ID}_2, \mathsf{pk}_{\mathsf{ID}_1}, \mathsf{pk}_{\mathsf{ID}_2}, g_1 \star x_1, ..., g_1 \star x_m, ..., g_m \star x_1, ..., g_m \star x_m) & \textbf{if } \mathsf{ID}_1 < \mathsf{ID}_2 \\ \mathsf{H}(\mathsf{ID}_2, \mathsf{ID}_1, \mathsf{pk}_{\mathsf{ID}_2}, \mathsf{pk}_{\mathsf{ID}_1}, g_1 \star x_1, ..., g_m \star x_1, ..., g_1 \star x_m, ..., g_m \star x_m) & \textbf{if } \mathsf{ID}_2 < \mathsf{ID}_1 \end{cases}$$

See Figure 13 for a schematic overview of our construction.

Again, twinning significantly increases the public key size and computation of $\mathsf{GA\text{-}Twin\text{-}HDH}_m$ compared to GA-HDH, but allows us to use the same techniques as in Theorem 6 to prove security without relying on strong assumptions. This is formalized in Theorem 7.

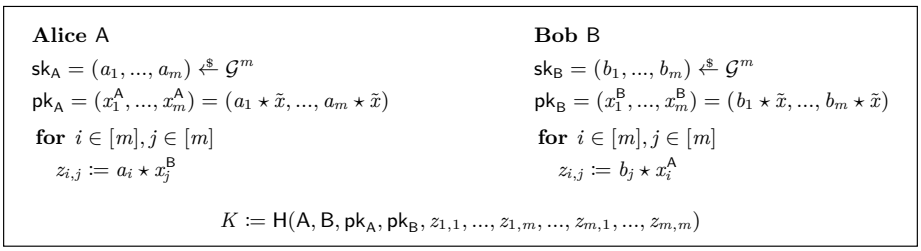

Fig. 13. Our NIKE Protocol $\mathsf{GA\text{-}Twin\text{-}HDH}_m$.

Theorem 7. *Let $\ell, m \in \mathbb{N}$ such that $1 < \ell < m/2$. For any quantum adversary $\mathcal{A}$ against the* CKS *security of* $\mathsf{GA\text{-}Twin\text{-}HDH}_m$ *that issues at most q queries to the quantum-accessible random oracle* H *of query depth d, there exists a quantum adversary $\mathcal{B}$ against* GA-CDH *such that*

$$\mathsf{Adv}^{\mathsf{CKS}}_{\mathsf{GA\text{-}Twin\text{-}HDH}_m}(\mathcal{A}) \leq \sqrt{8d\mathsf{Adv}^{\mathsf{GA\text{-}CDH}}_{\mathsf{EGA}}(\mathcal{B})} + 4\sqrt{\frac{(d+1)q}{\ell!\ell^{m-2\ell+1}}},$$

and the running time of $\mathcal{B}$ is about three times that of $\mathcal{A}$ plus the time needed to simulate $\mathcal{O}(\max\{q, q_R, q_T\})$ queries to the random oracle, to perform $\mathcal{O}(q_O)$ rerandomizations on set elements and to run the trapdoor test $\mathcal{O}(q)$ times, where q_O, q_R and q_T are the number of register-honest, reveal and test queries.

The proof is similar to the proof of Theorem 5 with the main difference that we use the trapdoor test whenever the other proof used the decision oracles. We defer the complete proof to the full version [18].

Proof (Sketch). As in the KEM proof, our goal is to use a variant of the O2H lemma in order to randomize all challenge keys and bound the advantage of the O2H extractor using the GA-CDH assumption. However, instead of just a decapsulation oracle, we have to simulate the CorruptReveal and Test oracles. Although the adversary is allowed to choose identities for honest keys, we can compute the public keys before it makes any queries, so we can vary the behavior of the random oracle when it interacts with honest or corrupted keys. Note that this technique is not generally possible as the key generation could depend on the provided ID in other schemes. This allows us to make similar conceptual changes as in the KEM proof, where we only hash $(\mathsf{ID}_1, \mathsf{ID}_1, \mathsf{pk}_1, \mathsf{pk}_2)$ without the $z_{i,j}$, when at least one key is honest. Additionally, we can use a different internal random oracle, when both keys are honest. By using the trapdoor test,

