VSS from Distributed ZK Proofs and Applications

Shahla Atapoor[1], Karim Baghery[1], Daniele Cozzo[2,1], and Robi Pedersen[1]

[1] COSIC, KU Leuven, Leuven, Belgium
[2] IMDEA Software Institute, Madrid, Spain
firstname.lastname@kuleuven.be
June 26, 2023

Abstract. Non-Interactive Verifiable Secret Sharing (NI-VSS) is a technique for distributing a secret among a group of individuals in a verifiable manner, such that shareholders can verify the validity of their received share and only a specific number of them can access the secret. VSS is a fundamental tool in cryptography and distributed computing. In this paper, we present an efficient NI-VSS scheme using Zero-Knowledge (ZK) proofs on secret shared data. While prior VSS schemes have implicitly used ZK proofs on secret shared data, we specifically use their formal definition recently provided by Boneh et al. in CRYPTO 2019. Our proposed NI-VSS scheme uses a quantum random oracle and a quantum computationally hiding commitment scheme in a black-box manner, which ensures its ease of use, especially in post-quantum threshold protocols. The practicality of the proposed NI-VSS is confirmed by our implementation results, establishing it as a viable choice for large-scale threshold protocols. Using the proposed NI-VSS scheme, a dealer can share a secret with 4096 parties in less than 2 *seconds*, and the shareholders can verify the validity of their shares in less than 2 *milliseconds*. We demonstrate the potential of new NI-VSS scheme by revisiting several threshold protocols and improving their efficiency. Specifically, we present two DKG protocols for CSIDH-based primitives, that outperform the current state of the art. Furthermore, we show similar improvements in some threshold signatures built based on Schnorr and CSI-FiSh signature schemes. We think, due to its remarkable efficiency and ease of use, the new NI-VSS scheme can be a valuable tool for a wide range of threshold protocols.

Keywords: Verifiable Secret Sharing · ZK Proofs on Secret Shared Data · Shamir Secret Sharing · DKG · Threshold Signatures · CSIDH.

1 Introduction

Secret sharing schemes are fundamental tools in the field of threshold cryptography and secure multi-party computation. Such schemes consist of a sharing phase, where a dealer shares a secret among the shareholders, followed by a reconstruction phase where qualified shareholders collaborate to reconstruct the original secret. Standard secret sharing schemes, such as Shmair's protocol [32],

assume the presence of honest parties but do not provide security against malicious participants. Verifiable Secret Sharing (VSS) schemes [14, 19] have been developed to address the challenges posed by malicious players. These schemes aim to withstand various attacks, including incorrect share distribution by the dealer and malicious behavior by the shareholders (e.g., using incorrect shares) during the reconstruction phase. Depending on the communication model, to incorporate verifiability, typically interaction among the dealer and shareholders is required. It can however be shown that, assuming the dealer has a broadcast channel, a single message from the dealer to the shareholders can be sufficient. This is known as a Non-Interactive VSS (NI-VSS).

Most existing constructions of VSS schemes are based on regular secret-sharing schemes, often starting with Shamir's scheme [32], and then adding verifiability features on top [4, 14, 19, 22–24, 28, 31]. The known discrete-logarithm (DL) based VSS schemes such as those by Feldman [19], Pedersen [28], Schoenmakers [31], and their variants, utilize Shamir secret sharing and achieve verifiability by having the dealer publish the shares and coefficients of the underlying secret polynomial in the group. Then, they leverage the homomorphic property of the group to convince anyone, not just the shareholders, that the secret sharing is performed correctly. DL-based VSS schemes are typically non-interactive and support public verifiability, allowing both shareholders and external verifiers to verify the validity of the shares without interaction. However, due to the threat posed by Shor's algorithm [33], discrete logarithm based VSS schemes are not suitable for cryptographic protocols (e.g., distributed key generation schemes, threshold signatures, etc.), that require post-quantum security. Gentry, Halevi, and Lyubashevsky [23] recently proposed a practical non-interactive publicly VSS scheme that relies on lattice-based and DL-based problems at the same time, unfortunately making it unsuitable for use in post-quantum secure threshold protocols. Given the limitations and vulnerabilities of existing NI-VSS schemes, it becomes imperative to develop an efficient post-quantum secure VSS scheme that can also address challenges of scalability and computational overhead. Such VSS schemes can pave the way for the realization of more efficient post-quantum threshold protocols.

In the VSS scheme proposed by Ben-or, Goldwasser, and Wigderson (BGW) [4], a dealer employs a distributed Zero-Knowledge (ZK) proof scheme based on bivariate polynomials to add (designated) verifiability to the Shamir secret sharing scheme. The BGW VSS scheme achieves Information-Theoretical (IT) security and can be employed in both classical and post-quantum secure threshold protocols. However, in their (non-interactive) ZK proof scheme the verifiers need to interact two-by-two for share validation and to achieve (perfect) soundness their scheme requires at least two-thirds of the designated verifiers to be honest.

In Crypto 2019, Boneh et al. [10] provided a formal definition for ZK proofs over secret shared data and presented several feasibility and infeasibility results. In a ZK proof scheme over secret shared data, there is a single prover P and n (designated) verifiers $\{V_i\}_{i=1}^{n}$, and each verifier V_i holds a piece (share) x_i of an input (statement) x, which is distributed among n participants. The prover's

task is to convince the (designated) verifiers that the main input x belongs to a specific language L. Essentially, the prover P possesses full knowledge of x, while each verifier V_i possesses a secret share denoted as x_i. In their best feasibility (and positive) result, Boneh et al. [10] demonstrated that, in the majority honest setting, using a robust encoding scheme, any multi-round public-coin linear Interactive Oracle Proof (IOP) for a non-distributed relation R_L can be compiled into a secure ZK proof scheme over secret shared data. The resulting distributed ZK proof scheme satisfies (computational) soundness against the prover and $t < n/2$ malicious verifiers. Moreover, it guarantees ZK even if t of the verifiers collude [10, Section 6.3], where t represents a threshold parameter in the underlying encoding scheme. Boneh et al. [10] coined the term "Strong zero-knowledge" (SZK) to describe this later variant of ZK, which ensures that even if up to t verifiers collude, they learn nothing about the witness. Based on the formal definitions, we can restate that the BGW VSS scheme [4] has been proven to achieve SZK and (perfect) soundness against the prover, given that at least two-thirds of the (designated) verifiers are honest.

Consequently, a generic approach for constructing a ZK proof scheme over secret shared data for n-distributed relations R_i involves first developing a multi-round public-coin IOP for the non-distributed relation R. Subsequently, Boneh et al.'s compiler can be utilized to transform it into a distributed ZK proof scheme, featuring a single prover P and n designated verifiers $\{V_i\}_{i=1}^n$. However, it is worth noting that generic approaches are typically less efficient compared to ad-hoc constructions tailored for specific purposes in practical implementations.

Our Contributions. We summarize the contributions of this paper as follows:

An Efficient Post-Quantum Secure NI-SZK for the Shamir Relation. Considering the feasibility result of Boneh et al. [10], we directly (without using their compiler [10]) construct an efficient Non-Interactive SZK (NI-SZK) proof scheme for the n-distributed relations $R_1, \ldots, R_n$, where

$$R_i = \{(x_i, f(X)) | f(i) = x_i\}. \tag{1}$$

Here $f(X) \in \mathbb{Z}_N[X]_t$ is a secret polynomial in X of degree (at most) t and with coefficients defined over the ring $\mathbb{Z}_N$. The proposed construction is built in the majority honest setting (i.e., the majority of the verifiers are honest) and utilizes a quantum computationally hiding commitment scheme. We prove (in Theorem 3.1) that in the Quantum Random Oracle Model (QROM), the proposed NI-SZK proof scheme satisfies completeness, SZK, and soundness against the prover and t malicious verifiers, as formally defined in [10].

NI-VSS Schemes from NI-SZK Proofs and a Quantum Secure Construction. We further show how one can use a secure NI-SZK proof scheme for the n-distributed relations given in equation (1), and build a *computational secure* NI-VSS scheme based on Shamir secret sharing in the majority honest setting. Building upon that, we use the proposed NI-SZK proof scheme and present an extremely efficient post-quantum computationally secure NI-VSS scheme that works over general rings, and is proven to be secure in the QROM (in Theorem 3.2).

We examined the empirical performance of the new NI-VSS scheme by implementing a prototype in C++, using the libraries `libsodium` and `NTL`. Results from the implementation show that our proposed NI-VSS scheme allows a dealer to securely share a secret with 4096 parties in less than 2 seconds. Furthermore, shareholders can efficiently verify the validity of their shares non-interactively in less than 2 milliseconds. When considering the same number of parties and aiming for 128-bit quantum security, the dealer broadcasts a proof of approximately 320KB in size. Additionally, the dealer privately sends less than 64B (excluding the share itself) to each shareholder. The simplicity and efficiency of our new NI-VSS scheme make it an attractive choice for various large-scale threshold protocols, especially those that require post-quantum security.

Our resulting NI-VSS scheme serves as a post-quantum secure alternative to the classical Pedersen VSS scheme [28] (or Feldman's VSS scheme [19]) when public verifiability is not needed. We later show that this scenario often arises in several threshold protocols. One can also view it as an alternative to the Information-Theoretically (IT) secure BGW VSS [4] in cases where quantum computational security suffices and one wants to reduce the communication between parties, or when assuming two-thirds honest parties among the shareholders is difficult. Table 1 provides a comprehensive summary of the key features of the proposed post-quantum secure NI-VSS scheme, comparing it to the BGW VSS scheme from various perspectives.

Applications of our NI-VSS. As the third major contribution of this paper, we leverage the proposed NI-VSS scheme to revisit several threshold protocols based on isogenies and discrete logarithms, aiming to enhance their efficiency and decrease the lower bound on the number of honest parties. Our initial observation reveals that our NI-VSS scheme can be integrated into various threshold protocols where the BGW VSS scheme [4] is used. By doing so, one can improve the target protocols in terms of communication costs and the maximum tolerance for malicious parties, but at the expense of mitigating IT security to quantum computational security.

Table 1. A comparison between BGW [4] and our proposed NI-VSS schemes.

BGW VSS [4]	**This Work**
designated verifier	designated verifier
uses a ZK proof over secret shared data to prove the relations in eq. (1), and the verification requires interaction between the verifiers	uses a ZK proof over secret shared data to prove the relations in eq. (1), and the verification does not require interaction between the verifiers
achieves IT security	achieves post-quantum computational security
requires $\geq \frac{2}{3}$ honest parties	requires $\geq \frac{1}{2}$ honest parties
$O(n)$ communication from dealer (to the n verifiers)	$O(n)$ communication from dealer (to the n verifiers)
verification is interactive, and induces $O(n)$ communication with other n verifiers	verification is non-interactive, so does not need communication with other verifiers

Based on the above observation, we revisit two DKG protocols recently proposed by Atapoor, Baghery, Cozzo, and Pedersen [2], which are currently the most efficient ones in terms of isogeny computations within the CSIDH (Commutative Supersingular Isogeny Diffie Hellman) setting [13]. We show that, by integrating the new NI-VSS scheme into the VSS step of their DKG protocols, we can address two bottlenecks present in their DKG protocols, i.e. we reduce the requirement of having at least $2/3$ honest shareholders to a more practical threshold of just $1/2$ and eliminates the need for pairwise interactive verification, which was a primary reason to the high communication overhead in the VSS step of their DKG protocols. While these enhancements do come at the cost of sacrificing IT security in the VSS step for quantum computational security, we do note that the DKG protocols proposed in [2] do in fact rely on quantum computational security anyways. These advancements make the revisited DKG protocols highly appealing to be used in CISDH-based threshold settings. The resulting DKG protocols come with the same efficiency in terms of the isogeny computations.

As further applications for the proposed NI-VSS scheme and DKG protocols, we revisit two robust threshold signatures [12, 21] based on CSI-FiSh [8] and Schnorr [30] signature schemes. The first scheme, proposed by Campos and Muth [12], is based on the basic version of CSI-FiSh, which features shorter public keys but longer signature sizes, as well as slower signing and verification algorithms. To enhance the efficiency of their threshold robust signing protocol [12], we apply two key modifications. First, we adapt their scheme to work with the CSI-SharK signature [1], which has been demonstrated to outperform CSI-FiSh in the threshold setting. This modification allows us to utilize one of the revisited DKG protocols from this paper, enabling more efficient key sampling for the resulting robust threshold signature scheme. Moreover, we use a similar strategy employed in the construction of revisited DKG protocols to enhance the efficiency of ephemeral key generation in the modified threshold signature scheme. This further improves the efficiency of the distributed signing protocol, resulting in a new scheme called ThreshER SharK. ThreshER SharK represents a **Thresh**old, **E**fficient, **R**obust CSI-**SharK**-based signature scheme, which addresses the limitations of the original scheme by achieving improved efficiency and shorter secret keys.

Our second revisited threshold signature scheme is based on Schnorr's scheme presented by Gennaro, Jarecki, Krawczyk, and Rabin [21], and is built upon Pedersen's DKG protocol for a key generation [27]. We start by constructing a NI-SZK proof scheme for the following n-distributed relations,

$$R_i = \{(y, x_i, f(X)) | y = g^{f(0)} \wedge f(i) = x_i\}, \tag{2}$$

for $i = 1, \ldots, n$. This NI-SZK proof scheme serves as the main component for our revisions and holds potential interest in other DL-based threshold protocols that utilize Shamir secret sharing. We then present a new variant of Pedersen's DKG protocol [27] along with a variant of the robust threshold signature scheme proposed by Gennaro, Jarecki, Krawczyk, and Rabin [21]. The security of new variants is proven in the random RO model.

When comparing our resulting variants to the original schemes, there are certain trade-offs. While our variants sacrifice public verifiability and slightly increase communication costs, they improve computational efficiency in both schemes. In our proposed variant of the Pedersen DKG protocol, each party is required to perform approximately $2n$ exponentiations in the group, $5n$ (short) hashes, and n degree-t polynomial evaluations in the field. This is an improvement compared to the secure version of Pedersen DKG protocol [21], which demands roughly $2tn+2n$ exponentiations in the group. In practical scenarios, the computation of $5n$ (short) hashes and n degree-t polynomial evaluations can be faster than $2nt \approx n^2$ exponentiations in the group, especially in large-scale applications. Furthermore, in comparison to Gennaro et al.'s threshold signature [21], our approach necessitates each party to compute approximately $2n$ exponentiations in the group, $5n$ hashes, and $2n$ degree-t polynomial evaluations in the field, as opposed to around $2tn + 4n$ exponentiations in the group. Similarly, in large-scale applications, performing $5n$ (short) hashes along with $2n$ degree-t polynomial evaluations can offer greater efficiency compared to $2nt+2n \approx n^2+2n$ exponentiations in the group.

Outline. In Sec. 2, we provide an overview of some preliminary concepts. In Sec. 3, we introduce a general construction for building NI-VSS schemes from NI-SZK schemes. We then present an efficient NI-VSS scheme that is secure in QROM and evaluate its performance through a prototype C++ implementation. In Sec. 4, leveraging the proposed NI-VSS scheme, we introduce two more efficient DKG protocols tailored for the CSIDH-based primitives. In Sec. 5, we revisit two threshold signature schemes based on Schnorr and CSI-FiSh signatures and present enhanced versions that offer improved efficiency. Finally, we summarize our findings and conclusions in Sec. 6.

2 Preliminaries

Notation. We let λ denote a security parameter. A function is called *negligible in* X, written $\mathsf{negl}(X)$, if for any constant c, there exists some X_0, such that $f(X) < X^{-c}$ for $X > X_0$. A function that is negligible in the security parameter λ is simply called negligible. We use the assignment operator $\leftarrow$ to denote uniform sampling from a set Ξ, e.g. $x \leftarrow \Xi$. We write $\mathbb{Z}_N := \mathbb{Z}/N\mathbb{Z}$ and $\mathbb{Z}_N[X]_t$ for polynomials of degree t in the variable X and with coefficients in $\mathbb{Z}_N$. For $n \in \mathbb{N}$, we write $[n] = \{1, \ldots, n\}$. Finally all logarithms are in base 2.

We also introduce the notion of *exceptional sets*, which occur naturally when working over rings $\mathbb{Z}_N$.

Definition 2.1 (Exceptional set [3, 9, 15]). *An* exceptional set (modulo N) *is a set $\Xi_k = \{c_1, \ldots, c_k\} \subseteq \mathbb{Z}_N$, where the pairwise difference of all distinct elements is invertible modulo N. If further the pairwise sum of* all *elements is invertible modulo N, Ξ_k is called a* superexceptional set (modulo N).

