CAPYBARA and TSUBAKI: Verifiable Random Functions from Group Actions and Isogenies

Yi-Fu Lai[1,2]

[1]University of Auckland, New Zealand
[2]Ruhr University Bochum
`Yi-Fu Lai @ ruhr-uni-bochum de`

18th December 2023

Abstract

In this work, we introduce two post-quantum Verifiable Random Function (VRF) constructions based on abelian group actions and isogeny group actions with a twist. The former relies on the standard Decisional Diffie-Hellman (DDH) assumption. VRFs serve as cryptographic tools allowing users to generate pseudorandom outputs along with publicly verifiable proofs. Moreover, the residual pseudorandomness of VRFs ensures the pseudorandomness of unrevealed inputs, even when multiple outputs and proofs are disclosed. Our work aims at addressing the growing demand for post-quantum VRFs, as existing constructions based on elliptic curve cryptography (ECC) or DDH-type assumptions are vulnerable to quantum threats.

In our contributions, our two VRF constructions, rooted in number-theoretic pseudorandom functions, are both simple and secure. We introduce a new proof system for the factorization of group actions and set elements, serving as the proofs for our VRFs. The first proposal is based on the standard DDH problem, and for its security proof, we introduce the master Decisional Diffie-Hellman problem over group actions, proving its equivalence to the standard DDH problem. In the second construction, we leverage quadratic twists to enhance efficiency, reducing the key size and the proof sizes, expanding input size with a relaxed the assumption to the square DDH problem.

Moreover, we employ advanced techniques from the isogeny literature to optimize the proof size to 39KB and 34 KB using CSIDH-512 without compromising VRF notions. To the best of our knowledge, these constructions represent the first two provably secure VRFs based on isogenies.

1 Introduction

Verifiable random functions (VRFs) are a cryptographic primitive that were first introduced by Micali, Rabin, and Vadhan [MRV99]. They are a more advanced form of pseudorandom functions (PRFs) that not only generate pseudorandom outputs, but also provide a non-interactive and publicly verifiable proof to validate the output. The security of VRFs is maintained even when numerous copies of the input, output, and proof are made public. In particular, the notion of residual pseudorandomness for VRFs ensures that the pseudorandomness remains for inputs that have not been evaluated and the unique provability guarantees that it is computationally infeasible for an attacker to generate distinct outputs for the same input with valid proofs.

The versatility of VRFs has been demonstrated through their applications in DNSSEC protocols [GNP+15] and, especially, blockchain technology and e-lottery [GHM+17, HMW18, EKS+21]. The growth of cryptocurrencies such as Bitcoin and Algorand has spurred significant interest in blockchain technology, which is being fueled by its potential. Early blockchain systems, such as Bitcoin, utilized the Proof-of-Work (PoW) consensus mechanism, where miners compete to solve a cryptographic puzzle and the winner is rewarded. In contrast, the Proof-of-Stake (PoS) consensus protocol provides a more environmentally sustainable solution by allowing validators to stake their tokens and conducting an online lottery. Due to the cryptographic properties, VRFs play a critical role in PoS blockchain applications for their applications in cryptographic sortition and Byzantine consensus [GHM+17, DGKR18, HMW18].

In practice, most existing VRFs are based on elliptic curve cryptography (ECC), pairing-based BLS-type signatures or other Diffie-Hellman-type assumptions [BGLS03, BMR10, ACF14, Jag15, PWH+17]. However, these VRFs are vulnerable to quantum computing attacks, as they rely on underlying assumptions that can be broken by a quantum adversary in polynomial time [Sho99]. Despite their versatility and significance, post-quantum VRFs are underdeveloped, with only five constructions out of four works to date [EKS+21, BDE+22, ESLR23, EEK+22]. The preliminary result of the lattice-based LB-VRF [EKS+21] provides limited residual pseudorandomness and requires updating the public key after, at most, five evaluations. Though it is sufficient in some scenarios, it cannot serve for long-term applications or on a large scale. The construction in [EEK+22] has the same limitation. Currently, only SL-VRF [BDE+22] from LowMC and the lattice-based LaV [ESLR23] offer full VRF capabilities. Regardless of the existence of Naor–Reingold-type PRFs (pseudorandom synthesizers [NR99]) [BPR12, Mon18] in lattices, the most versatile post-quantum branch, it seems challenging to push them forward to VRFs from this direction in a practical manner. Therefore, post-quantum VRFs have limited development, with only two full VRF proposals relying on a well-known assumption from symmetric primitives and lattices. Further research is necessary to address this challenge and further advance the capabilities of VRFs.

Isogeny-based cryptography, introduced relatively recently in comparison to other post-quantum branches, traces its origins to the CGL hash function [CLG09]. At its core, this cryptographic paradigm relies on the hardness of recovering an isogeny between two isogenous elliptic curves.

One of the most prominent isogeny-based cryptosystems is SIDH [JF11], a key exchange cryptosystem that relaxes the original isogeny assumption. The recent breakthroughs in the polynomial-timee SIDH attacks [CD23, MMP+23, Rob23] have marked significant advancements leading to the compromise of some relevant cryptosystems [YAJ+17, DDF+21]. Notwithstanding these developments, the original isogeny problem still remains as a solid foundation and several cryptosystems continue to be based on the original assumption [DKL+20, BCC+23].

A variant of isogeny-based cryptography takes the form of CSIDH, which is proposed by [CLM+18]. While it offers limited operations as the evaluation of the action is restricted to generating sets with small cardinality, it still results in the first secure and practical post-quantum non-interactive key exchange. In spite of a recognized subexponential vulnerability [Reg04, Kup05, Kup11, Pei20, BS20], recent research continues to demonstrate the versatility and competitivity of isogeny cryptography as a post-quantum branch, including signature schemes [BKV19, EKP20, DG19], UC-secure oblivious transfers [LGD21, BMM+22], theshold signatures [DM20], (linkable/accountable) ring and group signatures [BKP20, BDK+22], and PAKE [AEK+22]. Furthermore, recent optimization advancements [BKV19, DFK+23, CL23, PR23] are progressively enhancing the flexibility of the isogeny group action.

To have a VRF construction from isogenies is little to be known. Due to the less rich algebraic structure offered by the isogenies, translating classical constructions has shown to be a non-trivial task in general [BKP20, MOT20, LGD21, BDK+22]. For instance, the most practical classical counterpart ECVRF [PWH+17], based on a signature scheme with the unique signature property, requires hashing a string to a supersingular elliptic curve with unknown endomorphism ring, which is known to be a notorious bottleneck in isogeny-based cryptography [BBD+22, MMP22]. Also, the use of pairings, in [BGLS03, BLS01] for instance, could lead to a "partially post-quantum" only result as [DMPS19].

Hence, this leads to the central question of this work:

Can we have a post-quantum verifiable random functions from isogenies with an acceptable performance and the standard notions?

1.1 Related Works

To the best of our knowledge, there are four works to date [EKS+21, BDE+22, EEK+22, ESLR23] related to post-quantum VRFs. The lattice-based LB-VRF, X-VRF, iVRF in [EKS+21, BDE+22, EEK+22] provide compact proof sizes (0.6-7.3KB). Despite their limited number of evaluations, these constructions have proven to be effective in certain applications. iVRF, tailored to their applications, relaxes the unique provability (see CFU of [EEK+22] on P7) and leads to a compact proof size 0.6KB. On the other hand, SL-VRF and LaV in [BDE+22, ESLR23] provide full VRF capabilities from LowMC and the hybrid MSIS/MLWR respectively. They have proof sizes of 40, 12KB, and the secret key sizes of 24B and 6.4KB respectively.

In the field of isogeny cryptography, various protocols have been proposed that relate to random functions. For instance, Naor-Reingold type pseudorandom functions (PRF) have been proposed in [ADMP20, MOT20]. Additionally, there have been proposals for oblivious random functions using oblivious transfers with a Naor–Reingold-type PRF or one-more type assumptions [BKW20], however, the latter of which has been shown to be insecure [BKM+21]. Currently, a provably secure isogeny-based VRF has yet to be introduced in the literature.