Scheme	$\lvert\mathsf{pk}\rvert$	$\lvert\mathsf{ct}\rvert$	Gen	Encaps	Decaps	Assumption	Bound
GA-HEG (Fig. 3)	$\lvert\mathcal{X}\rvert$	$\lvert\mathcal{X}\rvert$	1	2	1	GA-FQ-StCDH	$d\,\mathsf{Adv}$
GA-HEG-KC (Fig. 10)	$\lvert\mathcal{X}\rvert$	$\lvert\mathcal{X}\rvert + 4\lambda$	1	2	1	GA-StCDH	$d\sqrt{\mathsf{Adv}}$
GA-Twin-HEG$_m$ (Fig. 12)	$m \cdot \lvert\mathcal{X}\rvert$	$\lvert\mathcal{X}\rvert$	m	$m+1$	m	GA-CDH	$d\,\mathsf{Adv}$
GA-EG-FO [12,17]	$\lvert\mathcal{X}\rvert$	$\lvert\mathcal{X}\rvert + 2\lambda$	1	2	2	GA-CDH	$q\sqrt{\mathsf{Adv}}$
GA-EG-FO [12,30]	$\lvert\mathcal{X}\rvert$	$\lvert\mathcal{X}\rvert + 3\lambda$	1	2	2	GA-DDH	$d^2\mathsf{Adv}$
GA-HDH (Fig. 4)	$\lvert\mathcal{X}\rvert$	-	1	1 (SharedKey)		GA-DFQ-StCDH	$\sqrt{d\,\mathsf{Adv}}$
GA-Twin-HDH$_m$ (Fig. 13)	m	-	m	m^2 (SharedKey)		GA-CDH	$\sqrt{d\,\mathsf{Adv}}$

Table 1. Overview of our different protocols and comparison to FO variants. By $\lvert\mathcal{X}\rvert$ we denote the length of a set element in bits. The columns "Gen", "Encaps" and "Decaps" state the number of group action evaluations that are needed in order to perform the corresponding algorithm. For NIKE schemes this refers to the SharedKey algorithm. Bounds are stated without statistical terms and q, d denote the number of random oracle queries and the query-depth. The security parameter is denoted by λ. For $\lambda = 128$ bit security, we need $m = 85$. For FO-EG we assume the implicit rejection variants.

we can remove the need for the secret keys completely. Finally, we can use the O2H lemma in its semi-classical variant and bound the success probability of its extractor with the GA-CDH assumption. □

6 Parameter Choices and Comparison

In order to compare the different schemes we need to elaborate on the parameter n, which is the bit length of the output of hash function G in the hashed ElGamal scheme with key confirmation, and the twinning parameter m. Both depend on the desired security level which is usually stated in bits. Taking the corresponding terms in the bounds of Theorems 4 and 6 into account, we determine the success ratio of an adversary $\mathcal{A}$. The success ratio of $\mathcal{A}$ is computed as its advantage $\epsilon_{\mathcal{A}}$ divided by its running time $t_{\mathcal{A}}$ [23]. For λ-bit security, we then require $\epsilon_{\mathcal{A}}/t_{\mathcal{A}} \leq 2^{-\lambda}$.

Key Confirmation. The output of the hash function G determines the length of the second ciphertext element. In order to determine the length, we analyze the statistical terms in Theorem 4. Note the one with the fourth root is the most dominating one. Thus, for λ-bit security, we need to set $n \approx 4\lambda$, where we assume $q_D \leq q \lessapprox t_{\mathcal{A}}$ and ignore additive constants.

Twinning. The efficiency of the Twin ElGamal encryption scheme GA-Twin-HEG$_m$ and the Twin NIKE scheme GA-Twin-HDH$_m$ depends on the twinning parameter m which directly translates to the length of the public key. The security level is determined by the value of $\ell!\ell^{m-2\ell+1}$, where $\ell \in [1, m/2]$ may be chosen

arbitrarily. Note that ℓ only appears in the proofs of Theorem 6 and Theorem 7, hence it has no direct effect on the corresponding protocols.

Again, we only analyze the statistical term in the bound. For λ-bit security, we need

$$\frac{4}{t_{\mathcal{A}}} \cdot \sqrt{\frac{(d+1)q}{\ell!\ell^{m-2\ell+1}}} \leq 2^{-\lambda}.$$

Similar as before, we may assume that $d \leq q \lessapprox t_{\mathcal{A}}$, hence for an optimal success ratio an adversary would choose $d = q$. This means that we need to choose m large enough so that $\ell!\ell^{m-2\ell+1} \geq 2^{2\lambda+4}$ for some $\ell \in [1, m/2]$. As an example, for $\lambda = 128$, optimality is achieved by $m = 85$ (with $\ell = 17$).

Instantiation of the Group Action. Every set element $x \in \mathcal{X}$ is represented by a bitstring. In CSIDH the length of this bitstring is $\log(p)$, where the size of $\mathcal{X}$ is in $O(\sqrt{p})$. Choosing the correct parameter size for CSIDH is an actively discussed topic in the community. Castryck et al. [12] propose a 1792-bit prime p to achieve $\lambda = 128$ bit quantum security.