2.1 Zero-Knowledge Proofs on Secret Shared Data

In typical NIZK (non-interactive zero-knowledge) arguments for NP languages, there is a single prover P and a single verifier V, where P knows both a statement x and witness for the statement w, while V only knows the statement x. In CRYPTO 2019, Boneh et al. [10], presented formal definitions for distributed ZK proofs which a prover interacts with several verifiers $\{V_i\}_{i=1}^n$ over a network that includes secure point-to-point channels. In such a model, each verifier V_j holds a piece (share) $x^{(j)} \in \mathbb{F}^{l_j}$ of an input (statement) x, and the prover's task is to convince the verifiers that the main input x is in some language $L \subseteq \mathbb{F}^l$.

Similar to the typical cases, such proof systems must be complete, meaning that if $x \in L$, an honest prover will be able to convince honest verifiers. Similarly, they should satisfy soundness, meaning that if $x \notin L$, then all verifiers will reject the verification except for a negligible probability. However, in certain settings, including ours, a limited number of verifiers may be malicious and collude with the adversarial prover. In such cases, the malicious verifiers might accept a fake proof. Finally, the proof system must satisfy a variant of ZK, so called *strong ZK*, as introduced by Boneh et al. [10]. SZK implies that any subset of the verifiers up to a certain bound should learn no additional information about statement x, beyond their own shares and the fact that $x \in L$. Note that in standard ZK, the verifier learns the statement x and the fact that $x \in L$, but in strong ZK, a single verifier only learns his share of x and the fact that $x \in L$. In other words, a set of verifiers only learn that they are jointly holding pieces (shares) of $x \in L$.

Definition 2.2 (Distributed Inputs, Languages, and Relations [10]). *Let n be a number of parties, $\mathbb{F}$ be a finite field, and $l, l_1, l_2, \cdots, l_n \in \mathbb{N}$ be length parameters, where $l = l_1 + l_2 + \cdots + l_n$. An n-distributed input over $\mathbb{F}$ (or just distributed input) is a vector $x = x^{(1)} \parallel x^{(2)} \parallel \cdots \parallel x^{(n)} \in \mathbb{F}^l$ where $x^{(i)} \in \mathbb{F}^{l_i}$, and it refers to a piece (or share) of x. An n-distributed language L is a set of n-distributed inputs. A distributed NP relation with witness length h is a binary relation $R(x, w)$ where x is an n-distributed input and $w \in \mathbb{F}^h$. We assume that all x in L and $(x, w) \in R$ share the same length parameters. Finally, we let $L_R = \{x : \exists w (x, w) \in R\}$.*

Next, we recall the formal definition provided by Boneh et al. [10] for ZK proofs over shared data which originally are defined over a field. In some cases, we employ an extended version of their definitions that naturally encompasses rings. In this model, parties can have synchronous communication over secure point-to-point channels.

Definition 2.3 (n-Verifier Interactive Proofs [10]). *An n-*Verifier Interactive Proof *protocol over $\mathbb{F}$ is an interactive protocol $\Pi = (P, V_1, V_2, \cdots, V_n)$ involving a prover P and n verifiers $V_1, V_2, \cdots, V_n$. The protocol proceeds as follows.*

- *In the beginning of the protocol the prover holds an n-distributed input $x = x^{(1)} \parallel x^{(2)} \parallel \cdots \parallel x^{(n)} \in \mathbb{F}^l$, a witness $w \in \mathbb{F}^h$, and each verifier V_j holds an input piece (or share) $x^{(j)}$.*

- *The protocol allows the parties to communicate in synchronous rounds over secure point-to-point channels. While honest parties send messages according to Π, malicious parties (i.e., adversary) can send arbitrary messages.*
- *At the end, each verifier outputs either 1 (accept) or 0 (reject) based on its view, where the view of V_j consists of its input piece $x^{(j)}$, random input $r^{(j)}$, and messages it received during the protocol execution.*

In the rest, $\Pi(x, w)$ denotes running Π on shared input x and witness w, and says that $\Pi(x, w)$ accepts (respectively, rejects) if at the end all verifiers output 1 (resp., 0). $View_{\Pi,T}(x, w)$ denotes the (joint distribution of) views of verifiers $\{V_j\}_{j\in T}$ in the execution of Π on the distributed input x and witness w.

Let $R(x, w)$ be a k-distributed relation over finite field $\mathbb{F}$. We say that an n-verifier interactive proof protocol $\Pi = (P, V_1, \cdots, V_n)$ is a distributed strong ZK proof protocol for R with t-security against malicious prover *and* malicious verifiers, and with soundness error ϵ, if Π satisfies the following properties [10]:

Definition 2.4 (Completeness). *For every n-distributed input $x = x^{(1)} \parallel x^{(2)} \parallel \cdots \parallel x^{(n)} \in \mathbb{F}^l$, and witness $w \in \mathbb{F}^h$, such that $(x, w) \in R$, the execution of $\Pi(x^{(1)} \parallel x^{(2)} \parallel \cdots \parallel x^{(n)}, w)$ accepts with probability 1.*

Definition 2.5 (Soundness Against Prover and t Verifiers.). *For every $T \subseteq [n]$ of size $|T| \leq t$, a malicious adversary $\mathcal{A}$ controlling the prover P and verifiers $\{V_j\}_{j\in T}$, n-distributed input $x = x^{(1)} \parallel x^{(2)} \parallel \cdots \parallel x^{(n)} \in \mathbb{F}^l$, and witness $w \in \mathbb{F}^h$ the following holds. If there is no n-distributed input $x' \in L_R$ such that $x'_H = x_H$, where $H = [n]/T$, the execution of $\Pi^\star(x, w)$ rejects except with at most ϵ probability, where here $\Pi^\star$ denotes the interaction of $\mathcal{A}$ with the honest verifiers.*

Definition 2.6 (Strong ZK). *For every $T \subseteq [n]$ of size $|T| \leq t$ and a malicious adversary $\mathcal{A}$ controlling the verifiers $\{V_j\}_{j\in T}$, there exists a simulator $\mathcal{S}$ such that for every n-distributed input $x = x^{(1)} \parallel x^{(2)} \parallel \cdots \parallel x^{(n)} \in \mathbb{F}^l$, and witness $w \in \mathbb{F}^h$ such that $(x, w) \in R$, we have $\mathcal{S}((x^{(j)})_{j\in T}) \equiv View_{\Pi^\star,T}(x, w)$. Here, $\Pi^\star$ denotes the interaction of adversary $\mathcal{A}$ with the honest prover P and the honest verifiers $\{V_j\}_{j\in T}$.*

Remark 2.1 (Strong Honest-Verifier ZK). In the context of Strong ZK, one may consider a relaxed definition, Strong *Honest-Verifier* ZK, that retains the same properties as the original definition, with the added requirement that the subset of verifiers, $\{V_j\}_{j\in T}$, is stipulated to follow the protocol honestly.

2.2 Verifiable Secret Sharing

Secret sharing is a technique for securely distributing a secret among a group of parties, where no single party can learn the secret individually. However, when a sufficient number of parties come together and combine their 'shares', the original secret can be reconstructed. Throughout the paper, our studied protocols use Shamir Secret Sharing [32] for securely sharing a secret, which we review below.

Shamir Secret Sharing. A $(t+1, n)$-Shamir secret sharing scheme [32] allows n parties to individually hold a share x_i of a common secret x_0, such that any subset of t parties or less are not able to learn any information about the secret x_0, while any subset of at least $t+1$ parties are able to efficiently reconstruct the common secret x_0. In more detail, this is achieved via polynomial interpolation over the ring $\mathbb{Z}_N$. A common polynomial $f(x) \in \mathbb{Z}_N[x]_t$ is chosen, such that the secret x_0 is set to be its constant term, namely $x_0 = f(0)$. Each party P_i for $i \in \{1, \cdots, n\}$ is assigned the secret share $x_i = f(i)$. Then any subset $Q \subseteq \{1, \ldots, n\}$ of at least t parties can reconstruct the secret x_0 via Lagrange interpolation by computing $x_0 = f(0) = \sum_{i \in Q} x_i \cdot L_{0,i}^Q$, where

$$L_{0,i}^Q := \prod_{j \in Q \setminus \{i\}} \frac{j}{j-i} \pmod N.$$

are the Lagrange basis polynomials evaluated at 0. Any subset of less than t parties are not able to find $x_0 = f(0)$, as this is information theoretically hidden from the other shares. In the case where $\mathbb{Z}_N$ is a ring, the difference of any elements in $\{1, \ldots, n\}$ must be invertible modulo N, thus $\{1, \ldots, n\}$ must be an exceptional set. This is only the case if n is smaller than the smallest prime divisor q of N. In the case where more than q parties want to participate in the protocol, we would have to work in a subgroup $\mathbb{Z}_{N'} \subset \mathbb{Z}_N$ such that the smallest divisor of N' is larger than q.

Verifiable Secret Sharing (VSS). A standard secret sharing scheme is designed to be resilient against passive attacks. In many applications, a secret sharing scheme needs to be secure against the malicious dealer or parties with active attacks. This is achieved through VSS schemes, which were first introduced in 1985 [14]. Shamir secret sharing scheme by default does not qualify as a VSS scheme, as it does not provide protection against malicious participants (i.e., the dealer and shareholders).

2.3 Isogenies: Introduction, (Structured) PVPs, DKG Protocols

Isogeny-based Cryptography. Isogenies are rational maps between elliptic curves that are also homomorphisms with respect to the natural group structure on these curves. Our investigation is limited to the set $\mathcal{E}$ of supersingular elliptic curves over prime fields $\mathbb{F}_p$ and separable $\mathbb{F}_p$-rational isogenies defined between them (the so-called CSIDH setting). Isogenies from an elliptic curve to itself are called endomorphisms. Under the addition and composition operations, the endomorphisms of elliptic curves form a ring. The subring of $\mathbb{F}_p$-rational endomorphism rings of curves in $\mathcal{E}$ is always isomorphic to an order $\mathcal{O}$ in the quadratic imaginary field $\mathbb{Q}(\sqrt{-p})$. Separable isogenies are uniquely defined by their kernel, which can be identified with the kernels of ideal classes in the ideal-class group $\mathrm{cl}(\mathcal{O})$. As a result, we can see the class group as acting on the set $\mathcal{E}$ via a free and transitive group action.

To ensure efficient computation of isogenies, the prime p is usually chosen such that $p - 1 = 4 \prod_i \ell_i$, where the ℓ_i are small prime factors. The factor 4

ensures that $p \equiv 3 \bmod 4$ and that the special elliptic curve $E_0 : y^2 = x^3 + x$ is supersingular. Throughout this work, we assume that the class group $\mathrm{cl}(\mathcal{O})$ is known, enabling the transformation of arbitrary ideals into efficiently computable isogeny chains of degrees l_i using the relation lattice. We note that this is not a trivial assumption as current class group computations in reach fall short of realistic security levels [8, 11, 29] or lead to very slow protocols [17]. We point out, however, that there are polynomial-time quantum algorithms to this end [25]. We refer to [6, 8, 13, 37] for more details on the explicit computations of isogenies. For a more thorough introduction to isogenies and isogeny-based cryptography, we recommend [13, 16, 34].

Finally, we note that class groups are generally of composite order. By working in cyclic subgroups of $\mathrm{cl}(\mathcal{O})$ with generator $\mathfrak{g}$ and order $N \mid \#\mathrm{cl}(\mathcal{O})$, we can redefine the group action as $[\ ] : \mathbb{Z}_N \times \mathcal{E} \to \mathcal{E}$, where ideals of the form $\mathfrak{g}^a$ for $a \in \mathbb{Z}_N$ can be reduced modulo the relation lattice and efficiently computed. To work in a subgroup $\mathbb{Z}_{N'} \subset \mathbb{Z}_N$, we can simply use the generator $\mathfrak{g}^{N/N'}$. For the rest of this work, we always assume the choice of the subgroup $\mathbb{Z}_N$ to be such that $\{1, \ldots, n\}$ defines an exceptional set modulo N, i.e. that n is smaller than the smallest divisor of N.

(Structured) Piecewise Verifiable Proofs. We provide a brief overview of Piecewise Verifiable Proofs (PVPs) [7], as well as two available constructions that are utilized in the various threshold schemes from [1, 2, 7, 12]. PVPs are particular ZK proofs over secret shared data [10] that similarly consist of a collection of distributed relations $R_0, \ldots, R_n$, with the same witness space, where each statement can be verified independently. The goal of a PVP is to prove the existence of a witness w that satisfies $(x_i, w) \in R_i$ for every $i \in \{0, \ldots, n\}$, given a list of statements $x_0, \ldots, x_n$. The proof itself takes the form of $(\tilde{\pi}, \pi_i)_{i \in \{0,\ldots,n\}}$, where $(\tilde{\pi}, \pi_0)$ enables verification of x_0 in relation to R_0 (also known as the main or central proof), and π_i for $i \in \{1, \ldots, n\}$ enables verification of x_i in relation to R_i. Roughly speaking, PVPs [7] can be considered as a special case of NI-SZK proofs over shared data [10], when the proof piece $(\tilde{\pi}, \pi_0)$ associated to R_0 is named the central proof, and the $(\tilde{\pi}, \pi_i)_{i \in \{0,\ldots,n\}}$ are the proof pieces that are only relevant for $\{R_i\}_{i \in \{0,\ldots,n\}}$. In [2, 7], the authors have presented PVP schemes for the following list of n-distributed relations $R = (R_0, \ldots, R_n)$,

$$\begin{aligned} R_0 = \{((\Xi_k, F_1, F_1', \ldots, F_k, F_k'), f(x)) \mid (F_l' = [c_l f(0)] F_l)_{l=1}^k \}, \\ \forall i = 1, \ldots, n : R_i = \{(x_i, f(x)) | f(i) = x_i\} \end{aligned} \tag{3}$$

where the statement of R_0 consists of a public exceptional[3] set $\Xi_k = \{c_1 = 1, c_2, \ldots, c_k\}$ and a set of curves $(F_1, F_1', \ldots, F_k, F_k') \in \mathcal{E}^{2k}$, and the statements for the relations $\{R_i\}_{i=1,\cdots,n}$ are elements of $\mathbb{Z}_N$. The witnesses for both PVP schemes is a secret polynomial $f(X) \in \mathbb{Z}_N[X]_t$, which is a polynomial in the variable X with coefficients defined over $\mathbb{Z}_N$ and have a maximum degree of t.

[3] We can use superexceptional sets in the case, where all $F_1 = \cdots = F_k = E_0$, where $E_0 : y^2 = x^3 + x$, see [3].

Prover: Given, a witness polynomial $f(X) \in \mathbb{Z}_N[X]_t$ and a statement $x = ((\Xi_k, F_1, F_1', \ldots, F_k, F_k'), x_1, \cdots, x_n))$, outputs a non-interactive piecewise proof π of the relations in equation (3).

1. Parse $\Xi_k = \{c_1, c_2, \ldots, c_k\}$.
2. For $j = 1, \ldots, \lambda$: sample $b_j \leftarrow \mathbb{Z}_N[X]_t$ and compute $\hat{F}_j^1 = [c_1 b_j(0)]F_1 \ , \ldots, \ \hat{F}_j^k = [c_k b_j(0)]F_k$
3. Sample $y_0, y_0' \leftarrow \{0,1\}^\lambda$ uniformly at random;
4. Set $\mathsf{C}_0 = \mathcal{C}(\hat{F}_1^1, \cdots, \hat{F}_1^k \parallel \cdots \parallel \hat{F}_\lambda^1, \ldots, \hat{F}_\lambda^k, y_0)$,
5. Set $\mathsf{C}_0' = \mathcal{C}(F_1, F_1' \parallel \cdots \parallel F_k, F_k', y_0')$.
6. For $i = 1, \ldots, n$: sample $y_i, y_i' \leftarrow \{0,1\}^\lambda$ and set $\mathsf{C}_i = \mathcal{C}(b_1(i) \parallel \cdots \parallel b_\lambda(i), y_i)$ and $\mathsf{C}_i' = \mathcal{C}(x_i, y_i')$;
7. $\mathbf{d} = d_1 \ldots d_\lambda = \mathcal{H}(\mathsf{C}, \mathsf{C}')$, where $\mathsf{C} = (\mathsf{C}_0, \ldots, \mathsf{C}_n), \mathsf{C}' = (\mathsf{C}_0', \ldots, \mathsf{C}_n')$;
8. For $j = 1, \ldots, \lambda$: compute $r_j(x) = b_j(x) - d_j f(x) \mod N$
9. Return $\tilde{\pi} = (\mathsf{C}, \mathsf{C}', \mathbf{r})$ and $\{\pi_i = (y_i, y_i')\}_{i=0}^n$, where $\mathbf{r} = (r_1, \ldots, r_\lambda)$.