Remark 1.1. *A recent work by Leroux introduces a novel VRF proposal from isogenies [Ler23], with a new application of Kani's criteria [Rob23]. The output of the evaluation is the hash value of a supersingular curve, computed through a large prime order isogeny. Remarkably, the proof involves a high-dimensional isogeny, resulting in the most compact VRF in the post-quantum literature. This scheme is based on a new one-more-type assumption of the high-dimensional isogeny computations, and time and effort are required to to establish its security. We also include its performance in Tab. 1 for comparison.*

1.2 Contributions

In this study, we present two VRFs, CAPYBARA and TSUBAKI[1], which provide an affirmative solution to the above question through the following three contributions.

1. Inspired and based on the Naor–Reingold pseudorandom function as in [ADMP20, BKW20, MOT20], we construct a proof system where the prover can demonstrate the knowledge of the action factorization of a set element based on a distinguished base point (see R_{fac} defined below). We use the technique from [BDK+22] to make the proof system online-extractable, providing tightly-secure unique provability. Additionally, we utilize the approach in [BKP20] to reduce the proof size. As a result, our VRFs have an exponentially large input space ($\{0,1\}^\lambda$) and expected proof sizes of 39KB and 34KB using CSIDH-512, which is comparable to the symmetric-primitive-based VRF [BDE+22]. The secret key can also be strored (compressed) as a 32B seed and generated efficiently usign PRNG on input of the seed.
2. We introduce a new decisional assumption, known as the master decisional Diffie-Hellman problem, which implies a variety of decisional problems. We show that it is as hard as the original DDH problem.
3. We show a new use of the quadratic twists (see Footnote 2) to expand the input space to be ternary ($\{-1,0,1\}^\kappa$). By using a similar method, we prove that this variant is as secure as the decisional square Diffie-Hellman problem, whose computational version is as hard as the group action inverse problem.

As a result, we introduce the first group action and isogeny-based VRFs in literature. CAPYBARA is based on the standard DDH assumption. Our method of construction and the techniques utilized are versatile and can be applied to other number-theoretic pseudorandom functions, demonstrating the promising potential of incorporating group actions and isogeny cryptography in the field of VRF research.

---

[1] Compact Action factorization Proofs Yielded By A RAndom function and Twist-SqaUre-BAsed tweaK from Isogenies.

1.3 Technical Overview

The ideas beneath this work are fairly simple. First, given a transitive and free (effective) group action $(G, \mathcal{E}, \star, h_0)$ for some distinguished element $h_0 \in \mathcal{E}$, we start from a Naor–Reingold-type pseudorandom function on input $x = (x_1 \cdots x_\kappa) \in \{0,1\}^\kappa$:

$$f(\mathsf{sk}, x) = (c_0 c_1 g_1^{x_1} \cdots g_\kappa^{x_\kappa}) \star h_0$$

where the secret key $\mathsf{sk} = (c_0, c_1, g_1, \cdots, g_\kappa)$ with the public key $\mathsf{vk} = (c_0 \star h_0, c_1 \star h_0, g_1 \star h_0, \cdots, g_\kappa \star h_0)$ as the evaluation of our verifiable random function. Remark that without c_1, it is a secure pseudorandom random function but not a secure verifiable random function since the adversary is given vk so that the evaluation at 0 is known.

Second, the factorization over the group $g = \Pi g_i$ (not necessarily unique) gives the factorization of $g \star h_0$ over the set with respect to h_0. We construct an action factorization proof system to prove the correctness of the evaluation of $f(\mathsf{sk}, x)$. Formally, let $h \leftarrow f(\mathsf{sk}, x)$ on input $x \in \{0,1\}^\kappa$. We consider the action factorization relation

$$R_{\mathsf{fac}} = \left\{ ((h_0, X_0, X_1, \{h_i\}_{i \in I}, h), (c_0, c_1, \{g_i\}_{i \in I})) \;\middle|\; \begin{array}{r} X_j = c_j \star h_0 \ \forall j \in \{0,1\} \\ g_i \star h_0 = h_i \ \forall i \in I \\ (c_0 c_1 \Pi_{i \in I} g_i) \star h_0 = h \end{array} \right\},$$

where $I = \{i \in [\kappa] | x_i = 1\}$. Notice that without h in the statement and the constraint, the proof system is trivial using a standard graph-isomorphism-type proof of knowledge in parallel. We show that one with the corresponding witness can prove a set element $h \in \mathcal{E}$ can be "factorized" through $\{h_i\}_{i \in I}$ and h_0 when the action is over an abelian group.

The three-move public-coin proof system starts from the prover who generates random $r, r_0, r_i \leftarrow G$ for $i \in I$, computes $(r \star X_0, r_0 \star X_1, \{r_i \star h_i\}_{i \in I}, (r r_0 \Pi_{i \in I} r_i) \star h) = (X'_0, X'_1, \{h'_i\}_{i \in I}, h')$, and sends it to the verifier. The verifier returns a random challenge $b \in \{0,1\}$ to the prover. Depending on b, the prover reveals $(r c_0^b, r_0 c_1^b, \{g_i^b r_i\}_{i \in I})$ to the verifier. Upon receiving $(r', r'_0, \{r'_i\}_{i \in I})$, if $b = 0$, the verifier checks whether $(r' \star X_0, r'_0 \star X_1, \{r'_i \star h_i\}_{i \in I}, (r' r'_0 \Pi_{i \in I} r'_i) \star h) = (X'_0, X'_1, \{h'_i\}_{i \in I}, h')$. If $b = 1$, the verifier checks whether $(r' \star h_0, r'_0 \star h_0, \{r'_i \star h_0\}_{i \in I}, (r' r'_0 \Pi_{i \in I} r'_i) \star h_0) = (X'_0, X'_1, \{h'_i\}_{i \in I}, h')$. The verifier accepts if it is the case or rejects otherwise. By λ times repetitions and applying the Fiat-Shamir transform, one can obtain NIZK for the relation R_{fac}. For the sake of clarity, we present the construction by assuming the group structure is known. We show in Rem. 4.1 that the construction is also feasible in the unknown group structure setting.

Third, instead of resorting to an ad-hoc assumption, we prove the residual pseudorandomness of our VRF is as hard as the decisional Diffie-Hellman problem. We first introduce a generalized decisional problem – the master decisional Diffie-Hellman problem. The problem starts with the challenger giving the adversary an instance $(g_1 \star h_0, \cdots, g_N \star h_0)$. The adversary can make queries for an arbitrary combination of $(g_{s_1} \cdots g_{s_k}) \star h_0$ for any $\{s_1, \cdots, s_k\} \subseteq [N]$, and also sends a challenge query, which has not been queried before. The challenger returns as instructed or a random set element from $\mathcal{E}$, and the adversary's task is to determine which is the case. The problem covers a variety of variants of group-action-based decisional problems. Then, we prove the problem is as hard as the original DDH problem.

Fourth, we make the proof compact and achieve online extractability. The latter notion gives a tight reduction for the full uniqueness where the adversary cannot forge two valid proofs on the same input for two distinct evaluations for any malicious generated keys without using a rewinding argument. To achieve online extractability, one can consider using Unruh's transform [Unr15] (or Pass' transform [Pas03] by hashing both responses and appending them to the commitment. This, however, will result in costly overhead. Instead, while running the proof above, the prover uses a seed and a pseudorandom number generator (PRNG) to generate the group elements $r, r_0, \{r_i\}_{i \in I}$. By employing the proof technique developed in [BDK$^+$22], the modification leads to an online-extractable proof system with much more compact proofs.

Fifth, as an independent interest in the CSIDH setting, we develop a new use of the quadratic twists and reduce the sizes of the public and secret keys and the computational cost for the user by relaxing the assumptions. In this way, the public key can be naturally expanded twice $(c_0 \star h_0, c_1 \star h_0, g_1 \star h_0, \cdots, g_\kappa \star h_0, (g_1 \star h_0)^t, \cdots, (g_\kappa \star h_0)^t)$.[2] The modification reduces 37% of the key size, the computational cost, and the maximal proof size. We prove that the underlying assumption for the residual pseudorandomness is as hard as the decisional square Diffie-Hellman problem in the appendix, of which the computational version is as hard as the group action inverse problem (i.e. Dlog).