Comparison. Table 1 provides an overview of the schemes analyzed in this paper and a comparison to the ElGamal KEMs that can be obtained by the FO transform. The base scheme is the most efficient one, with one ciphertext element and two group action evaluations for Encaps. It also achieves the best QROM bound without any square root terms, but it relies on the strongest non-standard assumption. Hashed ElGamal with key confirmation has a slightly larger ciphertext and comes with a worse bound, however, it relies only on the GA-StCDH assumption. Since twinning cannot be done efficiently in the group action setting, the twinned version of hashed ElGamal is the least efficient in terms of public key size and group action computation. Nevertheless, the ciphertext still consists of only one set element and we get security based on the standard GA-CDH assumption. At this point we want to stress again that this seems the only way to construct an actively-secure NIKE based on a standard assumption. Otherwise, one has to rely on the assumption with a quantum-accessible decision oracle.

Acknowledgments

The work of Julien Duman was supported by the German Federal Ministry of Education and Research (BMBF) in the course of the 6GEM Research Hub under Grant 16KISK037. Dominik Hartmann was supported by the BMBF iBlockchain project. Eike Kiltz was supported by the BMBF iBlockchain project, the Deutsche Forschungsgemeinschaft (DFG, German research Foundation) as part of the Excellence Strategy of the German Federal and State Governments – EXC 2092 CASA - 390781972, and by the European Union (ERC AdG REWORC - 101054911). Sabrina Kunzweiler, Jonas Lehmann and Doreen Riepel were funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy - EXC 2092 CASA - 390781972.

References

1. M. Abdalla, M. Bellare, and P. Rogaway. The oracle Diffie-Hellman assumptions and an analysis of DHIES. *CT-RSA 2001*, vol. 2020 of *LNCS*, 143–158. 2001.
2. N. Alamati, L. De Feo, H. Montgomery, and S. Patranabis. Cryptographic group actions and applications. *ASIACRYPT 2020, Part II*, vol. 12492 of *LNCS*, 411–439. 2020.
3. J. Alwen, B. Blanchet, E. Hauck, E. Kiltz, B. Lipp, and D. Riepel. Analysing the HPKE standard. *EUROCRYPT 2021, Part I*, vol. 12696 of *LNCS*, 87–116. 2021.
4. J. Alwen, D. Hartmann, E. Kiltz, and M. Mularczyk. Server-aided continuous group key agreement. Cryptology ePrint Archive, Report 2021/1456, 2021.
5. A. Ambainis, M. Hamburg, and D. Unruh. Quantum security proofs using semi-classical oracles. *CRYPTO 2019, Part II*, vol. 11693 of *LNCS*, 269–295. 2019.
6. R. Barnes, J. Millican, E. Omara, K. Cohn-Gordon, and R. Robert. Message layer security (mls) wg. https://datatracker.ietf.org/wg/mls/about/.
7. M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. *ACM CCS 93*, 62–73. 1993.
8. N. Bindel, M. Hamburg, K. Hövelmanns, A. Hülsing, and E. Persichetti. Tighter proofs of CCA security in the quantum random oracle model. *TCC 2019, Part II*, vol. 11892 of *LNCS*, 61–90. 2019.
9. D. Boneh, Ö. Dagdelen, M. Fischlin, A. Lehmann, C. Schaffner, and M. Zhandry. Random oracles in a quantum world. *ASIACRYPT 2011*, vol. 7073 of *LNCS*, 41–69. 2011.
10. D. Boneh and M. Zhandry. Secure signatures and chosen ciphertext security in a quantum computing world. *CRYPTO 2013, Part II*, vol. 8043 of *LNCS*, 361–379. 2013.
11. D. Cash, E. Kiltz, and V. Shoup. The twin Diffie-Hellman problem and applications. *EUROCRYPT 2008*, vol. 4965 of *LNCS*, 127–145. 2008.
12. W. Castryck, T. Lange, C. Martindale, L. Panny, and J. Renes. CSIDH: An efficient post-quantum commutative group action. *ASIACRYPT 2018, Part III*, vol. 11274 of *LNCS*, 395–427. 2018.
13. K. Cohn-Gordon, C. Cremers, K. Gjøsteen, H. Jacobsen, and T. Jager. Highly efficient key exchange protocols with optimal tightness. *CRYPTO 2019, Part III*, vol. 11694 of *LNCS*, 767–797. 2019.
14. J.-M. Couveignes. Hard homogeneous spaces. Cryptology ePrint Archive, Report 2006/291, 2006.
15. B. de Kock, K. Gjøsteen, and M. Veroni. Practical isogeny-based key-exchange with optimal tightness. *Selected Areas in Cryptography*, 451–479. 2021.
16. W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.
17. J. Don, S. Fehr, C. Majenz, and C. Schaffner. Online-extractability in the quantum random-oracle model. *Advances in Cryptology – EUROCRYPT 2022*, 677–706. 2022.
18. J. Duman, D. Hartmann, E. Kiltz, S. Kunzweiler, J. Lehmann, and D. Riepel. Group action key encapsulation and non-interactive key exchange in the QROM. Cryptology ePrint Archive, Report 2022/1230, 2022.
19. T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31:469–472, 1985.
20. T. B. Fouotsa and C. Petit. Sims: a simplification of sigamal. *International Conference on Post-Quantum Cryptography*, 277–295. 2021.