Verification: Given, an index $i \in \{0, \ldots, n\}$, a statement piece x_i of the form $x_0 = (\Xi_k, F_1, F_1', \cdots, F_k, F_k')$ if $i = 0$, or $x_i \in \mathbb{Z}_N$ if $i \neq 0$, as well as a proof piece $(\widetilde{\pi}, \pi_i) = ((\mathsf{C}, \mathsf{C}', \mathbf{r}), (y_i, y_i'))$, outputs `true` or `false`.

1. If $\mathsf{C}_i' \neq \mathcal{C}(x_i, y_i')$, then return `false`
2. $d_1 \ldots d_\lambda = \mathcal{H}(\mathsf{C}, \mathsf{C}')$;
3. If $i == 0$:
   - For $j = 1, \ldots, \lambda$:
     - If $d_j == 0$: $\widetilde{F}_j^1 \leftarrow [c_1 r_j(0)]F_1, \ \ldots \ , \widetilde{F}_j^k \leftarrow [c_k r_j(0)]F_k$,
     - else $\widetilde{F}_j^1 \leftarrow [c_1 r_j(0)]F_1', \ \ldots \ , \widetilde{F}_j^k \leftarrow [c_k r_j(0)]F_k'$
   - return $\mathsf{C}_0 == \mathcal{C}(\widetilde{F}_1^1, \cdots, \widetilde{F}_1^k \parallel \cdots \parallel \widetilde{F}_\lambda^1, \cdots, \widetilde{F}_\lambda^k, y_0)$
4. Else, return $\mathsf{C}_i == \mathcal{C}(r_1(i) + d_1 x_i \parallel \cdots \parallel r_\lambda(i) + d_\lambda x_i, y_i)$

Fig. 1. The prover and verification algorithms of the PVP schemes proposed in [7] (when $k = 1$, i.e., $\Xi_1 = \{c_1 = 1\}$) and [2].

In fact, the PVP scheme proposed in [7], is a special case of the PVP scheme proposed in [2], when $k = 1$, i.e., $\Xi_1 = \{c_1 = 1\}$ [4].

Fig. 1 describes the prover and verification algorithms of the PVP schemes proposed in [2, 7], where $\mathcal{H} : \{0,1\}^* \to \{0,1\}^\lambda$ is a random oracle and $\mathcal{C} : \{0,1\}^* \times \{0,1\}^\lambda \to \{0,1\}^{2\lambda}$ is a commitment scheme that is collapsing [35, Definition 12] and quantum computationally hiding [7, Definition 2]. In [2, 7], the authors show that their PVP schemes are correct, sound (against the prover and t malicious verifiers) and ZK (i.e., SZK, ZK against t malicious verifiers), for the relation in equation (3) Their PVP schemes [2, 7] have a binary/ternary challenge space, and allow a prover to convince a set of verifiers that she knows the secret element $f(0) \in \mathbb{Z}_N$ that connects a pair or a set of elliptic curves with correct factors through the group action $[f(0)]$, and where each verifier has a share of $f(0)$.

[4] Note that a PVP scheme is a NI-SZK proof for its underlying n-distributed relation, and actually in both PVP schemes [2, 7], the authors implicitly prove the SZK and soundness against the prover and t malicious verifiers, but do not call them as such.

Robust DKG Protocols for CSIDH. DKG protocols allow a group of parties to generate a secret and public key pair in a fully distributed manner. The property of robustness implies that this key pair generation always succeeds, even in the presence of malicious parties that can behave arbitrarily. In [7], Beullens, Disson, Pedersen, and Vercauteren constructed the first robust DKG protocol for CSIDH, called CSI-RAShi, that allows a set of parties to sample public key $F_1 = [x_0]E_0$ in a distributed manner, with x_0 being their Shamir secret shared value. CSI-RAShi works in the majority honest setting and consists of two steps. In the first step, each party P_i samples a polynomial $f_i(X) \in \mathbb{Z}_N[X]_t$, where $t \leq \lfloor \frac{n-1}{2} \rfloor$, and publishes $F_i = [f_i(0)]F_0$, while sharing the values $f_i(j)$ among the other parties $\{P_j\}_{j\in[n]}$. To ensure the correctness of the shared values, parties publish a PVP (i.e., a NI-SZK proof) for the relations $R_0, \ldots, R_n$ as described earlier (in Fig. 1 for $k = 1$, i.e., $\Xi_1 = \{c_1 = 1\}$). Then, each other party P_j verifies "their" relation R_j as well as the main relation R_0. In an honest-majority setting, if all parties agree that their shares are correct, they can be certain that each one of them possesses a share of a unique polynomial $f_i(X)$ of degree t, whose evaluation in 0 is in committed in $F_i = [f_i(0)]F_0$. In the next step, parties use their committed values and in a round-robin manner compute the final public key $[x_0]E_0$ and attach a ZK proof for their correct action (i.e., updating the PK correctly).

In follow-up work [1], Atapoor, Baghery, Cozzo, and Pedersen presented a new variant of the CSI-RAShi DKG protocol (so-called Structured CSI-RAShi), that allows a set of parties to sample a structured public key (SPK), i.e. a public key of the type $\{[c_i x_0]E_0\}_{i=1}^{k}$, where $\Xi_k := \{c_1 = 1, c_2, \cdots, c_k\}$ is an exceptional set, in a distributed manner. Structured CSI-RAShi also works in the majority-honest setting and allows a set of parties to sample an SPK at least $3\times$ faster than the case that one uses the extended and optimized version of CSI-RAShi [1]. Similar to CSI-RAShi, the structured CSI-RAShi also consists of two phases. Its first phase is the same as in the basic CSI-RAShi, but in the second phase, parties use multiples of their committed values $f_i(0)$ and in a round-robin manner compute the final SPK and attach a ZK proof for their correct action (i.e., updating the SPK correctly). Using multiples of the same secret allows to summarize multiple related statements into a single ZK proof.

In a different work [2], the same authors presented two new DKG protocols for CSIDH-based primitives, but using the BGW VSS scheme [4]. The latter uses bivariate polynomials and is very efficient and achieves information-theoretical security, but it needs at least 2/3 of the parties to be honest and also requires interaction between the verifiers (shareholders) to check the validity of the shares. Then, in the second phase of their DKG protocols, parties use their secret shared value from the previous phase, namely $f_i(0)$, and engage in a round-robin public key computation protocol. In order to prove correct action, parties can now use PVPs instead of ZK proofs. The authors propose two protocols, one using (extended) public keys, and one using SPKs. For the latter, the structured PVPs from Fig. 1 are used. Fig. 2 summarizes both their DKG protocols and highlights the differences. Note that the first phases (the VSS) are identical.

Verifiable Secret Sharing (BGW VSS) [4]:

1. For $i = 1, \ldots, n$, player P_i
   (a) samples $q^{(i)}(Z) \leftarrow \mathbb{Z}_N[Z]_t$ and sets $x^{(i)} = q^{(i)}(0)$,
   (b) samples $S^{(i)}(X, Y) \leftarrow \mathbb{Z}_N[X, Y]_t$ with $S^{(i)}(0, Z) = q^{(i)}(Z)$,
   (c) for $j = 1, \ldots, n$, defines $f_j^{(i)}(X) = S^{(i)}(X, j)$ and $g_j^{(i)}(Y) = S^{(i)}(j, Y)$ and sends $\{f_j^{(i)}(X), g_j^{(i)}(Y)\}$ privately to party P_j.
2. For $k = 1, \ldots, n$, each pair of players P_i, P_j checks that $f_i^{(k)}(j) = g_j^{(k)}(i)$ and $g_i^{(k)}(j) = f_j^{(k)}(i)$. Whenever one of these checks fails, the concerned player runs the conflict resolution procedure of BGW VSS scheme (reviewed in [2, Section 3.1]. In case the procedure outputs $\perp$, the dealer P_k of the concerned polynomials is disqualified, otherwise the protocol continues normally.
3. In the end, all the honest players agree on the same set of qualified players $Q \subseteq \{1, \ldots, n\}$, and the shared secret key x is given as the sum of the individual secrets of the qualified players $x = \sum_{i \in Q} x^{(i)}$, while the parties' shares of x can be constructed as $x_j = \sum_{i \in Q} f_j^{(i)}(0)$.

Computing the Single/Structured Public Key:

4. Let $Q = \{1, \ldots, n'\}$. Given a superexceptional set $\Xi_k = \{c_1 = 1, c_2, \ldots, c_k\}$, a qualified set of parties engage in a round-robin protocol, and party P_i computes
$$F_i^1 \leftarrow [x^{(i)}]F_{i-1}^1, \; F_i^2 \leftarrow [c_2 x^{(i)}]F_{i-1}^2, \quad \ldots, \quad F_i^k \leftarrow [c_k x^{(i)}]F_{i-1}^k,$$
where $F_0^1 = F_0^2 = \cdots = F_0^k = E_0$. At each step, party P_i also uses the Structured PVP scheme given in Fig. 1, and creates and publishes a PVP proof,
$$\pi^{(i)} = (\tilde{\pi}^{(i)}, \pi_1^{(i)}, \ldots, \pi_{n'}^{(i)}) \leftarrow \mathbf{Prover}\big(\Xi_k, (F_{i-1}^1, F_i^1, \cdots, F_{i-1}^k, F_i^k); q^{(i)}(Z)\big)$$
which includes a main proof $(\tilde{\pi}^{(i)}, \pi_0^{(i)})$ as well as individual proof pieces $\pi_j^{(i)}$ for each other player P_j. Note that in a single public key generation, they run the PVP scheme in Fig. 1 for $k = 1$, i.e., $\Xi_1 = \{c_1 = 1\}$.
5. Each other player P_j verifies both $\mathbf{Prover}(j, f_j^{(i)}(0), \tilde{\pi}^{(i)}, \pi_j^{(i)})$ and $\mathbf{Verifier}(0, (\Xi_k, (F_{i-1}^1, F_i^1, \cdots, F_{i-1}^k, F_i^k)), \tilde{\pi}^{(i)}, \pi_0^{(i)})$. Whenever a verification of $\pi^{(i)}$ fails, the verifier P_j broadcasts $f_j^{(i)}(X)$. All other parties verify the correctness of $f_j^{(i)}(X)$ as in step 2. If it is correct, since there are at least t honest players, they will be able to reconstruct $q^{(i)}(0)$, compute F_i and proceed with the protocol (and potentially disqualify P_i). Otherwise, if the checks of $f_j^{(i)}(0)$ fail, the complaint can be ignored (or P_j disqualified). In the latter case, the shares of P_j can also be reconstructed by the at least t honest players.
6. At the end of the round-robin, the parties return the *structured* public key
$$F_{n'}^1 = [x_0]E_0, \; F_{n'}^2 = [c_2 x_0]E_0, \; \cdots, F_{n'}^k = [c_k x_0]E_0 \,.$$

Fig. 2. The DKG protocols of Atapoor, Baghery, Cozzo, and Pedersen for a single [2, Section 3] and an SPK generation [2, Section 4]. Both schemes use the BGW VSS [4] for secret sharing. The highlighted terms are particular for single PK computation.

2.4 Threshold Signatures

A threshold signature scheme enables a group of authorized parties to collectively sign a message m, generating a signature σ that can be verified using a single

public key pk. Specifically, a threshold signature scheme, in terms of an $(t+1, n)$-threshold access structure, is defined as follows:

Definition 2.7. *A* threshold digital signature scheme *consists of three probabilistic algorithms:* KeyGen*,* Sign*, and* Verify*.*

- $\mathsf{KeyGen}\left(1^{\lambda}\right)$*: Given the security parameter as input and returns the public key* pk *along with a set of secret keys* sk_i *- one secret key per party. (For simplicity, we limit ourselves to the case where each party has a single share of the secret, focusing on Shamir and full-threshold secret sharing.)*
- $\mathsf{Sign}\left(\{\mathsf{sk}_i\}_{i\in Q}, m\right)$*: Given as input a* qualified set *of private keys and a message and returns a signature on the message.*
- $\mathsf{Verify}\left(\mathsf{pk}, (\sigma, m)\right)$*: Given* pk *and a signature* σ *on a message* m*, and outputs a bit that is equal to one if and only if the signature on* m *is valid.*

In essence, security for a threshold signature scheme means that an unqualified group of parties cannot forge a signature on a new message. In addition, for distributed signatures, we require that a valid output signature is indistinguishable from the signature produced by the signing algorithm of the underlying non-thresholdized scheme with the same public key.

3 VSS from ZK Proofs Over Shared Data

In this section, we propose a novel Non-Interactive Verifiable Secret Sharing (NI-VSS) scheme that utilizes ZK proofs over secret shared data [7, 10] to prove the validity and consistency of the individual shares. The proposed scheme does not rely on a concrete cryptographic hard problem, rather than a random oracle and a collapsing (quantum) computationally hiding commitment scheme.

To build the NI-VSS scheme, we first construct a non-interactive proof scheme which allows a single prover to convince a set of verifiers that they have each received a distinct evaluation of a polynomial $f(X) \in \mathbb{Z}_N[X]_t$.[5] It is worth noting that to achieve soundness, the number of honest verifiers is supposed to exceed t. On the other hand, to achieve strong zero-knowledge, we assume that an adversarial prover can corrupt at most t verifiers. Thus, we assume the number of verifiers to be greater than or equal to $2t + 1$. We then demonstrate that our proposed scheme satisfies completeness (Def. 2.4), soundness against the prover and t malicious verifiers (Def. 2.5), and strong ZK (Def. 2.6). We subsequently use the resulting Non-Interactive Strong ZK (NI-SZK) proof scheme and build an efficient NI-VSS scheme based on Shamir secret sharing.

3.1 A NI-SZK Proof Protocol for Shamir Secret Sharing

As the key building block for our novel NI-VSS scheme, in this section, we present an efficient NI-SZK proof scheme that can be used to build a NI-VSS

[5] In general $\mathbb{Z}_N$ will constitute a ring. In later applications, we sometimes choose N to be a prime, so that $\mathbb{Z}_N$ becomes a field.

Prover: Given, a witness polynomial $f(X) \in \mathbb{Z}_N[X]_t$, an input $x = (x_1, \cdots, x_n)$, proceed as follows and output a proof π of the relations in eq. (4).

1. Sample $b(X) \leftarrow \mathbb{Z}_N[X]_t$ uniformly at random;
2. For $i = 1, \ldots, n$: Sample $y_i, y_i' \leftarrow \{0,1\}^\lambda$ uniformly at random; Set $\mathsf{C}_i \leftarrow \mathcal{C}(b(i), y_i)$ and $\mathsf{C}_i' \leftarrow \mathcal{C}(x_i, y_i')$;
3. Set $d \leftarrow \mathcal{H}(\mathsf{C}, \mathsf{C}')$, where $\mathsf{C} = (\mathsf{C}_1, \ldots, \mathsf{C}_n), \mathsf{C}' = (\mathsf{C}_1', \ldots, \mathsf{C}_n')$;
4. Set $r(X) \leftarrow b(X) - d \cdot f(X) \mod N$;
5. Set $\pi := (\mathsf{C}, \mathsf{C}', r(X), \{\pi_i\}_{i=1}^n)$, where $\pi_i = (y_i, y_i')$;
6. Publish $(\mathsf{C}, \mathsf{C}', r(X))$; Send individual proof $\{\pi_i = (y_i, y_i')\}_{i=1}^n$ to verifier V_i.

Verification: For $i = 1, \cdots, n$, each verifier (shareholder) i has a statement $x_i \in \mathbb{Z}_N$, and a proof $((\mathsf{C}, \mathsf{C}', r(X)), (y_i, y_i'))$. Given the set of statements and proofs for $i \in 1, \ldots, n$ the verifiers (i.e., shareholders) proceed as follows:

1. Verifier i acts as below and outputs `true` or `false`.
   (a) If $\mathsf{C}_i' \neq \mathcal{C}(x_i, y_i')$ return `false`;
   (b) Set $d \leftarrow \mathcal{H}(\mathsf{C}, \mathsf{C}')$;
   (c) If $\mathsf{C}_i == \mathcal{C}(r(i) + d \cdot x_i, y_i)$ return `true`; otherwise `false`;
2. Return `true` if *all* the verifiers return `true`; otherwise returns `false`.