Finally, we optimize the proof size again using the unbalanced challenge space and the seed trees introduced in [BKP20], which reduces the proof sizes of both constructions by a factor of 3. The proof sizes of our final VRFs are expected to be 39KB and 34KB when using CSIDH-512.

Roadmap. We begin in Sec. 2 with some preliminary backgrounds on sigma protocols and proof systems (Secs. 2.1 and 2.2), VRFs (Sec. 2.3), and group actions and hardness assumptions (Secs. 2.4 to 2.6). We then introduce our action factorization proof system in Sec. 4. We present our VRF constructions, CAPYBARA, in Sec. 5 and its variant, TSUBAKI, in Sec. 6. We show the underlying assumption of CAPYBARA (resp. TSUBAKI) is as hard as the DDH problem in Sec. 3 (resp. the decisional square DDH problem in App. A). Finally, we give the final optimization for both constructions and the performance comparison in Sec. 7.

2 Preliminaries

Notations. We denote $\{1, \cdots, N\} \subset \mathbb{N}$ by $[N]$. Say G acts on $\mathcal{E}$ by $\star$. For $\mathbf{v} = (a_1, \cdots, a_N) \in G^N$ and $\mathbf{e} = (E_1, \cdots, E_N) \in \mathcal{E}^N$, we extend the action to an arbitrary dimension by writing $\mathbf{v} \star \mathbf{e} = (a_1 \star E_1, \cdots, a_N \star E_N) \in \mathcal{E}^N$. We also abuse the notation $\mathbf{v} \star E = (a_1 \star E, \cdots, a_N \star E) \in \mathcal{E}^N$ when the context is clear. Also, $\mathbf{e}_i$ represents the i-th elementary vector where the i-th entry is 1 and the others are zeros. For an array $\mathbf{v} = (v_1, \cdots, v_N)$, we may denote the the i-th entry v_i as $\mathbf{v}_i$. For a subset $I \subseteq [N]$, we let $\mathbf{v}_I$ denote the sub-array $(v_i)_{i \in I}$.

Two probability ensembles X_λ, Y_λ are said to be computationally indistinguishable, denoted by $X_\lambda \approx_c Y_\lambda$, if for any PPT adversary $\mathcal{A}$ there exist a negligible function $\mathsf{negl}(\lambda)$ such $|\Pr[\mathcal{A}(X_\lambda) = 1] - \Pr[\mathcal{A}(Y_\lambda) = 1]| \leq \mathsf{negl}(\lambda)$. Also, X_λ, Y_λ, defined over the same set, are said to be statistically indistinguishable, denoted by $X_\lambda \approx_s Y_\lambda$, if there exists a negligible function $\mathsf{negl}(\lambda)$ such that $\sum_a |\Pr[X_\lambda = a] - \Pr[Y_\lambda = a]| \leq \mathsf{negl}(\lambda)$.

2.1 Sigma Protocol

Definition 2.1 (Sigma Protocol)**.** *A sigma protocol* Π_Σ *is a three-move proof system for a relation* R *consists of oracle-calling PPT algorithms* $(P = (P_1, P_2), V = (V_1, V_2))$*, where* V_2 *is deterministic. We assume* P_1 *and* P_2 *share states and so does* V_1 *and* V_2*. Let* ChSet *denote the challenge space. Then,* Π_Σ *proceeds as follows.*

- *The prover, on input* $(\mathsf{st}, \mathsf{wt}) \in R$*, runs* $\mathsf{com} \leftarrow P_1^{\mathcal{O}}(\mathsf{X}, \mathsf{W})$ *and sends a* commitment com *to the verifier.*
- *The verifier runs* $\mathsf{ch} \leftarrow V_1^{\mathcal{O}}(1^\lambda)$*, drawing a random challenge from* ChSet*, and sends it to the prover.*
- *The prover, given* ch*, runs* $\mathsf{resp} \leftarrow P_2^{\mathcal{O}}(\mathsf{X}, \mathsf{W}, \mathsf{ch})$ *and returns a* response resp *to the verifier.*
- *The verifier runs* $V_2^{\mathcal{O}}(\mathsf{X}, \mathsf{com}, \mathsf{ch}, \mathsf{resp})$ *and outputs* $\top$ *(accept) or* $\bot$ *(reject).*

---

[2] Remark the reduction of the key size comes in different flavors in contrast to [BKV19, EKP20] where the twist reduces the public key size by decreasing the soundness error of the sigma protocol. Here, the twist decreases the key size by expanding a binary input to a ternary input instead of benefiting the proof system. The proof system is still BINARY challenge in this construction.

Here, $\mathcal{O}$ is modeled as a random oracle. For simplicity, we often drop $\mathcal{O}$ from the superscript when it is clear from the context. We assume the statement st *is always given as input to both the prover and the verifier. The protocol* transcript $(\mathsf{com}, \mathsf{ch}, \mathsf{resp})$ *is said to be valid in case* $V_2(\mathsf{com}, \mathsf{ch}, \mathsf{resp})$ *outputs* $\top$*.*

We require the sigma protocol to be correct conditioned on the prover not aborting the protocol. Below, if $\delta = 0$, then it corresponds to the case when the prover never aborts.

Definition 2.2 (Correctness)**.** *A sigma protocol Π_Σ is said to be correct if for all $\lambda \in \mathbb{N}$, $(\mathsf{st}, \mathsf{wt}) \in R$ and the prover and the verifier both follow the protocol specification, the verifier always outputs $\top$.*

Definition 2.3 (High Min-Entropy)**.** *We say a sigma protocol Π_Σ has $\alpha(\lambda)$ min-entropy if for any $\lambda \in \mathbb{N}$, $(\mathsf{st}, \mathsf{wt}) \in R$, and a possibly computationally-unbounded adversary $\mathcal{A}$, we have*

$$\Pr\left[\mathsf{com} = \mathsf{com}' \middle| \mathsf{com} \leftarrow P_1^{\mathcal{O}}(\mathsf{st}, \mathsf{wt}), \mathsf{com}' \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{st}, \mathsf{wt})\right] \leq 2^{-\alpha},$$

where the probability is taken over the randomness used by P_1 and by the random oracle. We say Π_Σ has high min-entropy *if $2^{-\alpha}$ is negligible in λ.*

Definition 2.4 (Honest Verifier Zero-Knowledge)**.** *We say Π_Σ is* honest-verifier-zero-knowledge *for relation R if there exists a PPT simulator $\mathsf{Sim}^{\mathcal{O}}$ with access to a random oracle $\mathcal{O}$ such that any statement-witness pair $(\mathsf{st}, \mathsf{wt}) \in R$, $\mathsf{ch} \in \mathsf{ChSet}$, $\lambda \in \mathbb{N}$ and any computationally-unbounded adversary $\mathcal{A}$ that makes at most a polynomial number of queries to $\mathcal{O}$, we have*

$$\mathsf{Adv}^{\mathsf{HVZK}}_{\Pi_\Sigma}(\mathcal{A}) := \left| \Pr[\mathcal{A}^{\mathcal{O}}(P^{\mathcal{O}}(\mathsf{st}, \mathsf{wt}, \mathsf{ch})) = 1] - \Pr[\mathcal{A}^{\mathcal{O}}(\mathsf{Sim}^{\mathcal{O}}(\mathsf{st}, \mathsf{ch})) = 1] \right| = \mathsf{negl}(\lambda),$$

where $P = (P_1, P_2)$ is a prover running on $(\mathsf{st}, \mathsf{wt})$ with a challenge fixed to ch and the probability is taken over the randomness used by (P, V) and by the random oracle.