21. E. S. V. Freire, D. Hofheinz, E. Kiltz, and K. G. Paterson. Non-interactive key exchange. *PKC 2013*, vol. 7778 of *LNCS*, 254–271. 2013.
22. E. Fujisaki and T. Okamoto. Secure integration of asymmetric and symmetric encryption schemes. *CRYPTO'99*, vol. 1666 of *LNCS*, 537–554. 1999.
23. J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999.
24. D. Hofheinz, K. Hövelmanns, and E. Kiltz. A modular analysis of the Fujisaki-Okamoto transformation. *TCC 2017, Part I*, vol. 10677 of *LNCS*, 341–371. 2017.
25. K. Hövelmanns, E. Kiltz, S. Schäge, and D. Unruh. Generic authenticated key exchange in the quantum random oracle model. *PKC 2020, Part II*, vol. 12111 of *LNCS*, 389–422. 2020.
26. A. Hülsing, K.-C. Ning, P. Schwabe, F. Weber, and P. R. Zimmermann. Post-quantum WireGuard. *2021 IEEE Symposium on Security and Privacy (SP)*, 304–321. 2021.
27. A. Hülsing, J. Rijneveld, and F. Song. Mitigating multi-target attacks in hash-based signatures. *PKC 2016, Part I*, vol. 9614 of *LNCS*, 387–416. 2016.
28. T. Kawashima, K. Takashima, Y. Aikawa, and T. Takagi. An efficient authenticated key exchange from random self-reducibility on CSIDH. *ICISC 20*, vol. 12593 of *LNCS*, 58–84. 2020.
29. H. Krawczyk and H. Wee. The optls protocol and tls 1.3. *2016 IEEE European Symposium on Security and Privacy (EuroS P)*, 81–96. 2016.
30. V. Kuchta, A. Sakzad, D. Stehlé, R. Steinfeld, and S. Sun. Measure-rewind-measure: Tighter quantum random oracle model proofs for one-way to hiding and CCA security. *EUROCRYPT 2020, Part III*, vol. 12107 of *LNCS*, 703–728. 2020.
31. T. Moriya, H. Onuki, and T. Takagi. SiGamal: A supersingular isogeny-based PKE and its application to a PRF. *ASIACRYPT 2020, Part II*, vol. 12492 of *LNCS*, 551–580. 2020.
32. E. Rescorla, K. Oku, N. Sullivan, and C. A. Wood. TLS Encrypted Client Hello. Internet-Draft draft-ietf-tls-esni-13, Internet Engineering Task Force, 2021.
33. E. Rescorla, N. Sullivan, and C. A. Wood. Semi-Static Diffie-Hellman Key Establishment for TLS 1.3. Internet-Draft draft-rescorla-tls-semistatic-dh-02, Internet Engineering Task Force, 2019.
34. A. Rostovtsev and A. Stolbunov. Public-Key Cryptosystem Based On Isogenies. Cryptology ePrint Archive, Report 2006/145, 2006.
35. P. Schwabe, D. Stebila, and T. Wiggers. Post-quantum TLS without handshake signatures. *ACM CCS 2020*, 1461–1480. 2020.
36. P. W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. *35th FOCS*, 124–134. 1994.
37. D. Unruh. Revocable quantum timed-release encryption. *EUROCRYPT 2014*, vol. 8441 of *LNCS*, 129–146. 2014.
38. K. Yoneyama. Post-quantum variants of ISO/IEC standards: Compact chosen ciphertext secure key encapsulation mechanism from isogeny. *Proceedings of the 5th ACM Workshop on Security Standardisation Research Workshop*, SSR'19, 13–21. 2019.
39. M. Zhandry. How to construct quantum random functions. *53rd FOCS*, 679–687. 2012.