Fig. 3. A NI-SZK Proof Scheme for Shamir Secret Sharing.

based on Shamir secret sharing. The new NI-SZK proof scheme is built for a collection of relations $R_1, \ldots, R_n$ with the same witness space, where each statement can be verified independently by individual verifiers. Given the shared input $x = x_1 \parallel x_2 \parallel ... \parallel x_n$, the prover proves the existence of a witness w that satisfies $(x_i, w) \in R_i$ for every $i \in 1, \ldots, n$. The proof includes proof pieces $\{\pi_i\}_{i \in 1, \ldots, n}$, where π_i allows the verifier V_i to check the validity of x_i in relation to R_i. The prover has a secret polynomial $f(X) \in \mathbb{Z}_N[X]_t$, and wants to prove the following n-distributed relations,

$$R_i = \{(x_i, f(X)) | f(i) = x_i\}, \tag{4}$$

where $i = 1, \ldots, n$. For the sake of convenience, we will refer to the relation mentioned above as the Shamir relation throughout the rest of the paper.

Fig. 3 describes the proof generation and verification of the new NI-SZK proof scheme for the Shamir relation given in equation (4), where $\mathcal{H} : \{0,1\}^* \to \Xi_k$ is a random oracle with Ξ_k an exceptional set of size k,[6] and $\mathcal{C} : \{0,1\}^* \times \{0,1\}^\lambda \to \{0,1\}^{2\lambda}$ is a commitment scheme that is collapsing [35, Def. 12] and quantum computationally hiding. Next, we show the proposed NI-SZK proof scheme (given in Fig. 3) satisfies the key security requirements of a ZK proof protocol over shared data, as defined in Sec. 2.1.

Remark 3.1. The challenge space of the protocol in Fig. 3 is $|\Xi_k| = k$. When $\mathbb{Z}_N$ is a cryptographically sized field, we can easily choose $\Xi_k = \mathbb{Z}_N$ to achieve a negligible soundness error, i.e. below $2^{-\lambda}$. In the case where $\mathbb{Z}_N$ is a ring, we might have the case that the largest exceptional set has size $k < 2^\lambda$. In that case the protocol from Fig. 3 would have to be repeated in the standard fashion:

[6] Such a hash function can easily be implemented by hashing into a set $\{1, \ldots, k\}$ and then using the output value $i \in \{1, \ldots, k\}$ as an index in Ξ_k, i.e. $c_i \in \Xi_k$.

defining $S = \lceil \lambda / \log k \rceil$, we would have to sample S different $b_j(X)$ and construct S responses $r_j(X) = b_j(X) - d_j f(X)$, sampling the d_j using the hash function $\mathcal{H} : \{0,1\}^* \to (\Xi_k)^S$.

Theorem 3.1 (NI-SZK Proof Scheme for Shamir Secret Shares). *Let L be an n-distributed language for the list of relations given in equation (4), $t \geq 0$ be a security threshold such that $n \geq 2t+1$, and $h = n - t$. Assuming that the commitment scheme $\mathcal{C}$ is collapsing and quantum computationally hiding, for any potential set $I \subseteq [n]$ of size $|I| \geq h$, the protocol described in Fig. 3 is a non-interactive distributed strong ZK protocol for L that satisfies completeness, strong ZK, and soundness against the prover and t malicious verifiers in the QROM.*

Proof. The proof is given in App. A. □

3.2 A NI-VSS Scheme from NI-SZK Proofs

Next, we use the NI-SZK proof scheme proposed in the last subsection and construct a NI-VSS scheme based on Shamir secret sharing. Our scheme operates on the assumption that each shareholder has an authenticated communication channel with the dealer, which can be achieved through a public key infrastructure. Therefore, the shares will only be hidden computationally. The proposed scheme works in the majority honest setting, and the validity of secret shares cannot be publicly verified (as in [4]), and it requires (non-interactive) collaboration among the shareholders to verify them. We demonstrate later that this is sufficient in many Shmair-based threshold protocols (e.g. DKGs, threshold signatures, etc.) that also work in the majority-honest setting.

Our Definitions. Before going through the proposed construction to build a NI-VSS scheme, we review our formal definitions of VSS schemes which are a minimally modified version of the ones from previous works [28, 31].

Definition 3.1. *An (n, t, x_0) non-interactive VSS consists of four PPT Algorithms of (Initialization, Share, Verification, Reconstruction) as follows:*

1. *Initialization: In this phase, the system parameters are generated and shared with the parties.*
2. *Share$(n, t, x_0) \to (x_1, \cdots, x_n, \pi)$: Given the number of parties n, threshold t, and the secret x_0, the algorithm secret shares x_0 and outputs the shares $\{x_1, \cdots, x_n\}$ and a proof π to prove that it has done the sharing correctly.*
3. *Verification$(n, t, x_1, \cdots, x_n, \pi) \to$ `true/false`: Given the number of parties n, threshold t, and the shares $(x_1, \cdots, x_n)$ (or commitment of them), and the proof π, generated by Share, the algorithm outputs either `true` or `false`.*
4. *Reconstruction$(n, t, x_1, \cdots, x_{t+1}) \to x_0/\{$true/false$\}$: Given any $t+1$ of the shares, e.g., $\{x_1, \cdots, x_{t+1}\}$, it reconstructs and returns x_0. Alternatively, given a candidate value for x_0 (or a function of it) and $t+1$ of the shares, the algorithm confirms the validity of the candidate secret x_0 (or the function of it), and returns either $\{$true/false$\}$.*

A verifiable secret sharing scheme further has two requirements as follows [31].

- **Verifiability constraint**: A shareholder must be able to determine whether a share of the secret is valid or not. If it is valid, then Reconstruction should produce a unique secret x_0 when run on any $t+1$ distinct valid shares, or alternatively any $t+1$ shareholders should be able to check the validity of a potential x_0 or any publicly computable function of it.
- **Unpredictability**: The protocol must be unpredictable, meaning that there is no strategy for selecting t shares of the secret that would enable someone to predict the secret x_0 with a significant advantage.

We highlight that our definition of VSS differs slightly from current ones [28, 31]. Our definitions utilize a ZK proof scheme over secret shared data for proving the validity of the shares, ensures the existence of a polynomial-time verification algorithm that can validate the shares, and also introduces a novel approach for reconstruction of the main secret. The first two features already are implicitly built in any VSS scheme, however the third one is new in our framework. In our Reconstruction algorithm, in addition to enabling the reconstruction of the secret value x_0 by any $t+1$ shareholders, we also consider the scenario where $t+1$ shareholders can validate the validity of a secret disclosed by the dealer or validate the correctness of a computation performed on x_0. Later, we show that this is natural in practice, and usually the shareholders do not reconstruct the plain value of x_0. Instead, each act as a dealer once and later use their shared secret and perform some computations and give a ZK proof for their correct action. In some cases, the proofs might be verifiable only by the shareholders.

Our Construction. In the VSS scheme, a dealer wants to distribute shares of a secret x_0 among n parties $P_1, \ldots, P_n$. Such that depending on the underlying access structure, a subset of shareholders are qualified to recover the secret value x_0. In our case, which is based on Shamir secret sharing, the secret can be recovered by any subset of more than t shareholders, where $t < n$. On the other hand, any subset of size $\leq t$ will not gain any information about x_0, unless the security of underlying NI-SZK proof scheme is broken. The complexity of our VSS scheme is linear in the security parameter and also linear in the number of shareholders which is essentially optimal. It achieves computational security, which is proven in the (Q)ROM, using a secure commitment scheme. We present our protocol in Fig. 4.

We note that in the *Reconstruction* phase of new schemes, unlike other approaches, the parties do not perform any decryption or proof generation to show the correctness of their actions. Instead, the dealer calculates and publishes the reconstructed secret value $f(0) = x_0$ (or a function thereof) along with a NI-SZK proof for the distributed relation $R_i = \{(x_i, f(X))|f(i) = x_i\}$, for $i = 0, 1, \ldots, n$. Then, any $t+1$ shareholders utilize their secret shares to verify the validity of the disclosed secret value $x_0 = f(0)$ (or a function of it, such as $x_0 = F(f(0))$). If the verification process returns `true`, this confirms that the disclosed value $f(0)$ or $F(f(0))$ (e.g., $g^{f(0)}$ in the case of DKG for Schnorr signature) represents the main secret or a function thereof. Note that as in other cases, if we have

Initialization: Parties $P_1, \cdots, P_n$ generate system parameters and each one registers a PK to facilitate secure communications.

Share: Given n and t, to share x_0, the dealer proceeds as follows:

1. Sample a uniformly random polynomial $f(X)$ of degree t with coefficients in a ring R, subject to $f(0) = x_0$.
2. For $i = 1, 2, \cdots, n$: set $x_i := f(i)$.
3. Given $f(X)$ and $x = (x_1, \cdots, x_n)$, run the prover of NI-SZK scheme in Fig. 3, and obtain the proof $\pi := (\mathsf{C}, \mathsf{C}', r(X), \{\pi_i\}_{i=1}^n)$.
4. Send the share and the individual proof (x_i, π_i) privately to party P_i and broadcast the elements $(\mathsf{C}, \mathsf{C}', r(X))$ as the proof.

Verification: To verify the received shares, $P_1, \cdots, P_n$ utilize their shares $\{x_i\}_{i=1}^n$ and run the verifier of the NI-SZK proof scheme given of Fig. 3. If the verification of P_i fails, then P_i broadcasts a complain against the dealer. If more than t shareholders complain against the dealer, then the *Verification* returns `false`. If P_i complains that his part of proof does not verify, the dealer broadcasts $\pi_i = (y_i, y_i')$ so that everyone can verify it using the verification algorithm of the NI-SZK scheme. If it passed the verification, the protocol continues as normal, otherwise the parties disqualify the dealer and *Verification* returns `false`. Since disqualifying the dealer or parties happens on the basis of only broadcasted information, at the end all the honest shareholders will agree on the same set of qualified parties $Q \subseteq \{1, 2, \cdots, n\}$ or will reject the final verification. At the end, if the verification returns `true`, all honest shareholders are sure that they have received a valid share of $x_0 = f(0)$, and any subset of size larger than t of them can retrieve the secret $x_0 = f(0)$.

Reconstruction: Proceed as follows:

1. Given the secret $f(X)$, the dealer computes (reconstructs) $x_0 = f(0)$.
2. Using $f(X)$ and $x = (x_0, x_1, \cdots, x_n)$, run the prover of the NI-SZK scheme in Fig. 3, to prove that $f(0) = x_0$ and $f(i) = x_i$ for $i = 1, \ldots, n$, and obtain $(x_0, y_0, \mathsf{C}, \mathsf{C}', r(X), \{\pi_i\}_{i=1}^n)$. A slightly modified version of the NI-SZK scheme allows the dealer to convince each party that their shares come from a polynomial of degree t with free term $x_0 = f(0)$.
3. Send the individual proof π_i privately to party P_i, and broadcast the elements $(x_0, y_0, \mathsf{C}, \mathsf{C}', r(X))$.
4. Each shareholder P_i has values x_0, y_0, x_i, and a proof $((\mathsf{C} := (\mathsf{C}_0, \mathsf{C}_1, \cdots, \mathsf{C}_n), \mathsf{C}' := (\mathsf{C}_1', \cdots, \mathsf{C}_n'), r(X)), (y_0, y_1, \cdots, y_n, y_1', \cdots, y_n'))$. Given the set of statements and proofs, the shareholders run the verification of the NI-SZK scheme in Fig. 3 and return either `true` or `false`. Note that in this case, in the last step of the verification, the shareholder P_i does an additional check and verifies if $\mathsf{C}_0 == \mathcal{C}(r(0)+d\cdot x_0, y_0) \wedge \mathsf{C}_i == \mathcal{C}(r(i)+d\cdot x_i, y_i)$.
5. At the end, the algorithm **return** `true` if *all* the shareholders return `true`; otherwise **return** `false`. Returning `true`, confirms that the value x_0 is the reconstruction of the main secret value $f(0)$.

Fig. 4. The proposed NI-VSS scheme.

$t+1$ shareholders, we only can achieve a reconstruction with *abort*, while with n shareholders we can have a *robust* reconstruction phase. It's worth noting that as with other VSS schemes, where any subset of qualified sets can recover the secret value, in this case, any qualified subset of the shareholders can validate

the validity of the revealed secret and the proof. Conversely, any group of parties that is smaller than the threshold value will be unable to verify the soundness of the ZK proof, and they cannot confirm that the disclosed secret (or function thereof) is valid or has been correctly computed using the main secret.

In practical distributed protocols, the dealer typically does not reveal the main secret $f(0)$ but instead discloses the outcome of certain computations that have been performed with it, such as $y = g^{f(0)}$ in the case of DKG for the Schnorr signature. In these cases, the dealer must provide proof that the computation was carried out using $f(0)$, for example by demonstrating that $y = g^{f(0)} \wedge f(i) = x_i$ in the context of DKG for the Schnorr signature. At first glance, this may seem unconventional. However, as we will show later, it is actually (sufficient and) common practice in many current (actively secure) threshold protocols, e.g., DKGs, threshold signatures and distributed commitments. In the upcoming section, we will explore some practical examples and types of NI-SZK proof systems that one would need in the *Reconstruction* phase of such protocols.

Theorem 3.2 (VSS from NI-SZK Proof Schemes). *If the proof scheme given in Fig. 3 is a secure NI-SZK protocol for the relations in equation (4), then, the non-interactive VSS scheme (given in Fig. 4) is secure. That is, (i) the Reconstruction protocol results in the secret distributed by the dealer for any qualified set of shareholders, (ii) any non-qualified set of shareholders is unable to recover the secret.*

Proof. The prove is given in App. B. □

Efficiency. We examine the empirical performance of new NI-VSS scheme in the follow-up subsection. Furthermore, we describe its asymptotic efficiency along with some applications in the next section.

3.3 Empirical Performance

To evaluate the empirical performance of the new VSS scheme, we implemented it in C++, using the libraries `libsodium` and `NTL`. To instantiate the commitment scheme and the RO, we used a SHA256 hash function. To evaluate its performance, we conducted experiments where we varied the number of parties and the threshold. Specifically, we report the run times of the *Sharing* and *Verification* phases for different numbers of parties, n, and thresholds, t.

To conduct these experiments, we ran our code on a desktop machine with Ubuntu 20.4.4 LTS, an Intel Core i9-9900 processor at base frequency 3.1 GHz, and 128GB of memory. All the operations in sharing, proof generation and verification are done in a single-thread mode. The software configurations included `libsodium` 1.0.18, `NTL` version 11.5.1, and `GMP` version 6.2.1. The random oracle $\mathcal{H}$ and commitment scheme $\mathcal{C}$ are instantiated using SHA256 hash function.

The performance results with different number of parties and threshold values, ranging from $(n, t) = (128, 62)$ to $(n, t) = (16384, 8190)$ are summarized in Table 2. For the *Sharing* phase, we report the time required for secret sharing

Table 2. Empirical performance of the NI-VSS scheme for various numbers of parties and threshold values (n, t). All the reported timings are in either milliseconds (msec) or microseconds (μsec). In the Sharing phase, we separately report the time spent on secret sharing, committing to the shares, and generating the proofs. Furthermore, we specify the total RAM consumption in Gigabytes, and also the verification time for each individual shareholder. Ideally, the total verification time would be close to the verification time of an individual shareholder.

(n, t)	Sharing Phase (msec)				Single Party Verification	RAM Usage (Gigabyte)
	Secret Sharing	Commit to Shares	Proof Generation	Total Time		
$(128, 62)$	< 1	< 1	< 1	< 1	38 μsec	2.10
$(256, 126)$	2	< 1	3	≈ 5	71 μsec	2.35
$(512, 254)$	11	< 1	12	≈ 24	137 μsec	2.43
$(1024, 510)$	46	< 1	47	94	268 μsec	2.56
$(2048, 1022)$	185	1.45	186	373	532 μsec	2.95
$(4096, 2046)$	753	2.91	758	1513	1.08 msec	4.41
$(8192, 4094)$	3025	5.78	3035	6065	2.16 msec	9.97
$(16384, 8190)$	12039	11.6	12067	24118	4.33 msec	32.0

and proof generation separately. We also specify the total RAM consumption in Gigabytes. As it can be seen, both Sharing and Verification phases are very fast and practical, even for larger numbers of parties and higher thresholds. This is mainly because of using lightweight operations, namely hashing and polynomial evaluation, in the underlying NI-SZK proof scheme. It is worth noting that our implementation is still quite naive, operating single-threaded and without concrete optimizations. This shows the practicality and scalability of new NI-VSS scheme for deployment in various (post-quantum secure) threshold protocols.