Definition 2.5 (Special Soundness)**.** *We say a sigma protocol Π_Σ has special soundness if there exists a polynomial-time extraction algorithm* $\mathsf{Extract}$ *such that, given a statement* st *and any two valid transcripts* $(\mathsf{com}, \mathsf{ch}, \mathsf{resp})$ *and* $(\mathsf{com}, \mathsf{ch}', \mathsf{resp}')$ *relative to* st *and such that* $\mathsf{ch} \neq \mathsf{ch}'$*, outputs a witness* wt *satisfying* $(\mathsf{st}, \mathsf{wt}) \in R$*.*

2.2 Proof System Under the Random Oracle Model

Definition 2.6 (Completeness)**.** *Let $\mathcal{O}$ be a random oracle and $\Pi_{\mathsf{NIZK}} = (\mathsf{Prove}, \mathsf{Verify})$ a* NIZK *proof system for a relation R. We say Π_{NIZK} for a relation R is complete if for all $\lambda \in \mathbb{N}$, $(\mathsf{st}, \mathsf{wt}) \in R$ and the prover and the verifier both follow the protocol specification, the verifier always accepts.*

Definition 2.7 (Zero-Knowledge)**.** *Let $\mathcal{O}$ be a random oracle, $\Pi_{\mathsf{NIZK}} = (\mathsf{Prove}, \mathsf{Verify})$ a* NIZK *proof system for a relation R, and* Sim *a zero-knowledge simulator with access to $\mathcal{O}$ for Π_{NIZK}. For $(\mathsf{st}, \mathsf{wt}) \in R$, the advantage of an zero-knowledge adversary $\mathcal{A}$ against* Sim *is*

$$\mathsf{Adv}^{\mathsf{ZK}}_{\Pi_{\mathsf{NIZK}}}(\mathcal{A}) = \left| \Pr\left[\mathcal{A}^{\mathcal{O}}(P^{\mathcal{O}}(\mathsf{st}, \mathsf{wt})) = 1\right] - \Pr\left[\mathcal{A}^{\mathcal{O}}(\mathsf{Sim}^{\mathcal{O}}(\mathsf{st})) = 1\right] \right|,$$

We say Π_{NIZK} is zero-knowledge if there exists a PPT simulator Sim *such that for any $(\mathsf{st}, \mathsf{wt}) \in R$, (possibly computationally-unbounded) adversary $\mathcal{A}$ making at most polynomially many queries to the random oracle, we have a negligible function* $\mathsf{negl}(\lambda)$ *such that* $\mathsf{Adv}^{\mathsf{ZK}}_{\Pi_{\mathsf{NIZK}}}(\mathcal{A}) \leq \mathsf{negl}(\lambda)$*.*

Definition 2.8 (Online Extractability)**.** *Let Π_{NIZK} be a NIZK proof system for a relation R. We said Π_{NIZK} has online-extractability if for any (possibly computationally-unbounded) adversary $\mathcal{A}$, there exists a PPT extractor* Ext *with only extractability access to $\mathcal{O}$ such that $\mathcal{A}$ wins the following game with a negligible advantage:*

(i) $\mathcal{A}$ can make polynomial number queries of the random oracle.

(ii) $\mathcal{A}$ outputs st *and π.*

We say $\mathcal{A}$ wins if $\mathsf{Verify}^{\mathcal{O}}(\mathsf{st}, \pi) = \top$ *and* $(\mathsf{st}, \mathsf{wt}) \notin R$ *where* $\mathsf{wt} \leftarrow \mathsf{Ext}(\mathsf{st}, \pi)$*. The advantage of $\mathcal{A}$ is defined as* $\mathsf{Adv}^{\mathsf{OE}}_{\Pi_{\mathsf{NIZK}}}(\mathcal{A}) = \Pr[\mathcal{A}$ *wins*$]$ *where the probability is taken over the randomness used by the random oracle.*

2.3 Verifiable Random Functions

In this subsection, we give a brief introduction to the verifiable random functions, and their notions [MRV99].

Definition 2.9. *(Verifiable Random Function) A* verifiable random function (VRF) *consists of four probabilistic polynomial-time algorithms* $\Pi_{\mathsf{VRF}} = \{\mathsf{ParGen}, \mathsf{KeyGen}, \mathsf{VRFEval}, \mathsf{Ver}\}$ *where:*

- $\mathsf{ParGen}(1^\lambda)$*: On input a security parameter* 1^λ*, this probabilistic algorithm outputs some global, public parameter* pp.
- $\mathsf{KeyGen}(\mathsf{pp})$*: On input public parameter* pp*, this probabilistic algorithm outputs two binary strings, a secret key* sk *and a public key* vk.
- $\mathsf{VRFEval}(\mathsf{sk}, x)$*: On input a secret key sk and an input* $x \in \{0,1\}^{\ell(\lambda)}$*, this algorithm outputs* (v, π) *for the VRF value* $v \in \{0,1\}^{m(\lambda)}$ *and the corresponding proof* π *proving the correctness of* v.
- $\mathsf{Ver}(\mathsf{vk}, v, x, \pi)$*: On input* (vk, v, x, π)*, this probabilistic algorithm outputs either* 1 *or* 0.

The residual pseudorandomness guarantees the pseudorandomness of the function even if the user has revealed many evaluations together with the proofs. In some applications, it is sufficient to have a few-times relaxed notion where the pseudorandomness is ensured for only limited copies of evaluations are revealed [EKS$^+$21]. In this work, we consider the original version of the notion.

Definition 2.10. *((Residual) Pseudorandomness) Let* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *be a PPT adversary. The pseudorandomness experiment* $\mathsf{ExpVRF}^{\mathsf{PR}}_{\mathcal{A},\Pi_{\mathsf{VRF}}}(\lambda)$ *of a VRF scheme* Π_{VRF} *proceeds as follows.*

1. $Q \leftarrow \emptyset$
2. $\mathsf{pp} \leftarrow \mathsf{ParGen}(1^\lambda)$
3. $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$
4. $(\tilde{x}, st) \leftarrow \mathcal{A}_1^{\mathcal{O}_{\mathsf{VRFEval}}(\cdot)}(\mathsf{vk})$
5. $(v_0, \pi_0) \leftarrow \mathsf{VRFEval}(\mathsf{sk}, \tilde{x})$
6. $v_1 \leftarrow \{0,1\}^{m(\lambda)}$
7. $b \leftarrow \{0,1\}$
8. $b' \leftarrow \mathcal{A}_2^{\mathcal{O}_{\mathsf{VRFEval}}(\cdot)}(v_b, st)$
9. *The output of the experiment is defined to be 1 if* $b' = b$ *and* $\tilde{x} \notin Q$*, and 0 otherwise.*

$\mathcal{O}_{\mathsf{VRFEval}}(x)$:

1. $Q \leftarrow Q \cup \{x\}$
2. Return $\mathsf{VRFEval}(\mathsf{sk}, x)$

We say $\mathcal{A}$ *wins if* $\mathsf{ExpVRF}^{\mathsf{PR}}_{\mathcal{A},\Pi_{\mathsf{VRF}}}(\lambda) = 1$*. The advantage of* $\mathcal{A}$ *is defined to be*

$$\mathsf{Adv}^{\mathsf{PR}}_{\Pi_{\mathsf{VRF}}}(\mathcal{A}) := \left|\Pr\left[\mathcal{A}\ wins\right] - 1/2\right|,$$

where the probability is taken over the randomness used by $\mathcal{A}$ *and the randomness used in the experiment. A VRF protocol* Π_{VRF} *is said to be* pseudorandom *if for any PPT adversary* $\mathcal{A}$ *there exists a negligible function* negl *such that*

$$\mathsf{Adv}^{\mathsf{PR}}_{\Pi_{\mathsf{VRF}}}(\mathcal{A}) \leq \mathsf{negl}(\lambda).$$

Definition 2.11. *(Complete Provability) Let* $\Pi_{\mathsf{VRF}} = \{\mathsf{ParGen}, \mathsf{KeyGen}, \mathsf{VRFEval}, \mathsf{Ver}\}$ *be a VRF scheme.* Π_{VRF} *is said to have* provability *if for any* $\mathsf{pp} \leftarrow \mathsf{ParGen}(1^\lambda)$ *and* $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$*, the output* $(v, \pi) \leftarrow \mathsf{VRFEval}(\mathsf{sk}, x)$ *satisfies*

$$\mathsf{Ver}(\mathsf{vk}, v, x, \pi) = 1.$$

The following notion, unique provability, implies that for any adversary (possibly computationally unbounded with at most polynomial public coin queries) it is difficult to generate a malicious public key such that the adversary can produce two valid proofs for two distinct evaluations of the same input.

Definition 2.12. *(Unique Provability) Let* $\Pi_{\mathsf{VRF}} = \{\mathsf{ParGen}, \mathsf{KeyGen}, \mathsf{VRFEval}, \mathsf{Ver}\}$ *be a VRF scheme and* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *be an adversary. A* uniqueness provability *experiment proceeds as follows.*

1. $\mathsf{pp} \leftarrow \mathsf{ParGen}(1^\lambda)$
2. $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathcal{A}_1(\mathsf{pp})$
3. $(\mathsf{vk}, x, v_1, v_2, \pi_1, \pi_2) \leftarrow \mathcal{A}_2(\mathsf{vk})$

We say an adversary $\mathcal{A}$ *wins if* $v_1 \neq v_2$ *and* $\mathsf{Ver}(\mathsf{vk}, v_1, x, \pi_1) = \mathsf{Ver}(\mathsf{vk}, v_2, x, \pi_2) = 1$*. The advantage of* $\mathcal{A}$ *is defined to be* $\mathsf{Adv}^{\mathsf{UP}}_{\Pi_{\mathsf{VRF}}}(\mathcal{A}) := \Pr[\mathcal{A} \text{ wins}]$ *where the probability is taken over the randomness used by* $\mathcal{A}$ *and in the experiment. A VRF protocol* Π_{VRF} *is said to be* uniqueness provability *if for any PPT adversary* $\mathcal{A}$ *there exists a negligible function* negl *such that*

$$\mathsf{Adv}^{\mathsf{UP}}_{\Pi_{\mathsf{VRF}}}(\mathcal{A}) \leq \mathsf{negl}(\lambda).$$

2.4 Group Actions

Throughout this work *we consider only free, transitive and effective group action.* In this section, we give a brief introduction to the main component in our protocols – the group actions.

Definition 2.13 (Group Action)**.** *A group* G *is said to act on a set* $\mathcal{E}$ *if there is a map* $\star : G \times \mathcal{E} \to \mathcal{E}$ *that satisfies the*

1. *Identity: if* 1 *is the identity element of* G*, then for any* $E \in \mathcal{E}$*, we have* $1 \star E = E$*.*
2. *Compatibility: for any* $g, h \in G$ *and any* $E \in \mathcal{E}$*, we have* $(gh) \star E = g \star (h \star E)$*.*

For the cryptographic purpose, we need the following propositions.

Definition 2.14. *A group action* $(G, \mathcal{E}, \star)$ *is said to be*

1. *transitive if for any* $x_1, x_2 \in \mathcal{E}$ *there exists* $g \in G$ *such that* $x_2 = g \star x_1$*, or*
2. *free if for any* $g \in G$*,* g *is the identity element if and only if there exists some* $x \in \mathcal{E}$ *such that* $x = g \star x$*.*

For constructing a feasible construction from a group action, we require some efficient (PTT) algorithms. We adopt the *effective group action* framework introduced in [ADMP20].

Definition 2.15 (Effective Group Action)**.** *A group action* $(G, \mathcal{E}, E_0, \star)$ *is effective if the following properties are satisfied:*

1. *The group* G *is finite and there exist PPT algorithms for (i.) the membership testing, (ii.) equality testing, (iii.) group operations, (iv.) element inversions, and (v.)a sampling method over* G*. The sampling method is required to be statistically indistinguishable from the uniform distribution over* G*.*
2. *The set* $\mathcal{E}$ *is finite, and there exist PPT algorithms for the membership testing and generating a unique bit-string representation for every element in* $\mathcal{E}$*.*
3. *There exists a distinguished element* $E_0 \in \mathcal{E}$ *and the bit-string representation is publicly known.*
4. *There exists a PPT algorithm that given any* $(g, x) \in G \times \mathcal{E}$ *outputs* $g \star x$*.*

Post-quantum instantiations of EGA currently rely exclusively on isogenies. However, recent works [Pei20, BS20, CSCJR22] reveal that the existing EGA instances [BKV19, DFK$^+$23] fall short of meeting the post-quantum NIST 1 security level. Moreover, Section 2.5 of [Lai23] and [Pan23] show that in order to obtain EGA for a more robust parameter, evaluating isogenies in the way described in [DF19] is not polynomial-time in theory (even with the preprocessing using quantum computers) but may also be slow in practice, which is evaluated using the lattice heuristics. The ongoing pursuit of a more robust parameter set for isogeny-based EGA remains an active focus in research [DFK$^+$23, CL23, PR23], which is out of the scope of this paper.

Within this model, diverse constructions have be proposed, including logarithmic (linkable) ring signatures [BKP20], threshold signatures [DM20], logarithmic (and tightly secure) group signatures [BDK$^+$22], and compact blind signatures [KLLQ23].

Remark 2.16 (**Additional Requirements.**)**.** *We have two additional requirements for our group actions. Firstly, for security parameter λ, we require the group size $|G|$ to be larger than 2^λ. The requirement naturally holds due to the known quantum subexponential attacks $2^{O(\sqrt{|G|})}$ [Reg04, Kup05, Kup11, Pei20, BS20]. This is necessary to ensure that we have adequate min-entropy for our proof system in Sec. 4. The second requirement is that every G has a unique representation, which can be efficiently computed. The requirement is directly implied by the known-order effective group (KEGA) model [ADMP20]. We do not adopt the model since we use neither the group's structure nor the group's order (KEGA). With this assumption, we can ensure that revealing $g + g'$ will not leak the information of g where g' is sampled uniformly from G for our proof system in Sec. 4. It is worth noting that this requirement is for simplicity of presentation and is not strictly necessary (see next remark).*

Remark 2.17. *For the sake of clarity, we present the work using the EGA model. A weaker version (restricted effective group action) restricted the feasible evaluation of the action to a generating set of small cardinality (e.g. the original CSIDH setting [CLM$^+$18, DG19]). Our construction can also be realized with a few modifications for the proof system, requiring Fiat-Shamir with aborts [Lyu09, DG19]. We give a brief discussion in Rem. 4.1.*

Throughout this work, we assume the action is always *free, transitive and effective* and denote it by a tuple $(G, \mathcal{E}, E_0, \star)$ where E_0 is the distinguished element. Also, we assume the sampling method over G is uniform.

In our second construction, we require a special operation–the quadratic twist. In the CSIDH group action $(G, \mathcal{E})$ [CLM$^+$18], when the prime equals 3 modulo 4, there exists a special operation, the quadratic twist t, such that for any $E \in \mathcal{E}$, we have $E^t \in \mathcal{E}$, and has the proposition $(g \star E)^t = g^{-1} \star E^t$. Also, there exists a special element E_0, usually used as the distinguished element in the literature, of j-invariant 1728, satisfies $(E_0)^t = E_0$. The quadratic twist has been shown to be a useful tool in some cryptosystems [BKV19, EKP20, LGD21, AEK$^+$22].

We will only need the twist operation in Secs. 2.6 and 6, and we will declare this at the beginning of the sections.

2.5 Hardness Assumptions of Group Actions for CAPYBARA

In this subsection, we introduce a few standard assumptions in group actions. We start from two computational assumptions, which we will not use in our construction, but it is helpful to understand the hierarchy of the decisional versions.

Definition 2.18 (Group Action Inverse Problem (GAIP))**.** *Let $(G, \mathcal{E}, \star, E_0)$ be a group action. Given E sampled from the uniform distribution over $\mathcal{E}$, the* GAIP *problem consists in finding an element $g \in G$ such that $g \star E_0 = E$.*

Definition 2.19 (Computational Diffie-Hellman (CDH) Problem)**.** *Let $(G, \mathcal{E}, \star, E_0)$ be a group action. Given a tuple $(g_1 \star E_0, g_2 \star E_0)$ where g_1, g_2 are sampled uniformly from G, the computational Diffie-Hellman problem is to compute $(g_1 g_2) \star E_0$.*

The following is the core hardness assumption for our first VRF in Sec. 5.