4 More Efficient DKG Protocols for CSIDH

We demonstrate the potential of the new NI-VSS scheme by revisiting several threshold protocols and improving their efficiency.

In this section, we revisit two new DKG protocols that have recently been proposed by Atapoor, Baghery, Cozzo and Pedersen [2] for CSIDH-based primitives. Their DKG protocols [2] can be considered variants of the DKG protocols CSI-RAShi [1, 7] and Structured CSI-RAShi [1].

The DKG protocols proposed in [2] are computationally secure and consist of two stages: an IT secure VSS step and a computationally secure (public key) computation step. During the VSS step, the parties engage in the BGW VSS scheme [4] and share a secret x_0 (i.e., the secret key) among themselves. Then, in the (public key) computation step, they use their shares obtained from the first step and compute the target public key (either $\{E_i = [x_i]E_0\}_{i=1}^{k}$, or $\{E_i = [c_i x_0]E_0\}_{i=1}^{k}$ for public integers $\{c_i\}_{i=1}^{k}$) in a round-robin fashion.

These protocols have reduced the computational complexity in terms of isogeny computations, when compared to [1, 7] but achieve this at the cost of higher communication complexity, a reduced number of corrupted parties to $n/3$, and an interactive share verification in the final DKG protocols.

In this section, we show that by integrating our new NI-VSS scheme into the VSS step of their DKG protocols, we can resolve all of these drawbacks at the same time. Our protocols achieve lower communication than either of these protocols, allow $n/2$ corrupted parties and are non-interactively verifiable, while achieving the same computational complexity as the fastest protocols from [2].

For formal definitions of the security properties of DKGs, we refer to [7, 21].

4.1 CSI-RAShi++: A More Efficient DKG for CSIDH

The first DKG protocol proposed in [2, Section 3] is a new variant of CSI-RAShi DKG protocol [7] that requires less number of isogeny computations but at the cost of asymptomatically higher communication, interactive share verification, and a lower bound on the number of corrupted parties. Fig. 5 describes a new variant of their DKG protocol in [2, Section 3] that its BGW VSS scheme is replaced with our computationally secure NI-VSS scheme from Section 3. This replacement results in a reduction of IT security in the VSS step to computational security, but overall, as in the original case, the resulting DKG protocol achieves computational quantum security. The new DKG protocol can be seen as an improved version of the computationally secure DKG protocol CSI-RAShi that similarly works in a majority honest setting, while with reduced computation and communication costs. Note that this can also be extended to the larger public keys, as in [1].

Verifiable Secret Sharing Step: This is done using the NI-VSS scheme presented in Fig. 4 in a standard distributed manner. Namely, each party P_i one time plays the role of the dealer in Fig. 4, samples $f^{(i)}(X)$, and then in a verifiable manner shares $f^{(i)}(0)$ with other parties. In the end, all the shareholders get a share of the joint secret key x_0, where implicitly is defined as $x_0 = \sum_{i \in Q} f^{(i)}(0)$ for a qualified set Q. Each party P_j obtains its share of x_0 as $x_j = \sum_{i \in Q} f^{(i)}(j)$.

PK Computation Step: This is done as in the (public key) computation step of the DKG protocol presented in [2, Section 3], which is reviewed in Fig. 2. At the end, the parties return the public key $[x_0]E_0$.

Fig. 5. CSI-RAShi++: an efficient DKG protocol for a single PK $[x_0]E_0$.

Theorem 4.1 (CSI-RAShi++ DKG Protocol). *If the VSS scheme given in Fig. 4 is a secure verifiable secret sharing scheme in the QROM, then the DKG protocol of Fig. 5 is secure in the QROM. That is correct, robust, and satisfies the secrecy property.*

Proof. The proof is almost the same as the proof of [2, Theorem 3.5], except that it will rely on the security of the new NI-VSS, proven in Theorem 3.2, rather than the security of the BGW VSS scheme [4], used in their DKG protocol. □

4.2 Structured CSI-RAShi++

Recall that an SPK takes the form $\{E_i = [c_i x_0]E_0\}_{i=1}^k$ for a given secret $x_0 \in \mathbb{Z}_N$, and a set of public distinct coefficients c_i, sampled from a public (super-)exceptional set Ξ_k [3]. The usefulness of SPKs has recently been demonstrated in the construction of more efficient primitives in the CSIDH setting [1, 3].

The second DKG protocol proposed in [2, Section 4] is a new variant of Structured CSI-RAShi DKG protocol [1] that needs fewer isogeny computations but similar to the previous case at the cost of asymptomatically higher communication, interactive share verification, and a lower bound on the number of corrupted parties.

In Fig. 6, we present a new variant of their second DKG protocol in [2, Section 4], that can be used to generate an SPK in a distributed manner. Again, we replace the BGW VSS scheme used in their protocol with our NI-VSS scheme. This results in a computationally secure DKG protocol for SPKs. Similar to CSI-RAShi++, our second DKG protocol can be seen as an improved version of Structured CSI-RAShi [1], which requires less computation and communication.

Verifiable Secret Sharing Step: This is done using the NI-VSS scheme (given in Fig. 4) as described in Fig. 5. In the end, all the shareholders get a share of the joint secret key x_0, where implicitly is defined as $x_0 = \sum_{i \in Q} f^{(i)}(0)$ for a qualified set Q. Each party P_j obtains its share of x_0 as $x_j = \sum_{i \in Q} f^{(i)}(j)$.

SPK Computation Step: This is done as in the structured public key computation step of the DKG protocol presented in [2, Section 4], which is reviewed in Fig. 2. At the end, the parties return the structured public key $\{E_i = [c_i x_0]E_0\}_{i=1}^k$.

Fig. 6. Structured CSI-RAShi++: an efficient DKG protocol for a structured public key $\{E_i = [c_i x_0]E_0\}_{i=1}^k$.

Theorem 4.2 (Structured CSI-RAShi++ DKG Protocol). *If the VSS scheme given in Fig. 4 is a secure verifiable secret sharing scheme in the QROM, then the DKG protocol of Fig. 6 is secure in the QROM. That is correct, robust, and satisfies the secrecy property.*

Proof. The proof is almost identical to the proof of [2, Theorem 4.1], except that in this case we will rely on the security of the new NI-VSS, proven in Theorem 3.2, rather than the security of the BGW VSS scheme [4] which is employed in the secret sharing step of their DKG protocol. □

4.3 Efficiency of the Revised DKG Protocols

In Tables 3 and 4, we summarize the computational and communication costs of our proposed DKG protocols, CSI-RASHI++ (from Fig. 5) and Structured CSI-RAShi++ (from Fig. 6), and compare them with current DKG protocols in the CSIDH setting. To have a fair comparison we express the computational cost as the sequential runtime of the protocol steps, i.e. the total runtime from

Table 3. Sequential computational costs (including idle time) of the different DKGs from [2] and from this work, in terms of polynomial evaluations, isogeny computations and calls to the commitment scheme and random oracle. For compactness, we assume $\gcd(k,n) = \min\{k,n\}$ and do not explicitly write down the gains through the twist trick. See [2] for more details.

	Polynomial Eval.	Isogenies	Commitments	RO queries
Basic DKG [2]	$2(n-1)^2 + n\lambda(n+2)$	$2n\lambda + n$	$2n(n+3)$	$2n$
Extended DKG [1,2]	$2(n-1)^2k + n\lambda(n\lceil\frac{k}{n}\rceil + k)$	$n(n\lambda+1)\lfloor\frac{k}{n}\rfloor$	$2n((n-1)\lceil\frac{k}{n}\rceil + 2k)$	nk
Structured DKG [2]	$2(n-1)^2 + n\lambda(2n+1)$	$n(n\lambda+1)\lfloor\frac{k}{n}\rfloor$	$2n(3n-1)$	n^2
Our Basic DKG (Sec. 4.1)	$(3n-1) + n\lambda(n+2)$	$2n\lambda + n$	$2n(n+5) - 2$	$3n$
Our Extended DKG (Sec. 4.1)	$(3n-1)k + n\lambda(n\lceil\frac{k}{n}\rceil + k)$	$n(n\lambda+1)\lfloor\frac{k}{n}\rfloor$	$2n((n-2)\lceil\frac{k}{n}\rceil + +4k) - 2k$	$2nk$
Our Structured DKG (Sec. 4.2)	$(3n-1) + 2n^2\lambda$	$n(n\lambda+1)\lfloor\frac{k}{n}\rfloor$	$2n(3n+1) - 2$	$n(n+1)$

Table 4. Communication costs of different DKGs from [2] and this work, in terms of elements in $\mathbb{Z}_N$ and $\mathcal{E}$, and the number of commitments and proof pieces (i.e. elements of size 2λ). The cost represents the outgoing cost per party. The cost of the basic DKG follows by setting $k = 1$.

	Element of Z_N	Element of $\mathcal{E}$	Commitment/Proof Piece
Extended DKG [1,2]	$2k(n-1)(n+t-1) + kn\lambda(t+1)$	nk	$nk(3n+2)$
Structured DKG [2]	$2(n-1)(n+t-1) + n\lambda(t+1)$	nk	$n(3n+2)$
Our Extended DKG (Sec. 4.1)	$k(n\lambda(t+1) + n + t)$	nk	$k(n(3n+5) - 1)$
Our Structured DKG (Sec. 4.2)	$n\lambda(t+1) + n + t$	nk	$n(3n+5) - 1$

start to finish, including when some of the parties are idle. We quantify the communication cost as the amount of outgoing communication per party. Our cost analysis methodology builds on that of [2] with some optimizations from [1].

In Fig. 7, we further plot the computational and communication costs of our protocols and compare them to the literature. We note that the number of isogeny computations coincides exactly with [2], currently the fastest in the literature, while other costs are negligible in comparison. In terms of communication, both the extended and structured versions of our protocols outperform their counterparts from the literature. For asymptotically large n, the communication cost of our protocols tend towards the communication cost of CSI-RAShi, as the commmunication cost of PVPs starts to dominate in these regions.

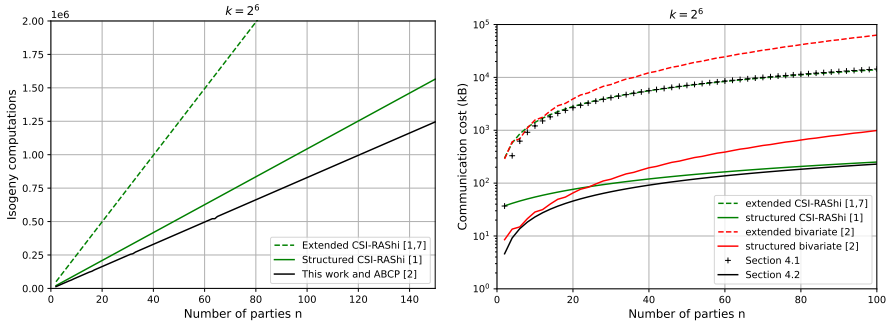

Fig. 7. Computational and communication costs of the DKG protocols [1, 2, 7] for the CSIDH-512 parameter set, shown as a function of the number of parties for $k = 2^6$.

5 More Efficient Threshold Signatures

As a further application of the NI-VSS scheme from Section 3 and the improved DKG protocols from Section 4, we revise two threshold signature schemes and make them more efficient. The first scheme is based on the isogeny-based signature scheme CSI-FiSh [8] and is proposed by Campos and Muth [12]. The second one is based on the Schnorr signature scheme [30] and is proposed by Gennaro, Jarecki, Krawczyk, and Rabin [21]. The latter is built upon Pedersen's DKG protocol [27].

5.1 Threshold, Efficient, and Robust CSI-SharK

We revisit the CSI-FiSh-based threshold signing protocol of Campos and Muth [12], and construct ThreshER SharK, which is a **Thresh**old, **E**fficient and **R**obust signature scheme based on CSI-**SharK** [1].

Campos and Muth's robust threshold signature scheme [12] is based on the basic version of CSI-FiSh [8], with its public key and the ephemeral keys being sampled by the CSI-RAShi DKG protocol [7]. The basic version of CSI-FiSh is based on an ID protocol that has a binary challenge space, leading to a signature scheme with a more concise public key. However, the shorter public key comes at the expense of reduced efficiency in signing and verification time, as well as an increase in signature size. One can extend the robust threshold signing protocol of Campos and Muth to work with the extended version of CSI-FiSh [8]. The extended version uses k independent curves in the public key, generated with independent secret keys and therefore comes at the cost of longer (public and secret) keys and a less efficient distributed DKG phase, since parties would need to repeat the same protocol k times.

Based on the comparisons presented in Tables 3-4 and Fig. 7, we can see that the DKG protocols for extended public keys are less efficient compared to those designed for structured public keys. It feels natural to work with structured public keys and therefore the structured CSI-RAShi++ DKG (Fig. 6), as

it outperforms other DKG protocols. Considering this, we propose two modifications to enhance the efficiency of Campos and Muth's robust threshold signature scheme [12]. First, we adapt their DKG scheme to work with CSI-SharK [1], which utilizes an SPK and has the same efficiency as the CSI-FiSh signature scheme. Specifically, we employ a variant of CSI-SharK that possesses a slightly larger public key size but yields a more efficient threshold signature scheme in practical scenarios. Consequently, the parties can use the structured CSI-RAShi++ DKG protocol to sample the keys and need to store only a single secret key. Following this, we translate their distributed signing protocol to work with the extended version of CSI-FiSh. Using a similar strategy to the CSI-RAShi++ DKG protocol (Fig. 5) allows to improve the efficiency of sampling the ephemeral keys the distributed signing protocol. Fig. 8 describes the algorithms of the resulting robust threshold signature scheme, which is called ThreshER SharK. In the figure, $\mathcal{H} : \{0,1\}^* \to (\Xi_k)^{t_k}$ is a random oracle which returns t_k elements from an exceptional set $\Xi_k = \{c_0 = 0, c_1 = 1, c_2, \ldots, c_{k-1}\}$ of size k. One may notice that ThreshER SharK uses the new NI-VSS scheme in both the key generation and singing protocols.

Efficiency. ThreshER SharK, utilizing an SPK, benefits from the ability to sample keys more efficiently using Structured CSI-RAShi++. While it is possible to extend Campos and Muth's robust threshold signature scheme [12] to accommodate the extended version of CSI-FiSh and gain efficiency through our proposed DKG protocols, it should be noted that the result would be less efficient than ThreshER SharK (We refer to Tables 3-4 for a more detailed comparison).

Security. We discuss security of our scheme in App. C.

5.2 More Efficient Schnorr Threshold Signatures

In this section, we leverage our protocols from Sections 3 and 4 and revisit Pedersen's DKG protocol [27] along with the threshold signature of Gennaro, Jarecki, Krawczyk, and Rabin [21], which uses Schnorr's signature [30] for signing and Pedersen's DKG protocol for generating the keys.

A Designated Verifier DKG Protocol for DL. We present an efficient Pedersen-like robust DKG protocol based on our proposed VSS scheme, which allows a set of parties to sample $h = g^x$ in a distributed manner, g being the generator of the DL group. To this end, we first construct an efficient NI-SZK proof scheme for the DL problem that acts as a building block in our proposed DKG protocol and threshold signature. The NI-SZK proof scheme allows a prover to convince a set of verifiers (i.e., shareholders) that $h = g^{f(0)} \wedge f(i) = x_i$, for the shared input $x = x_1 \parallel x_2 \parallel \cdots \parallel x_n$, a secret polynomial $f(X) \in \mathbb{F}_p[X]t$ and a secret input $x_0 = f(0)$. One is usually faced with a similar scenario in the DL-based threshold protocols (e.g., threshold variants of El Gamal, ECDSA, etc.).

KeyGen: Given an integer k, as the design parameter in CSI-SharK signature, the parties first agree on a public exceptional set $\Xi_k = \{c_0 = 0, c_1 = 1, c_2, \ldots, c_{k-1}\}$. Then, they engage in the $\mathsf{DKG}^{\text{Structured CSI-RAShi++}}$ protocol (refer to Fig. 6) to sample the public key $\mathsf{pk} := (E_0, E_1, \ldots, E_{k-1})$, where $E_i = [c_i x]E_0$. At the end of this phase, we have a qualified set, which w.l.o.g. we assume to be $Q_0 := \{P_1, P_2, \ldots, P_n\}$.

$\mathsf{Sign}(m, \langle x_i \rangle)$**:** To generate a signature on m, the parties in Q_0 act as follows.