Definition 2.20 (Decisional Diffie-Hellman (DDH) Problem)**.** *Let* $(G, \mathcal{E}, \star, E_0)$ *be a group action. The decisional Diffie-Hellman problem is that the adversary* $\mathcal{A}$ *is given one instance of* $T_b = (g_1 \star E_0, g_2 \star E_0, h_b \star E_0)$ *where* $h_0 = g_1 g_2, h_1 = g_3$ *and* $g_1, g_2, g_3, b \leftarrow G^3 \times \{0, 1\}$ *and output* $b' \in \{0, 1\}$*.*

We denote the advantage of the decisional problem adversary $\mathcal{A}$ by

$$\mathsf{Adv}^{\mathsf{DDH}}(\mathcal{A}) = |\Pr[\mathcal{A}(T_0) \to 1] - \Pr[\mathcal{A}(T_1) \to 1]|,$$

where b is the randomness in the experiment, and the probability is taken over the randomness used by $\mathcal{A}$ and the randomness used in the experiment. The group action $(G, \mathcal{E}, \star, E_0)$ is implicitly parameterized in the experiment. We say the DDH problem is hard, if for any PPT adversary $\mathcal{A}$, there exists a negligible function negl such that $\mathsf{Adv}^{\mathsf{DDH}}(\mathcal{A}) \leq \mathsf{negl}(\lambda)$.

Note that when using CSIDH as an instance, we require $p = 3 \pmod 4$ to avoid the attacks presented in [CSV20, CHVW22] exploiting distinct pairings. Both attacks rely on the nontrivial characters derived from the nontrivial 2-torsion subgroup in the ideal class group, which is not the case when $p = 3 \pmod 4$. Therefore, when CSIDH is instantiated in this setting, DDH is believed to be hard.

Definition 2.21 (Multi-Challenge Decisional Diffie-Hellman (mcDDH) Problem)**.** *Let* $(G, \mathcal{E}, \star, E_0)$ *be a group action and* $b \in \{0, 1\}$*. The multi-challenge decisional Diffie-Hellman experiment* $\mathsf{Exp}^{\mathsf{mcDDH}}(b)$ *on input* b *proceeds as follows. The adversary* $\mathcal{A}$ *is given* $(g_1 \star E_0)$ *where* $g_1 \leftarrow G$ *together with access to the oracle* $\mathcal{O}_b^{\mathsf{mcDDH}}$ *defined as follows:*

1. $\mathcal{O}_0^{\mathsf{mcDDH}}$*:* $(g_2 \star E_0, (g_1 g_2) \star E_0)$ *where* g_2 *are sampled uniformly from* G*,*
2. $\mathcal{O}_1^{\mathsf{mcDDH}}$*:* $(g_2 \star E_0, g_3 \star E_0)$ *where* g_2, g_3 *are sampled uniformly from* G*,*

and outputs $b' \in \{0, 1\}$*.*

We denote the advantage of a multi-challenge decisional Diffie-Hellman problem adversary $\mathcal{A}$ problem by

$$\mathsf{Adv}^{\mathsf{mcDDH}}(\mathcal{A}) = \left| \Pr[\mathcal{A}(\mathsf{Exp}^{\mathsf{mcDDH}}(b = 0)) \to 1] - \Pr[\mathcal{A}(\mathsf{Exp}^{\mathsf{mcDDH}}(b = 1)) \to 1] \right|,$$

where b is the randomness in the experiment, and the probability is taken over the randomness used by $\mathcal{A}$ and the randomness used in the experiment. The group action $(G, \mathcal{E}, \star, E_0)$ is implicitly parameterized in the experiment. We say the mcDDH problem is hard, if for any PPT adversary $\mathcal{A}$, there exists a negligible function negl such that $\mathsf{Adv}^{\mathsf{mcDDH}}(\mathcal{A}) \leq \mathsf{negl}(\lambda)$. One can use a standard hybrid argument and give a reduction from the DDH problem to the mcDDH problem.

A standard hybrid argument can lead to a reduction looseness that is proportional to the number of queries made. The equivalence is tight in the classical setting (i.e. the group setting) due to the randomizer introduced [NP01] which can keep regenerating a DH instance or a random instance depending on the input instance. Achieving a tight equivalence in the group action setting remains an open problem.

We introduce a generalized version of the decisional problem – the master decisional problem, analogue to the generalized DDH assumption [BLMW07] and similar to the Uber-family assumptions [Boy08]. In the master decisional problem, the starting instance consists of several random set elements, and the adversary can query any combination of them with respect to the group elements. We will show that the generalized version is as hard as the DDH problem using a hybrid argument.

Definition 2.22 (Master Decisional Diffie-Hellman (MDDH) Problem)**.** *Let* $(G, \mathcal{E}, \star, E_0)$ *be a group action,* $n \in \mathbb{N}$*, and* $b \in \{0, 1\}$*. The decisional master Diffie-Hellman problem experiment* $\mathsf{Exp}^{\mathsf{MDDH}}(n, b)$ *on input* (n, b) *proceeds as follows.*

1. *The challenger* $\mathcal{C}$ *generates a tuple* $(g_1 \star E_0, \cdots, g_n \star E_0)$ *where* $g_1, \cdots, g_n \leftarrow G$*, and sends the tuple to the adversary* $\mathcal{A}$*.*
2. $\mathcal{A}$ *is given access to a Diffie-Hellman (DH) oracle on input* $(x_1, \cdots, x_n) \in \{0,1\}^n$ *returning* $\prod_i^n g_i^{x_i} \star E_0$*.*
3. $\mathcal{A}$ *sends a string* $v = (v_1, \cdots, v_n) \in \{0,1\}^n$ *to the challenge oracle* $\mathcal{C}$*.*
4. $\mathcal{C}$ *ignores if* v *has been queried before or is of Hamming weight less than* 2*. Otherwise,* $\mathcal{C}$*, depending on the input* b*, computes* $X_0 = \prod_i^n g_i^{v_i} \star E_0$ *or* $X_1 = r \star E_0$ *for some* $r \leftarrow G$*, and sends* X_b *to* $\mathcal{A}$*. This process will only output for one time.*
5. $\mathcal{A}$ *outputs* $b' \in \{0,1\}$*.*

We denote the advantage of a decisional master Diffie-Hellman problem adversary $\mathcal{A}$ by

$$\mathsf{Adv}^{\mathsf{MDDH}}(\mathcal{A}) = \left|\Pr[\mathcal{A}(\mathsf{Exp}^{\mathsf{MDDH}}(n, b=0)) \to 1] - \Pr[\mathcal{A}(\mathsf{Exp}^{\mathsf{MDDH}}(n, b=1)) \to 1]\right|,$$

where b is the randomness in the experiment, and the probability is taken over the randomness used by $\mathcal{A}$ and the randomness used in the experiment. The group action $(G, \mathcal{E}, \star, E_0)$ is implicitly parameterized in the experiment. We say the MDDH problem is hard, if for any PPT adversary $\mathcal{A}$, there exists a negligible function negl such that $\mathsf{Adv}^{\mathsf{MDDH}}(\mathcal{A}) \leq \mathsf{negl}(\lambda)$.

The assumption implies a variety of forms of decisional problems. For instance, given $(a \star x, b \star x, c \star x, ab \star x, bc \star x, cd \star x)$ to distinguish between $abc \star x$ or a random element in $\mathcal{E}$ is an instance of the problem. The interactivity of the assumption appears to be strange at a glance. It is, however, very reasonable. Otherwise, when n is linear in λ, giving all combinations implies revealing almost the entire set $\mathcal{E}$. Looking ahead, we will use this problem to show our verifiable random function has residual pseudorandomness. Unlike pseudorandomness, where the adversary has access to either the pseudorandom function or a random function, the MDDH experiment allows the adversary to learn the evaluations of any combination of the instances adaptively. We show in Sec. 3 the equivalence of the master DDH and the original DDH.

2.6 Relaxed Decisional Assumptions for CSIDH-based Actions for TSUBAKI

This section introduces a few relaxed decisional assumptions that allow us to construct a more efficient verifiable random function variant. We use the quadratic twists in this section, and for a group action $(G, \mathcal{E}, \star, E_0)$ we let $E_0 \in \mathcal{E}$ denote the element that has the property that $E_0^t = E_0$. Also, for any $(g, E) \in G \times \mathcal{E}$, we have $(g \star E)^t = g^{-1} \star E^t$.