1. For $l = 1, \ldots, t_k$, parties run $(b_l, F_l, B_l) \leftarrow \mathsf{DKG}^{\text{CSI-RAShi++}}(E_0)$. Note that at each invocation, malicious parties might be disqualified. We assume to end every step with a set $Q_l \subseteq Q_{l-1}$. Only the parties in Q_{t_k} continue the protocol.
2. The parties compute the challenge $d_1, \ldots, d_{t_k} \leftarrow H(F_1, \ldots, F_{t_k} || m)$.
3. For $l = 1, \ldots, t_k$, the parties in Q_{t_k} behave as follows.
   (a) each party P_i computes $r_l^{(i)}(X) = b_l^{(i)}(X) - d_l s^{(i)}(X)$.
   (b) using their secret values shared during the NI-VSS protocol, namely $s^{(i)}$ and $b_l^{(i)}$, each other party P_j verifies
   $$r_l^{(i)}(j) \stackrel{?}{=} b_l^{(i)}(j) - d_l s^{(i)}(j)$$
   (c) Whenever one of these checks fails, P_j broadcasts a complaint against P_i. When a player P_i has $t+1$ or more complaints against them, they are disqualified. The remaining players can then construct $r_l^{(i)}(0)$ by reconstructing both $b_l^{(i)}(0)$ and $s^{(i)}(0)$ using the information from the DKGs. This is always possible when there are at least $t+1$ honest parties.
   (d) For each party P_i in $Q_l \backslash Q_{t_k}$, reconstruct $r_l^{(i)}(0)$ in the same way.
   (e) For each party P_i in $Q_0 \backslash Q_l$, set and reconstruct $r_l^{(i)}(0) = s^{(i)}(0)$.
   (f) Using $\{r_l^{(i)}(0)\}_{i=1,\ldots,n}$, parties build the responses $r_l = \sum_{i \in Q} r_l^{(i)}(0)$.
4. Finally, parties output the signature (r, d), where $r = (r_1, \ldots, r_{t_k})$.

$\mathsf{Verify}((r, d_1, \ldots, d_{t_k}), m, \mathsf{pk})$**:** To verify a signature $(r, d_1, \ldots, d_{t_k})$ on m using the public key $\mathsf{pk} = (E_1, \ldots, E_{k-1})$, the verifier proceeds as follows.

1. For $l = 1, \ldots, t_k$, compute $F_l = [r_l]E_{d_l}$, where
2. Compute $d'_1, \ldots, d'_{t_k} \leftarrow H(F_1, \ldots, F_{t_k} || m)$
3. If $d_1 = d'_1 \wedge \cdots \wedge d_{t_k} = d'_{t_k}$, return valid, otherwise $\mathsf{invalid}$.

Fig. 8. ThreshER SharK: a Threshold, Efficient, and Robust signature scheme based on CSI-SharK.

The new NI-SZK proof scheme is built for the following n-distributed relations,

$$R_i = \{(g, h, x_i, f(X)) | h = g^{f(0)} \wedge f(i) = x_i\}, \tag{5}$$

where $i = 1, \ldots, n$. Fig. 9 describes the algorithms of our proposed NI-SZK proof scheme for the DL relation, where $\mathcal{H}$ is a random oracle and $\mathcal{C}$ is a computationally hiding commitment scheme. Roughly speaking, the protocol is obtained by slightly modifying the conjunction of the Schnorr ID protocol with the NI-SZK scheme presented in Fig. 3. This is another instance of different NI-SZK proof schemes that one would need in the *Reconstruction* phase of the new VSS

Prover: Given, $f(X) \in \mathbb{F}_p[X]_t$, and the input $x = (g, h := g^{f(0)}, x_1, \cdots, x_n)$, proceed as follows and output a proof π of the relations in equation (5).

1. Sample $b(X) \leftarrow \mathbb{F}_p[X]_t$ uniformly at random;
2. For $i = 1, \ldots, n$: Sample $y_i, y_i' \leftarrow \{0,1\}^\lambda$ uniformly at random and set $\mathsf{C}_i = \mathcal{C}(b(i), y_i)$ and $\mathsf{C}_i' = \mathcal{C}(x_i, y_i')$;
3. Sample $y_0, y_0' \leftarrow \{0,1\}^\lambda$ and set $\mathsf{C}_0 = \mathcal{C}(g^{b(0)}, y_0), \mathsf{C}_0' = \mathcal{C}(g \| h, y_0')$
4. Set $d \leftarrow \mathcal{H}(\mathsf{C}, \mathsf{C}')$, where $\mathsf{C} = (\mathsf{C}_0, \mathsf{C}_1, \ldots, \mathsf{C}_n), \mathsf{C}' = (\mathsf{C}_0', \mathsf{C}_1', \ldots, \mathsf{C}_n')$;
5. Set $r(X) \leftarrow b(X) - d \cdot f(X) \mod p$;
6. Return $\pi := (g, h, \mathsf{C}, \mathsf{C}', r(X), \{\pi_i\}_{i=1}^n)$, where $\pi_i = (y_0, y_i, y_0', y_i')$;

Verification: Given $\pi := (g, h, \mathsf{C}, \mathsf{C}', r(X), \{\pi_i := (y_0, y_i, y_0', y_i')\}_{i=1}^n)$, the verifiers $\{V_i\}_{i=1}^n$ use their shares and individual proofs (x_i, π_i), and the verification proceeds as follows:

1. Verifier i acts as below and outputs `true` or `false`.
   (a) If $\mathsf{C}_i' \neq \mathcal{C}(x_i, y_i')$ or $\mathsf{C}_0' \neq \mathcal{C}(g \| h, y_0')$ return `false`
   (b) Set $d' \leftarrow \mathcal{H}(\mathsf{C}, \mathsf{C}')$;
   (c) Compute $h' = g^{r(0)} \cdot h^d$.
   (d) If $\mathsf{C}_i == \mathcal{C}(r(i) + d \cdot x_i, y_i) \wedge \mathsf{C}_0 == \mathcal{C}(h', y_0)$ return `true`; otherwise `false`;
2. Return `true` if *all* the verifiers return `true`; otherwise returns `false`.

Fig. 9. A NI-SZK Proof Scheme for Discrete Logarithm.

scheme, where parties reconstruct a function of the main secret $f(0)$, namely $h = g^{f(0)}$, rather than the plain value of it.

Theorem 5.1 (NI-SZK Proofs for DL). *Let L be an n-distributed language for the list of relations given in equation (5), $t \geq 1$ be a security threshold such that $n \geq 2t + 1$, and $h = n - t$. Assuming that the commitment scheme $\mathcal{C}$ is computationally hiding, for any potential set $I \subseteq [n]$ of size $|I| \geq h$, the protocol described in Fig. 9 is a non-interactive distributed strong ZK protocol for L that satisfies completeness, strong ZK, and soundness against the prover and t malicious verifiers in the ROM.*

Proof. The proof is analogous to the proof of Theorem 3.1 which is omitted. We highlight that in this case, in the soundness proof, one reduces the security of scheme to the DL problem, thus this scheme does not achieve post-quantum security. □

Now, we can use the general NI-VSS of Fig. 4 and the NI-ZSK proof scheme given in Fig. 9, and construct a DKG protocol with designated verifiers for DL. The resulting DKG is described in Fig. 11 and can be considered as an adaption of the Pedersen DKG protocol version from [21] to work with NI-SZK proofs and the new VSS scheme.

Theorem 5.2. *Under the DL assumption, the protocol in Fig. 10 is a secure DKG protocol, namely it satisfies the correctness and secrecy properties against a malicious adversary corrupting up to t parties, with $t < n/2$.*

Round 1 (VSS and Committing): Each party P_i proceed as follows:

1. Sample $f^{(i)}(X) \leftarrow \mathbb{F}_p[X]_t$ subject to $f^{(i)}(0) = x_i$ and set $x_{ij} = f^{(i)}(j)$
2. Using $(g, h_i := g^{x_i}, \{x_{ij}\}_{j=1}^n)$, run the prover of NI-SZK proof scheme in Fig. 9, and obtain $\pi := (g, h_i, \mathsf{C}, \mathsf{C}', r(X), \{\pi_{ij}\}_{j=1}^n)$, where $\pi_{ij} = (y_{i0}, y_{ij}, y'_{i0}, y'_{ij})$.
3. Publish $(g, \mathsf{C}, \mathsf{C}', r(X))$ and store $(h_i, \{\pi_{ij}\}_{j=1}^n)$ for next round.

Round 2 (Opening, Verification, PK Computation):

1) **Opening**: Each party $\{P_i\}_{i=1}^n$ broadcasts $h_i = g^{x_i}$ and sends π_{ij} privately to party P_j. If a party refuses to open a commitment, then that party is disqualified.
2) **Verification**: Each party $\{P_j\}_{j=1}^n$ verifies the correctness of the share x_{ij} it got from P_i with respect to $g, h_i = g^{x_i}$ for $i \neq j$, by running the verifier of NI-SZK proof scheme given in Fig. 3. If the verification fails, then P_j broadcasts a complaint against P_i. Any player with at least $t+1$ complaints is disqualified. If P_j complains that P_i's proof does not verify, then P_i broadcasts $\pi_{ij} = (y_{ij}, y'_{ij})$, so that everyone can verify it using the verification algorithm of NI-SZK scheme. If this verification succeeds, the protocol continues as normal, otherwise P_i is disqualified. Since disqualifying the parties happens on the basis of only broadcasted information, at the end, all parties will agree on the same set of qualified parties $Q \subseteq \{1, \ldots, n\}$ such that $x = \sum_{i \in Q} L_{0,i}^Q x_i$.
3) **PK Computation:** Parties compute the public key as $g^x = \prod_{i=1}^n g^{x_i}$.

Fig. 10. Designated verifier DKG protocol for DL-based schemes.

Proof. The proof is analogous to the ones in [7,21], but in this case the simulator of DKG scheme runs the simulator of NI-SZK proof scheme as a subroutine. □

Finally, using the DKG protocol given in Fig. 10, we modify the Schnorr-based threshold signature scheme of Gennaro, Jarecki, Krawczyk, and Rabin [21], and present a new variant. Fig. 11 represents the description of the proposed robust threshold signature scheme that uses Fig. 10 for the DKG and the distributed generation of the ephemeral key g^b.

Theorem 5.3. *Under the DL problem the threshold signature scheme described in Fig. 11, is secure against a static adversary corrupting up to t parties, with $t < n/2$.*

We refer to App. D, for the security proof of the scheme.

Security Against Wagner's Attack. The threshold signature in Fig. 11 is secure against the concurrent attack using Wagner's algorithm described in [5]. Intuitively, the attack crucially relies on the fact that an adversary can open ℓ sessions of the protocol in parallel, that is in the same round. At each session r the adversary gets g^{b_i} from the honest parties and computes its commitment shares g^{b_j}, $j \in A$, based on the honest parties' commitment shares, before submitting g^b to the RO. For big enough ℓ this is enough for the adversary to forge a signature [5]. As mentioned in the same paper [5], a countermeasure is to let parties commit to the shares g^{b_i} and only after a round of broadcast they open them.

KeyGen: Parties run the DKG protocol of Fig. 10. At the end, each party holds a verified secret share x_i of x. The resulting public key is g^x. We assume $Q_0 = \{1, \ldots, n\}$ to be the qualified set at the end of this step.

Sign$(m, \langle x_i \rangle)$**:** To sign a message m, the parties in Q_0 act as follows.

1. Parties run the DKG protocol of Fig. 10 to compute g^b. Malicious parties might be disqualified, so we end up with a set $Q_1 \subseteq Q_0$. Only the parties in Q_1 continue the protocol. Each party P_i in Q_1 holds a share b_i of b.
2. The parties compute the challenge $d \leftarrow H(g^b||m)$.
3. Parties in Q_1 behave as follows.
   (a) Each party P_i computes and broadcasts $r^{(i)}(X) = b^{(i)}(X) - ds^{(i)}(X)$.
   (b) Using the shared values from VSS phase, each other party P_j verifies
   $$r^{(i)}(j) \stackrel{?}{=} b^{(i)}(j) - ds^{(i)}(j)\,.$$
   (c) Whenever one of these checks fails, P_j broadcasts a complaint against P_i. When a player P_i has $t+1$ or more complaints against them, they are disqualified. The remaining players can then construct $r^{(i)}(0)$ by reconstructing both $b^{(i)}(0)$ and $s^{(i)}(0)$. This is always possible when there are at least $t+1$ honest parties.
   (d) For each party P_i in $Q_0 \backslash Q_1$, set and reconstruct $r_l^{(i)}(0) = s_{d_l}^{(i)}(0)$.
   (e) Using $\{r^{(i)}(0)\}_{i=1,\ldots,n}$, parties build the response $r = \sum_{i \in Q} r^{(i)}(0)\,.$
4. Finally, parties output the signature $((r, d), m)$.

Verify$((r, d), m, \mathsf{pk})$**:** To verify a signature (r, d) on m using the public key $\mathsf{pk} = g^x$, the verifier proceeds as follows.

1. Compute $h = g^r \cdot (g^x)^{-d}$.
2. Compute $d' \leftarrow H(h||m)$.
3. If $d = d'$, return valid, otherwise invalid.

Fig. 11. A designated-verifier robust threshold signature scheme based on [21]

This clearly prevents the adversary to compute his shares adaptively. A typical way for implementing this is using PK-based commitments such as Pedersen's, as done for example in [21, 26]. We have a similar approach in our protocol, and reveal the commitments and opening at the beginning of the second round.

Efficiency. Next, we summarize the efficiency of the proposed DKG protocol (Fig. 10) and the threshold signature (Fig. 11) and compare it with the ones proposed by Gennaro et al. [21]. In comparison with the variant of Pedersen DKG, given in [21], in our DKG protocol, each party needs to compute about $2n$ exponentiations in the group, $5n$ hashes (for committing), and n degree-t polynomial evaluations in the field, instead of $2tn + 2n$ exponentiations in the group. In terms of communication, in our DKG protocol each party needs to broadcast about $2n$ images of a hash function (i.e., the commitments) and t field elements (i.e., coefficients of the $r(X)$), instead of $2t$ group elements.

In comparison with the threshold signature of Gennaro et al. [21], in our new variant (given in Fig. 11) each party needs to compute about $2n$ exponentiations in the group, $5n$ hashes, and $2n$ degree-t polynomial evaluations in the field, instead of $2tn + 4n$ exponentiations in the group. In terms of communication, in

our threshold signing protocol, each party needs to broadcast about $2n$ images of a hash function and $2t$ field elements, instead of $2t$ group elements.

As can be seen, in our proposed DKG and threshold signature the communication is slightly higher, but we can save considerably on the computational cost of each party, especially in larger-scale applications.

6 Conclusion

In this paper, we presented a general construction for building a NI-VSS scheme using a ZK proof scheme over secret shared data, as formally defined by Boneh et al. [10]. Leveraging this construction, we proposed a practical post-quantum secure NI-VSS scheme based on Shamir secret sharing.

The proposed NI-VSS scheme can be viewed as a modification of the variant of the Pedersen VSS scheme used in Gennaro et al.'s DKG protocol [21], where we replace the Pedersen commitment with a hash-based commitment of the form $C = H(m, r)$ instead of $C = g^m h^r$. The later modification pushes the protocol to the designated verifier setting, requires a ZK proof over secret shared data, but allows one to achieve post-quantum security. Consequently, the proposed NI-VSS scheme serves as a post-quantum secure alternative to the Pedersen VSS scheme [28] (or Feldman's VSS scheme [19]) in scenarios where public verifiability is not necessary. This holds true for various (post-quantum secure) threshold protocols such as DKG protocols and threshold signatures.

A key advantage of new NI-VSS scheme, when compared to the IT-secure BGW VSS scheme [4], is its reduced communication overhead and improved robustness. Specifically, our scheme requires half of the shareholders to be honest, as opposed to two-thirds in the BGW VSS scheme. To evaluate the performance of new NI-VSS scheme, we did a C++ implementation and obtained promising results: the dealer is capable of sharing a secret with 4096 parties in approximately 2 sec, while shareholders can verify their shares in less than 2 msec.

The proposed NI-VSS scheme allowed us to revisit and improve various threshold DKG and signing protocols [1, 2, 7, 12, 21]. Through our revisions, we have not only improved their performance but also relaxed the requirements on the number of honest parties in some cases. In the CSIDH setting, our modifications have led to the development of two DKG protocols that surpass the state-of-the-art solutions. Furthermore, we have presented an efficient threshold robust signature scheme based on isogenies. Our scheme builds upon the CSI-SharK signature [1], but it can also be adapted to work with the CSI-FiSh signature scheme [8], although with lower efficiency in the key generation phase.