Firstly, we relax the DDH problem by introducing the standard square variant problem. The problem has been used to construct some cryptographic protocols [DM20, AEK+22]. A very recent work [DHK+23] justifies the hardness of the assumption in a generic model for group actions.

Definition 2.23 (Decisional Square CSIDH (sDDH) Problem)**.** *Let* $(G, \mathcal{E}, \star, E_0)$ *be a group action. The decisional square CSIDH problem is that the adversary* $\mathcal{A}$ *is given* $T_b = (g_1 \star E_0, h_b \star E_0)$ *where* $h_0 = g_1^2, h_1 = g_2$ *and* $(g_1, g_2, b) \leftarrow G^2 \times \{0,1\}$ *and return* $b' \in \{0,1\}$*.*

We denote the advantage of an sDDH adversary $\mathcal{A}$ by

$$\mathsf{Adv}^{\mathsf{sDDH}}(\mathcal{A}) = |\Pr[\mathcal{A}(T_0) \to 1] - \Pr[\mathcal{A}(T_1) \to 1]|,$$

where b is the randomness in the experiment, and the probability is taken over the randomness used by $\mathcal{A}$ and the randomness used in the experiment. The group action $(G, \mathcal{E}, \star, E_0)$ is implicitly parameterized in the experiment. We say the sDDH problem is hard, if for any PPT adversary $\mathcal{A}$, there exists a negligible function negl such that $\mathsf{Adv}^{\mathsf{sDDH}}(\mathcal{A}) \leq \mathsf{negl}(\lambda)$.

The computational version of the problem is quantum equivalent to the computational problem [LGD21], and quantum equivalent to the GAIP problem [GPSV21]. A full quantum equivalence is given in [MZ22]. One can reduce the sDDH problem to the DDH problem by mapping the instance $(g_1 \star E_0, h_b \star_0)$ to $(g_1 \star E_0, (gg_1) \star E_0, (gh_b) \star E_0)$ where $g \leftarrow G$. Though the reverse reduction is not known, sDDH is still believed to be a hard problem.

We introduce the decisional assumptions for our VRF variant where the input is ternary from $\{-1, 0, 1\}$, naturally corresponding to the following queries.

Definition 2.24 (Twisted Master Decisional CSIDH (tMDDH) Problem)**.** *Let $(G, \mathcal{E}, \star, E_0)$ be a group action, $n \in \mathbb{N}$, and $b \in \{0, 1\}$. The twisted master DDH problem experiment $\mathsf{Exp}^{\mathsf{tMDDH}}(n, b)$ on input (n, b) proceeds as follows.*

1. *The challenger $\mathcal{C}$ computes $E = g \star E_0$ where $g \leftarrow G$.*
2. *$\mathcal{C}$ generates a tuple $(g_1 \star E, \cdots, g_n \star E)$ where $g_1, \cdots, g_n \leftarrow G$, and sends the tuple to the adversary $\mathcal{A}$.*
3. *$\mathcal{A}$ is given access to a Diffie-Hellman (DH) oracle on input $(x_1, \cdots, x_n) \in \{0, \pm 1\}^n$ returning $\prod_i^n g_i^{x_i} \star E$.*
4. *$\mathcal{A}$ sends a string $v = (v_1, \cdots, v_n) \in \{0, \pm 1\}^n$ to $\prod_i^n g_i^{v_i} \star E$ to the challenge oracle $\mathcal{C}$.*
5. *$\mathcal{C}$ ignores if v has been queried before or is of Hamming weight less than 2. Otherwise, $\mathcal{C}$, depending on b, computes $X_0 = \prod_i^n g_i^{v_i} \star E$ or $X_1 = r \star E$ for some $r \leftarrow G$, and send X_b to $\mathcal{A}$. This process will only output for one time.*
6. *$\mathcal{A}$ outputs $b' \in \{0, 1\}$.*

We denote the advantage of the decisional problem adversary $\mathcal{A}$ by

$$\mathsf{Adv}^{\mathsf{tMDDH}}(\mathcal{A}) = \left| \Pr[\mathcal{A}(\mathsf{Exp}^{\mathsf{tMDDH}}(n, b = 0)) \to 1] - \Pr[\mathcal{A}(\mathsf{Exp}^{\mathsf{tMDDH}}(n, b = 1)) \to 1] \right|,$$

where b is the randomness in the experiment, and the probability is taken over the randomness used by $\mathcal{A}$ and the randomness used in the experiment. The group action $(G, \mathcal{E}, \star, E_0)$ is implicitly parameterized in the experiment. We say the tMDDH problem is hard, if for any PPT adversary $\mathcal{A}$, there exists a negligible function negl such that $\mathsf{Adv}^{\mathsf{tMDDH}}(\mathcal{A}) \leq \mathsf{negl}(\lambda)$.

We show in App. A that the twisted decisional master CSIDH problem is not easier than the decisional square CSIDH problem. To see this, we are introducing a non-standard intermediate assumption, which will make the proof easier to follow. The assumption coincides with a decisional version of a problem proposed in [LGD21].

Definition 2.25 (Decisional Reciprocal CSIDH (rDDH) Problem)**.** *Let $(G, \mathcal{E}, \star, E_0)$ be a group action. The decisional reciprocal CSIDH problem is that the adversary $\mathcal{A}$ is given $T_b = (g_1 \star E_0, g_2 \star E_0, h_b \star E_0, h'_b \star E_0)$ where $h_0 = g_1 g_2, h_1 = g_3, h'_0 = g_1 g_2^{-1}, h'_1 = g_4$ and $(g_1, g_2, g_3, g_4, b) \leftarrow G^4 \times \{0, 1\}$, and return $b' \in \{0, 1\}$.*

We denote the advantage of an rDDH adversary $\mathcal{A}$ by

$$\mathsf{Adv}^{\mathsf{rDDH}}(\mathcal{A}) = \left| \Pr[\mathcal{A}(T_0) \to 1] - \Pr[\mathcal{A}(T_1) \to 1] \right|,$$

where b is the randomness in the experiment, and the probability is taken over the randomness used by $\mathcal{A}$ and the randomness used in the experiment. The group action $(G, \mathcal{E}, \star, E_0)$ is implicitly parameterized in the experiment. We say the rDDH problem is hard, if for any PPT adversary $\mathcal{A}$, there exists a negligible function negl such that $\mathsf{Adv}^{\mathsf{rDDH}}(\mathcal{A}) \leq \mathsf{negl}(\lambda)$.

The computational version proposed in [LGD21] has been proven to be equivalent to the computation square CDH problem, which is equivalent to the GAIP problem. The following proposition shows that the decisional reciprocal problem is not easier than the decisional square problem. In the appendix App. A, we will use the multi-challenge version of the decisional reciprocal problem to show the hardness of the twisted decisional master problem.

Proposition 2.26. *Let* $(G, \mathcal{E}, \star, E_0)$ *be a group action. Given an adversary* $\mathcal{A}$ *against the* rDDH *problem, there exists an* sDDH *adversary* $\mathcal{B}_1$ *and a decisional CSIDH problem* $\mathcal{B}_2$ *such that*

$$\mathsf{Adv}^{\mathsf{rDDH}}(\mathcal{A}) \leq \mathsf{Adv}^{\mathsf{sDDH}}(\mathcal{B}_1) + \mathsf{Adv}^{\mathsf{DDH}}(\mathcal{B}_2).$$

Proof. We prove this by introducing a sereis of hybrid games $\mathsf{Game}_1, \mathsf{Game}_2, \mathsf{Game}_3$ by gradually changing the experiment, where Game_1 corresponds to the case of $b = 0$ in the experiment (Def. 2.25) and Game_3 corresponds to the case $b = 1$.

Game_2 : the same as Game_1 except that the pair $(g_1 \star E_0, g_2 \star E_0, g_1 g_2 \star E_0, g_1 g_2^{-1} \star E_0)$ given to $\mathcal{A}$ is modified as $(g_1 \star E_0, g_2 \star E_0, g_1 g_2 \star E_0, g_4 \star E_0)$ where $g_4 \leftarrow G$. Claim $\mathsf{Game}_1 \approx_c \mathsf{Game}_2$ thanks to the sDDH problem. Concretely, we build an sDDH adversary $\mathcal{B}_1$ using $\mathcal{A}$. Upon receiving a square CSIDH challenge $(s \star E_0, X)$, the reduction $\mathcal{B}_1$ proceeds as follows

1. Generate $a \leftarrow G$.
2. Forward $(a \star (s \star E_0), (s \star E_0)^t, a \star E_0, a \star X)$ to $\mathcal{A}$.
3. Output whatever $\mathcal{A}$ returns.