Finally, we have also introduced a new version of the Pedersen DKG combined with Gennaro et al.'s threshold signature scheme [21]. While sacrificing public verifiability, our new versions achieve better efficiency, particularly in large-scale applications. One notable factor contributing to these improvements is the practical advantage of using a hash function for commitment, which often outperforms the Pedersen commitment. Our results show that, in practical set-

tings, DKG and threshold signing protocols with designated verifiers suffice for constructing a threshold signature scheme with public-verifier.

The notable efficiency and simple nature of the NI-VSS scheme make it a valuable tool for a wide range of threshold protocols, extending beyond the protocols revisited in this paper. Future research can explore the integration of the new NI-VSS scheme and the revised threshold protocols into other existing threshold protocols and assess their impact on their efficiency and security.

Acknowledgments

This work has been supported in part by the Defense Advanced Research Projects Agency (DARPA) under contract No. HR001120C0085, by the FWO under an Odysseus project GOH9718N, by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (Grant agreement No. 101020788 - Adv-ERC-ISOCRYPT), by CyberSecurity Research Flanders with reference number VR20192203, by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program under project PICOCRYPT (grant agreement No. 101001283), by the Spanish Government under project PRODIGY (TED2021-132464B-I00), and by the Madrid Regional Government under project BLOQUES (S2018/TCS-4339). The last two projects are co-funded by European Union EIE, and Next Generation EU/PRTR funds.

Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the ERC, DARPA, the US Government, the Spanish Government, Cyber Security Research Flanders or the FWO. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

References

1. Shahla Atapoor, Karim Baghery, Daniele Cozzo, and Robi Pedersen. CSI-SharK: CSI-FiSh with Sharing-friendly Keys. In Leonie Simpson and Mir Ali Rezazadeh Baee, editors, *Information Security and Privacy - 28th Australasian Conference, ACISP 2023, Brisbane, QLD, Australia, July 5-7, 2023, Proceedings*, volume 13915 of *Lecture Notes in Computer Science*, pages 471–502. Springer, 2023.
2. Shahla Atapoor, Karim Baghery, Daniele Cozzo, and Robi Pedersen. Practical robust DKG protocols for CSIDH. In Mehdi Tibouchi and Xiaofeng Wang, editors, *Applied Cryptography and Network Security - 21st International Conference, ACNS 2023, Kyoto, Japan, June 19-22, 2023, Proceedings, Part II*, volume 13906 of *Lecture Notes in Computer Science*, pages 219–247. Springer, 2023.
3. Karim Baghery, Daniele Cozzo, and Robi Pedersen. An isogeny-based ID protocol using structured public keys. In M.B. Paterson, editor, *Cryptography and Coding - 18th IMA International Conference, IMACC 2021, Oxford, UK, December 14-15, 2021, Proceedings*, volume 13129 of *Lecture Notes in Computer Science*, pages 179–197. Springer, 2021.

4. Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In *20th Annual ACM Symposium on Theory of Computing*, pages 1–10, Chicago, IL, USA, May 2–4, 1988. ACM Press.
5. Fabrice Benhamouda, Tancrède Lepoint, Julian Loss, Michele Orrù, and Mariana Raykova. On the (in)security of ROS. In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology – EUROCRYPT 2021, Part I*, volume 12696 of *Lecture Notes in Computer Science*, pages 33–53, Zagreb, Croatia, October 17–21, 2021. Springer, Heidelberg, Germany.
6. Daniel Bernstein, Luca De Feo, Antonin Leroux, and Benjamin Smith. Faster computation of isogenies of large prime degree. *arXiv preprint arXiv:2003.10118*, 2020.
7. Ward Beullens, Lucas Disson, Robi Pedersen, and Frederik Vercauteren. CSI-RAShi: Distributed key generation for CSIDH. In Jung Hee Cheon and Jean-Pierre Tillich, editors, *Post-Quantum Cryptography - 12th International Workshop, PQCrypto 2021, Daejeon, South Korea, July 20-22, 2021, Proceedings*, volume 12841 of *Lecture Notes in Computer Science*, pages 257–276. Springer, 2021.
8. Ward Beullens, Thorsten Kleinjung, and Frederik Vercauteren. CSI-FiSh: Efficient isogeny based signatures through class group computations. In Steven D. Galbraith and Shiho Moriai, editors, *Advances in Cryptology – ASIACRYPT 2019, Part I*, volume 11921 of *Lecture Notes in Computer Science*, pages 227–247, Kobe, Japan, December 8–12, 2019. Springer, Heidelberg, Germany.
9. Anurag Bishnoi, Pete L Clark, Aditya Potukuchi, and John R Schmitt. On zeros of a polynomial in a finite grid. *Combinatorics, Probability and Computing*, 27(3):310–333, 2018.
10. Dan Boneh, Elette Boyle, Henry Corrigan-Gibbs, Niv Gilboa, and Yuval Ishai. Zero-knowledge proofs on secret-shared data via fully linear PCPs. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019, Part III*, volume 11694 of *Lecture Notes in Computer Science*, pages 67–97, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Heidelberg, Germany.
11. Xavier Bonnetain and André Schrottenloher. Quantum security analysis of CSIDH. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology – EUROCRYPT 2020, Part II*, volume 12106 of *Lecture Notes in Computer Science*, pages 493–522, Zagreb, Croatia, May 10–14, 2020. Springer, Heidelberg, Germany.
12. Fabio Campos and Philipp Muth. On actively secure fine-grained access structures from isogeny assumptions. In Jung Hee Cheon and Thomas Johansson, editors, *Post-Quantum Cryptography - 13th International Workshop, PQCrypto 2022, Virtual Event, September 28-30, 2022, Proceedings*, volume 13512 of *Lecture Notes in Computer Science*, pages 375–398. Springer, 2022.
13. Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. CSIDH: An efficient post-quantum commutative group action. In Thomas Peyrin and Steven Galbraith, editors, *Advances in Cryptology – ASIACRYPT 2018, Part III*, volume 11274 of *Lecture Notes in Computer Science*, pages 395–427, Brisbane, Queensland, Australia, December 2–6, 2018. Springer, Heidelberg, Germany.
14. Benny Chor, Shafi Goldwasser, Silvio Micali, and Baruch Awerbuch. Verifiable secret sharing and achieving simultaneity in the presence of faults (extended abstract). In *26th Annual Symposium on Foundations of Computer Science*, pages 383–395, Portland, Oregon, October 21–23, 1985. IEEE Computer Society Press.

15. Anders Dalskov, Eysa Lee, and Eduardo Soria-Vazquez. Circuit amortization friendly encodings and their application to statistically secure multiparty computation. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 213–243. Springer, 2020.
16. Luca De Feo. Mathematics of isogeny based cryptography. *arXiv preprint arXiv:1711.04062*, 2017.
17. Luca De Feo, Tako Boris Fouotsa, Péter Kutas, Antonin Leroux, Simon-Philipp Merz, Lorenz Panny, and Benjamin Wesolowski. SCALLOP: Scaling the CSI-FiSh. In Alexandra Boldyreva and Vladimir Kolesnikov, editors, *PKC 2023: 26th International Conference on Theory and Practice of Public Key Cryptography, Part I*, volume 13940 of *Lecture Notes in Computer Science*, pages 345–375, Atlanta, GA, USA, May 7–10, 2023. Springer, Heidelberg, Germany.
18. Jelle Don, Serge Fehr, Christian Majenz, and Christian Schaffner. Security of the Fiat-Shamir transformation in the quantum random-oracle model. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019, Part II*, volume 11693 of *Lecture Notes in Computer Science*, pages 356–383, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Heidelberg, Germany.
19. Paul Feldman. A practical scheme for non-interactive verifiable secret sharing. In *28th Annual Symposium on Foundations of Computer Science*, pages 427–437, Los Angeles, CA, USA, October 12–14, 1987. IEEE Computer Society Press.
20. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *Advances in Cryptology – CRYPTO'86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194, Santa Barbara, CA, USA, August 1987. Springer, Heidelberg, Germany.
21. Rosario Gennaro, Stanislaw Jarecki, Hugo Krawczyk, and Tal Rabin. Secure distributed key generation for discrete-log based cryptosystems. *Journal of Cryptology*, 20(1):51–83, January 2007.
22. Rosario Gennaro, Michael O. Rabin, and Tal Rabin. Simplified VSS and fast-track multiparty computations with applications to threshold cryptography. In Brian A. Coan and Yehuda Afek, editors, *17th ACM Symposium Annual on Principles of Distributed Computing*, pages 101–111, Puerto Vallarta, Mexico, June 28 – July 2, 1998. Association for Computing Machinery.
23. Craig Gentry, Shai Halevi, and Vadim Lyubashevsky. Practical non-interactive publicly verifiable secret sharing with thousands of parties. In Orr Dunkelman and Stefan Dziembowski, editors, *Advances in Cryptology – EUROCRYPT 2022, Part I*, volume 13275 of *Lecture Notes in Computer Science*, pages 458–487, Trondheim, Norway, May 30 – June 3, 2022. Springer, Heidelberg, Germany.
24. Jens Groth. Non-interactive distributed key generation and key resharing. Cryptology ePrint Archive, Report 2021/339, 2021. https://eprint.iacr.org/2021/339.
25. Alexei Y. Kitaev. Quantum measurements and the abelian stabilizer problem. *Electron. Colloquium Comput. Complex.*, TR96-003, 1996.
26. Chelsea Komlo and Ian Goldberg. FROST: flexible round-optimized schnorr threshold signatures. In Orr Dunkelman, Michael J. Jacobson Jr., and Colin O'Flynn, editors, *Selected Areas in Cryptography - SAC 2020 - 27th International Conference, Halifax, NS, Canada (Virtual Event), October 21-23, 2020, Revised Selected Papers*, volume 12804 of *Lecture Notes in Computer Science*, pages 34–65. Springer, 2020.
27. Torben P. Pedersen. A threshold cryptosystem without a trusted party (extended abstract) (rump session). In Donald W. Davies, editor, *Advances in Cryptology –*

EUROCRYPT'91, volume 547 of *Lecture Notes in Computer Science*, pages 522–526, Brighton, UK, April 8–11, 1991. Springer, Heidelberg, Germany.
28. Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In Joan Feigenbaum, editor, *Advances in Cryptology – CRYPTO'91*, volume 576 of *Lecture Notes in Computer Science*, pages 129–140, Santa Barbara, CA, USA, August 11–15, 1992. Springer, Heidelberg, Germany.
29. Chris Peikert. He gives C-sieves on the CSIDH. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology – EUROCRYPT 2020, Part II*, volume 12106 of *Lecture Notes in Computer Science*, pages 463–492, Zagreb, Croatia, May 10–14, 2020. Springer, Heidelberg, Germany.
30. Claus-Peter Schnorr. Efficient identification and signatures for smart cards. In Gilles Brassard, editor, *Advances in Cryptology – CRYPTO'89*, volume 435 of *Lecture Notes in Computer Science*, pages 239–252, Santa Barbara, CA, USA, August 20–24, 1990. Springer, Heidelberg, Germany.
31. Berry Schoenmakers. A simple publicly verifiable secret sharing scheme and its application to electronic. In Michael J. Wiener, editor, *Advances in Cryptology – CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 148–164, Santa Barbara, CA, USA, August 15–19, 1999. Springer, Heidelberg, Germany.
32. Adi Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, 1979.
33. Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *35th Annual Symposium on Foundations of Computer Science*, pages 124–134, Santa Fe, NM, USA, November 20–22, 1994. IEEE Computer Society Press.
34. Joseph H Silverman. *The arithmetic of elliptic curves*, volume 106. Springer Science & Business Media, 2009.
35. Dominique Unruh. Computationally binding quantum commitments. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016, Part II*, volume 9666 of *Lecture Notes in Computer Science*, pages 497–527, Vienna, Austria, May 8–12, 2016. Springer, Heidelberg, Germany.
36. Dominique Unruh. Post-quantum security of Fiat-Shamir. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology – ASIACRYPT 2017, Part I*, volume 10624 of *Lecture Notes in Computer Science*, pages 65–95, Hong Kong, China, December 3–7, 2017. Springer, Heidelberg, Germany.
37. Jacques Vélu. Isogénies entre courbes elliptiques. *CR Acad. Sci. Paris, Séries A*, 273:305–347, 1971.

A Proof of Theorem 3.1

A.1 Completeness

Lemma A.1. *The protocol described in Fig. 3 constitutes a complete non-interactive distributed strong ZK protocol in the QROM for the k-distributed language L list of relations of (4) if the used commitment scheme is collapsing and quantum computationally hiding.*

Proof. If the protocol is followed correctly and if the input was a valid statement-witness pair $(x, w) \in R$, where each verifier V_i, $1 \leq i \leq n$, has an input piece (share) x_i, then the verification will accept the proof with probability 1. Namely, the prover computes $b(i)$ for $i = 1, \cdots, n$, and the verifier V_i computes $r(i) +$

$dx_i = b(i) - df(i) + dx_i = b(i)$, if $x_i = f(i)$. So if the witness is valid, then the C_i match and all the verification (i.e., all the shareholders) will return `True` and accept the proof. □

A.2 Soundness Against the Prover and t Malicious Verifiers

The NI-SZK scheme presented in Fig. 3 is made non-interactive using a variant of Fiat-Shamir transform which is proposed by Boneh et al. [10] for proofs on distributed data, and its security is also proven formally in [7] for a particular protocol. The commonly used transform of Fiat-Shamir [20], which is analysed in the (Quantum) Random Oracle model [18, 36], is applied on a public coin interactive proof systems. Such that, instead of getting the challenge from the verifier, in the non-interactive protocol the prover applies a random oracle H to the concatenation of the input (i.e., the statement), and the communication transcript up to that point. In the case of sigma protocols the communication transcript is the commitment made in the first round. But in this variant of the Fiat-Shamir transform, the challenge is public, but the input (i.e., the statement) is shared among the verifiers and cannot be revealed to any single verifier. To deal with this concern, earlier works [7,10] proposed to generate the random challenge using the joint view of the verifiers in previous rounds. Namely, the prover obtains the random challenge value as the hash of concatenation of n public commitments to the individual shares (i.e., shares of statement), and n public commitments produced in the initial round of the sigma protocol. Note that in this variant, each individual secret share is linked to a public commitment which satisfies (perfect) binding and hiding and can be verified by the corresponding shareholder.

The following Lemma is proven in [7], which proves the soundness of a NI-SZK argument that is built using the above variant of Fiat-Shamir transform.

Lemma A.2. *Suppose $\Sigma = (P_1, V_1, P_2, V_2)$ is a sigma protocol for the relation R with super-polynomially sized challenge space $\mathcal{C}h$, special soundness, and quantum computationally unique responses. Let $\Sigma' = (P_1', V_1', P_2', V_2')$ be the following sigma protocol:*

$$\begin{aligned} P_1'(x,w) &: y \leftarrow \{0,1\}^\lambda,\ \mathsf{C}_x \leftarrow \mathcal{C}(x,y), \\ &\quad\ com \leftarrow P_1(x,w),\ com' = (\mathsf{C}_x, com) \\ V_1'(com') &: ch \leftarrow \mathcal{C}h \\ P_2'(ch) &: rsp \leftarrow P_2(ch), rsp' \leftarrow (x,y,rsp) \\ V_2'(x,com',ch,rsp') &: \textit{accept if } \mathsf{C}_x = \mathcal{C}(x,y) \textit{ and } V_2(x,com,ch,rsp) = 1 \end{aligned}$$

Then the non-interactive version of Σ', transformed by the mentioned variant of Fiat-Shamir transform is a non-interactive quantum proof of knowledge for the same relation R, assuming that $\mathcal{C}$ is a collapsing commitment.

In the rest, we prove the protocol (given in Fig. 3) satisfies soundness against the prover and t malicious verifiers. Note that we structure our proof along the lines of [7], but do this for a different relation, which has very different implications.

Lemma A.3. *The proof system given in Fig. 3 constitutes a NI-SZK argument in the QROM for the list of relations of equation (4) if the deployed commitment scheme is collapsing.*

Proof. The results from Boneh et al. [10] show that in a NI-SZK proof scheme over secret shared data, the best combination of soundness and ZK that we can achieve is strong zero-knowledge combined with soundness against prover and t malicious verifiers. In order to achieve this, we require to have at least $t+1$ honest parties among $n \geq 2t+1$ verifiers, i.e. be in the honest majority setting. Achieving these combinations means that in the target NI-SZK proof scheme, the prover can collude with t malicious verifiers to break the soundness, and at most t verifiers are allowed to collude to break the ZK and learn about the witness.

As a result, we need to prove that for any set $I \subset \{1, \cdots, n\}$ of honest parties where $|I| > t$ and any poly-time quantum adversary $\mathcal{A}^{RO}$, the following advantage is negligible:[7]

$$\mathsf{Adv}^{\mathsf{sound}}_{\mathcal{A},I}(\lambda) = \Pr\left[\begin{matrix}\forall i \in I : V^{RO}(i, x_i, \widetilde{\pi}, \pi_i) = 1 \\ \nexists w \forall i : (x_i, w) \in R_i\end{matrix}\middle| \{(x_i, \pi_i)\}_{i\in I} \leftarrow \mathcal{A}^{RO}(1^\lambda)\right].$$

For compactness, we use the index I to denote the set of elements with index $i \in I$, e.g. $x_I = \{x_i\}_{i\in I}$. We further define the function F, which on the input of the data available to the set I, outputs the commitments as follows.

$$\begin{aligned} F : \Xi_k \times \mathbb{Z}_N^{|I|} \times \{0,1\}^{\lambda|I|} \times \mathbb{Z}_N[X]_{\leq t} &\to \{0,1\}^{2\lambda} \\ (d, x_I, y_I, r(X)) &\mapsto \{\mathcal{C}(r(i) + dx_i, y_i)\}_{i\in I}, \end{aligned}$$

Let us define the following protocol $\Sigma' = (P'_1, V'_1, P'_2, V'_2)$.

$$\begin{aligned} P'_1(x_I, w) &: \forall i \in I : y_i, y'_i \leftarrow \{0,1\}^\lambda,\ \mathsf{C}'_i \leftarrow \Xi(x_i, y'_i) \\ &\quad b(X) \leftarrow \mathbb{Z}_N[X]_{\leq t},\ \mathsf{C}_I = F(0, x_I, y_I, b(X)) \\ V'_1(\mathsf{C}_I, \mathsf{C}'_I) &: d \leftarrow \Xi_k \\ P'_2(\mathbf{c}) &: r(X) = b(X) - dw,\ rsp' = (y_I, y'_I, r(X)) \\ V'_2(rsp) &: \text{accept if } \mathsf{C}'_I = \mathcal{C}(x_I, y_I) \text{ and } \mathsf{C}_I = F(d, x_I, y_I, r(X)). \end{aligned}$$

It is clear that, if $F(RO(\mathsf{C}, \mathsf{C}'), x_I, y_I, r(X)) = \mathsf{C}_I$ and $\mathsf{C}'_I = \mathcal{C}(x_I, y'_I)$, then $V^{RO}(I, x_I, \widetilde{\pi}, \pi_I) = 1$.[8] This implies that $\mathsf{Adv}^{\mathsf{sound}}_{\mathcal{A},I}(\lambda)$ is indeed negligible if the previously discussed variant of the Fiat-Shamir transform of Σ' is a (quantum) computationally sound proof of R_I as per [18, Definition 9]. This, in turn, is implied by proving that the transformed protocol is a quantum proof of knowledge as per [18, Definition 14]. We prove this last step by using Lemma A.2, which implies the soundness of our protocol from Figure 3, when the following protocol has super-polynomially sized challenge space Ξ_k, special soundness, and quantum computationally unique responses.

$$P_1(x_I, w) : \forall i \in I : y_i \leftarrow \{0,1\}^\lambda,$$

[7] For $|I| \leq t$, there always exists a witness that satisfies the relation.
[8] Read component-wise, e.g. $\forall i \in I :\ \mathsf{C}'_i = \mathcal{C}(x_i, y'_i)$.

$$
\begin{aligned}
b(X) \leftarrow \mathbb{Z}_N[X]_{\leq t}, \mathsf{C}_I = F(0, x_I, y_I, b(X)) \\
V_1(\mathsf{C}_I) : d \leftarrow \Xi_k \\
P_2(\mathbf{c}) : r(X) = b(X) - dw,\ rsp' \leftarrow (y_I, r(X)) \\
V_2(rsp) :\ \text{accept if } \mathsf{C}_I = F(d, x_I, y_I, r(X))
\end{aligned}
$$

We end this proof by discussing that these three properties are satisfied.

- **Challenge space**: In the case where $\mathbb{Z}_N$ is a field, the exceptional set Ξ_k is simply the field itself, which by definition has size superpolynomial in λ. Otherwise, the maximal size of Ξ_k is limited by the smallest divisor of N. In that case, as mentioned in Remark 3.1 we can amplify the challenge space size to above 2^λ by repeating the protocol $\lceil \lambda / \log k \rceil$ times.
- **Special Soundness**: Let $(x_I, C_I, d, r(X))$ and $(x_I, C_I, d', r'(X))$ with $d \neq d'$ be two accepting transcripts. Now, if for some $i \in I$, we have $r(i) + dx_i \neq r'(i) + d'x_i$, then we have found a collision in $\mathcal{C}$. Otherwise, we can compute a witness for x_I via $\frac{r(X) - r'(X)}{d' - d}$. Note that $d' - d$ is invertible because they are distinct elements from an exceptional set Ξ_k.
- **Unique responses**: Using the results from [7, Section A.2], this property is guaranteed in case $\mathcal{C}$ is collapsing and that $r(X)$ are unique. The latter follows from the fact that the function $(r, d, x) \mapsto r + dx$ is injective if d is an element from an exceptional set. □

A.3 Strong zero-knowledge

We begin this section by stating the following Lemma.

Lemma A.4. *The protocol in Fig. 3 satisfies the strong zero-knowledge property in the QROM for the list of relations of (4) if the used commitment scheme is quantum computationally hiding and collapsing, and if the underlying sigma protocol has honest-verifier zero-knowledge, completeness, and unpredictable commitments.*

We leave this Lemma without proof as it immediately follows from the discussion in [7, Section A.3]. There, the authors show that it suffices to show zero-knowledge for any $I \subset \{0, \ldots, n\}$.[9] The case $i = 0$ is not relevant in this work and for the cases $i = 1, \ldots, n$, we can readily apply the results of their Lemmas 4 and 5 to our protocol. By definition, our commitment scheme is quantum computationally hiding and collapsing. We can therefore finish the proof by showing that the sigma protocol underlying Fig. 3 is complete, HVZK, and has unpredictable commitments.

- **Completeness**: It follows immediately from Lemma A.1 that our protocol has perfect completeness.

[9] Actually, the authors implicitly prove strong zero-knowledge, but do not call it as such.

- **HVZK**: We can define a simulator that samples $y_I, y'_I \leftarrow \{0,1\}^\lambda$, $r(X) \leftarrow \mathbb{Z}_N[X]_t$ and $d \leftarrow \Xi_k$ uniformly at random, then sets $C_I = F(d, x_I, y_i, r(X))$ and $C'_I = \mathcal{C}(x_I, y'_I)$. Since these sampled elements are also uniformly random in the real execution of the protocol, the transcripts are perfectly indistinguishable.
- **Unpredictable commitments**: By [36, Definition 4], unpredictable commitments imply that two for every $(x_I, w) \in R_I$, different commitments $(C_I^{(1)}, C_I'^{(1)})$ and $(C_I^{(2)}, C_I'^{(2)})$ that satisfy the probability

$$\Pr\left[(C_I^{(1)}, C_I'^{(1)}) = (C_I^{(2)}, C_I'^{(2)}) \middle| \begin{matrix}(C_I^{(1)}, C_I'^{(1)}) \leftarrow P_1(x_I, w) \\ (C_I^{(2)}, C_I'^{(2)}) \leftarrow P_1(x_I, w)\end{matrix}\right]$$

is negligible in the security parameter λ. There are two options to get such a collision, either the inputs to $\mathcal{C}$ are equal, or we find a collision in $\mathcal{C}$. The former happens with negligible probability, since the inputs to $\mathcal{C}$ are uniformly distributed in $\mathbb{Z}_N \times \{0,1\}^\lambda$ and the latter is prevented by the fact that $\mathcal{C}$ is collapsing. □

B Proof of Theorem 3.2

Proof. In Theorem 3.1, we showed that the protocol presented in Fig. 3 is a NI-SZK scheme for L, that is an n-distributed language for the list of relations given in equation (4), satisfies completeness, strong ZK, and soundness against the prover and t malicious verifiers in the QROM.

Completeness of the NI-SZK scheme implies that if the parties $P_1, \cdots, P_n$ follow the protocol, then at the end of the *Sharing* phase, each of them obtain a distinct evaluation of a polynomial degree t, where $t < n$. Relaying on the fact that any degree t polynomial is uniquely determined by $t+1$ distinct evaluations, any $t+1$ of n shareholders can use Lagrange interpolation and reconstruct $f(X)$ and retrieve the value $f(0)$, which is the secret value in the NI-VSS scheme. Similarly, any $t+1$ shareholders can also verify the proof given by dealer in the *Reconstruction* phase, and ensure that the secret value x_0 revealed by the dealer, is equal to the secret shared value $f(0)$.

The NI-SZK scheme's soundness against prover and t malicious verifiers implies that if there is no n-distributed input $x' \in L_R$ such that $x_i = x'_i$, for all honest parties P_i, then the protocol (honest verifiers) will reject the proof except with negligible probability. Therefore, a malicious dealer would have to either break the soundness of the underlying NI-SZK proof scheme or it will be caught with an overwhelming probability. It is important to note that, during the verification process any conflicts between the dealer and shareholders are resolved using the method outlined in the Verification algorithm. In the scenario where the majority of shareholders are honest, this enables the parties to achieve robustness within the resulting NI-VSS scheme.

The strong ZK property of the underlying NI-SZK scheme guarantees that any polynomial-time adversary $\mathcal{A}$ that controls up to t verifiers cannot learn

anything about the secret polynomial $f(x)$, including the value $f(0)$. As a result, any non-qualified set of shareholders is unable to recover the secret $x_0 = f(0)$. In other words, if an adversary who controls a non-qualified set of shareholders can recover the secret value x_0, they can be used as an adversary against the strong ZK property of the NI-SZK proof scheme. □

C On the Security of ThreshER SharK

Here we give an intuitive proof of the security of the protocol in Fig. 8. The proof is a reduction-based argument, which means that we show that given an adversary that breaks the sEU-CMA security of the signing protocol in Fig. 8, then we can construct an efficient algorithm $\mathcal{S}$ that solves a hard problem. The problem will be the one behind the CSI-SharK signature scheme, which is the Vectorization Problem with Auxiliary Inputs (VPwAI) [3]. The proof is analogous to the DL-based one given in [21, Theorem 2].

Assume $\mathcal{A}$ is an adversary controlling up to t parties in the protocol in Fig. 8 and that breaks the unforgeability property of the threshold signature. The simulator $\mathcal{S}$ receives a set of k structured supersingular elliptic curves along with the exceptional set $\Xi_k = \{c_0 = 0, c_1 = 1, \cdots, c_{k-1}\}$,

$$\left\{\Xi_k, E_0, E_1^* = [c_1 x^*]E_0, \cdots, E_{k-1}^* = [c_{k-1}x^*]E_0\right\}$$

from the challenger. He needs to find the secret isogeny x^*. The simulator embeds the elliptic curves he got from the challenger into the protocol for computing the public key. Specifically, he simulates the DKG protocol, so that the resulting public key is $E_0, E_1^* = [c_1 x^*]E_0, \cdots, E_{k-1}^* = [c_{k-1}x^*]E_0$. To do this, like in the proof of the protocol in Fig. 6, he follows the protocol faithfully for all honest parties except one, say P_{j^*}, for which he sets $E_i^j = E_i^*$ for $i = 1, \cdots, k-1$. This way, the contribution of party P_{j^*} is implicitly set to be $x^* - \sum_{i=1}^{j^*-1} x_i$.

After the key generation, the simulator samples $d = (d^1, \cdots, d^{t_k})$ and $r = (r^{(1)}, \cdots, r^{(t_k)})$ uniformly at random. Then, he computes $F_i = [d^{(i)}]E_{d_i}$ for each $i = 1, \cdots, t_k$ and programs the random oracle so that

$$d = H(F_1 \| \cdots \| F_{t_k} \| m).$$

Then the simulator plays the role of the honest parties in step 1 of Fig. 8 so as to force the ephemeral curves to be $F_{(1)}, \cdots, F_{(t_k)}$, for each one following exactly the same strategy as in the proof of Fig. 5. This succeeds except if the query $(F_1, \cdots, F_{t_k}, m)$ was already asked the RO and the output was not $d_1, \cdots, d_{t_k}$. One can prove that this probability is negligible.

After successfully simulating the ephemeral keys, the simulator can easily simulate the partial responses. For $r_i^{(j)}$ with $i \neq j^*, j = 1, \cdots, t_k$, the simulator computes it by using the shares x_i and $b_i^{(j)}$ it either chose or got from the adversary. For $r_{j^*}^{(j)}$, the simulator simply computes $r_{j^*}^{(j)} = r^{(j)} - \sum_{i \neq j^*} r_i^{(j)}$. This concludes the simulation part.

In order to extract the secret isogeny x^*, the simulator follows the following strategy. After getting a forgery $(d_1, \cdots, d_{t_k}), (r^{(1)}, \cdots, r^{(t_k)})$ on a message m, he rewinds the adversary to the point where he makes the query $(F_1 \| \cdots, F_{t_k} \| m)$ to H. Then he continues to simulate from that point on with fresh randomness. By the forking lemma, with non-negligible probability, the adversary will produce a new forgery $(\tilde{d}_1, \cdots, \tilde{d}_{t_k}), (\tilde{r}^{(1)}, \cdots, \tilde{r}^{(t_k)})$, for the same message m. Then, the simulator computes

$$x^* = \frac{r^{(l)} - \tilde{r}^{(l)}}{\tilde{d}^{(l)} - d^{(l)}} - \sum_{j \neq j^*} x_j,$$

which is a solution for the VPwAI.

D Security of the Schnorr-based Threshold Singapore

The proof is analogous to that of Gennaro et al. [21]. Let $\mathcal{A}$ be an adversary that breaks the unforgeability of the threshold signing protocol in Fig. 11. Then we construct a simulator Sim that uses $\mathcal{A}$ as a subroutine to break the DL problem.

Specifically, the challenger for the DL problem gives h_T to Sim which needs to find x_T such that $h_T = g^{x_T}$. The strategy is to embed h_T into the public key computation of the protocol in Fig. 11. For simplicity[10], assume that the set of malicious parties is $A = \{P_1, \ldots, P_t\}$ while the set of honest parties is $H = \{P_{t+1}, \ldots, P_n\}$.

Sim simulates the parties in H correctly except one, say P_n. In other words, for $P_{t+1}, \ldots, P_{n-1}$ it follows the DKG protocol as an honest party would, therefore producing a partial public key h_i. While for party P_n, it simulates the NI-SZK in such a way that h_T is the partial public key of P_n.

To sign a message m, Sim samples $r_n, c \leftarrow \mathbb{Z}_p$ uniformly at random, and sets $z_n := g^{r_n} h_T^{-c}$. Then Sim simulates the DKG protocol for computing the ephemeral key z the same way as it did for the public key, by forcing the contribution of P_n to be z_n. It finally programs the RO so that $c = H(z \| m)$. The simulation of the ephemeral key succeeds except when the query (z, m) is queried to the RO before Sim gets the value z. One can prove that this only happens with negligible probability. After that, Sim simply follows the protocol normally, using the shares r_i that it computed honestly and r_n to compute the response $r = \sum_i r_i$, therefore simulating the transcripts of the protocol.

If the adversary is able to forge a signature, say (c, r) on a message m with non-negligible probability ε, then Sim re-winds $\mathcal{A}$ to the point where it asked the query $(z \| m)$ to H, where $z = g^r y^{-c}$. From that point, the simulator keeps simulating the protocol with fresh randomness and eventually will get a new forgery (c', r') on m with $(c' \neq c)$. The probability that this happens is non-negligible in ε by the forking lemma. This means that, with non-negligible probability, Sim gets two pairs $(c, r), (c', r')$ such that

$$z = g^r h^{-c} = g^{r'} h^{-c'},$$

[10] This is without loss of generality in that the protocol is symmetric for all parties

or, by expanding each factor,

$$g^r \left(\prod_{i=1}^{n-1} (g^{x_i})^{-c} \right) \cdot h_T^{-c} = g^{r'} \left(\prod_{i=1}^{n-1} (g^{x_i})^{-c'} \right) \cdot h_T^{-c'}$$

from which Sim can compute the discrete logarithm of h_T as $x_T = \frac{r-r'}{c'-c} - \sum_{i=1}^{n-1} x_i$, given that it has all the shares x_i for $i \neq n$.