Note that $(s \star E_0)^t = s^{-1} \star E_0$ and $a = (as)s^{-1}$. Therefore, when the challenge is the second case in the sDDH experiment (i.e. a random curve), $\mathcal{B}_1$ generates Game_2. On the other hand, if the challenge is the first case in the experiment (i.e. $X = s^2 \star E_0$), then $\mathcal{B}_1$ generates Game_1 since $a \star X = as^2 \star E_0$ and $as^2 = as(s^{-1})^{-1}$. Therefore, $\mathsf{Adv}^{\mathsf{sDDH}}(\mathcal{B}_1) = |\Pr[\mathcal{A}(\mathsf{Game}_1) \to 1] - \Pr[\mathcal{A}(\mathsf{Game}_2) \to 1]|$.

Game_3 : the same as Game_2 except that the pair $(g_1 \star E_0, g_2 \star E_0, g_1 g_2 \star E_0, g_4 \star E_0)$ given to $\mathcal{A}$ is modified as $(g_1 \star E_0, g_2 \star E_0, g_3 \star E_0, g_4 \star E_0)$ where $g_3 \leftarrow G$. This is exactly the second case in the rDDH problem. Claim $\mathsf{Game}_2 \approx_c \mathsf{Game}_3$ thanks to the DDH problem. Concretely, we build an DDH adversary $\mathcal{B}_2$ using $\mathcal{A}$. Upon receiving a square CSIDH challenge $(g_1 \star E_0, g_2 \star E_0, X)$, the reduction $\mathcal{B}_2$ proceeds as follows

1. Generate $g_4 \leftarrow G$.
2. Forward $(g_1 \star E_0, g_2 \star E_0, X, g_4 \star E_0)$ to $\mathcal{A}$.
3. Output whatever $\mathcal{A}$ returns.

Note that when the challenge is the second case in the DDH experiment (i.e. a random curve), $\mathcal{B}_2$ generates Game_3. On the other hand, if the challenge is the first case in the experiment (i.e. $X = g_1 g_2 \star E_0$), then $\mathcal{B}_1$ generates Game_2. Hence, $\mathsf{Adv}^{\mathsf{DDH}}(\mathcal{B}_2) = |\Pr[\mathcal{A}(\mathsf{Game}_2) \to 1] - \Pr[\mathcal{A}(\mathsf{Game}_3) \to 1]|$.

Therefore, we have

$$\mathsf{Adv}^{\mathsf{rDDH}}(\mathcal{A}) \leq \mathsf{Adv}^{\mathsf{sDDH}}(\mathcal{B}_1) + \mathsf{Adv}^{\mathsf{DDH}}(\mathcal{B}_2).$$

□

3 Hardness of Master Decisional Diffie-Hellman Problem

The following theorem shows that the MDDH problem is as hard as the DDH problem. It is worth highlighting the reduction is inspired by the pseudorandomness treatment in the literature [BMR10, ADMP20, BKW20, MOT20].

Theorem 3.1. *The* MDDH *problem is not easier than the* mcDDH *problem. Concretely, let* $(G, \mathcal{E}, \star, E_0)$ *be a group action,* $\mathcal{A}$ *be a* MDDH *problem adversary with parameter* $n \in \mathbb{N}$*. If at most* $q_{\mathsf{DH}} = poly(\lambda)$ *queries are made in the experiment by* MDDH $\mathcal{A}$ *then there exists* mcDDH *problem adversaries* $\mathcal{B}_2, \cdots \mathcal{B}_n$ *such that*

$$\mathsf{Adv}^{\mathsf{MDDH}}(\mathcal{A}) \leq \sum_{i=2}^{n} \mathsf{Adv}^{\mathsf{mcDDH}}(\mathcal{B}_i).$$

Proof. We prove the theorem via a hybrid argument by introducing a series of games $\mathsf{Game}_1, \cdots, \mathsf{Game}_n$ by modifying the responses of the DH oracle and the challenge oracle in the MDDH experiment gradually. Among the games, Game_1 is the original MDDH experiment, We will modify the response of the challenge oracle and the DH oracle together, which will be explained later. For $i \in [n]$ where $b \in \{0, 1\}$, let $\mathcal{A}(\mathsf{Game}_i(b))$ represent $\mathcal{A}$ running the Game_i, the modified MDDH experiment with the random coin b used in the experiment, and $\mathcal{A}$ will return 0 or 1. Therefore, by definition,

$$\mathsf{Adv}^{\mathsf{MDDH}}(\mathcal{A}) = |\Pr[\mathcal{A}(\mathsf{Game}_1(b = 1)) \to 1] - \Pr[\mathcal{A}(\mathsf{Game}_1(b = 0)) \to 1]|. \tag{1}$$

Looking ahead, Game_n is the modified MDDH experiment where both the DH oracle and the challenger reply with random elements in $\mathcal{E}$. Therefore, since b is information theoretically hidden from $\mathcal{A}$,

$$|\Pr[\mathcal{A}(\mathsf{Game}_n(b = 1)) \to 1] - \Pr[\mathcal{A}(\mathsf{Game}_n(b = 0)) \to 1]| = 0. \tag{2}$$

Game_1 : the original MDDH experiment starting with a tuple $(g_1 \star E_0, \cdots, g_n \star E_0)$ where $g_1, \cdots, g_n \leftarrow G$ and the oracle responds as specified.

Game_2 to Game_n: for $j \in \{2, \cdots, n\}$, Game_j is the same as Game_{j-1} except that the response of the DH oracle and the challenge oracle is modified as follows. The modification starts with a list L which is initially

$$\{(\mathbf{0}, E_0), (\mathbf{e}_1, g_1 \star E_0), \cdots, (\mathbf{e}_j, g_j \star E_0)\} \subseteq \{0, 1\}^j \times \mathcal{E}.$$

On the query $x = (x_1, \cdots, x_n) \in \{0, 1\}^n$, if $((x_1, \cdots, x_j), X) \in L$ for some $X \in \mathcal{E}$, the oracle returns $(\prod_{i=j+1}^{n} g_i^{x_i}) \star X$; otherwise, it draws $g' \leftarrow G$, computes $X = g' \star E_0$, adds $((x_1, \cdots, x_j), X)$ to the list L, and returns $(\prod_{i=j+1}^{n} g_i^{x_i}) \star X$ to $\mathcal{A}$. The reply for the challenge query is modified in the same way if the random coin $b = 0$.

Claim that $\mathsf{Game}_{j-1} \approx_c \mathsf{Game}_j$ for $\mathcal{A}$ for any $2 \leq j \leq n$. Concretely, a reduction $\mathcal{B}_j$ to the mcDDH problem proceeds as follows

1. Obtain $(g' \star E_0, \{(X_i, X_i')\}_{i \in [q_{\mathsf{DH}} + j - 1]})$ from the mcDDH oracle.
2. Then, $\mathcal{B}_j$ initializes with a list

$$L = \left\{ \begin{matrix} (\mathbf{e}_1, X_1), \cdots, (\mathbf{e}_{j-1}, X_{j-1}), (\mathbf{0}, E), \\ (\mathbf{e}_1 + \mathbf{e}_j, X_1'), \cdots, (\mathbf{e}_{j-1} + \mathbf{e}_j, X_{j-1}'), (\mathbf{e}_j, g' \star E_0) \end{matrix} \right\} \subset \{0, 1\}^j \times \mathcal{E},$$

   where $\mathbf{e}_i$ is the i-th elementary vector in $\{0, 1\}^j$, and set a counter $\mathsf{ct} = j$ to record the number of the pairs (X_i, X_i') taken into the list L.
3. Invoke $\mathcal{A}$ on input $(E, X_1, \cdots, X_{j-1}, g' \star E_0, g_{j+1} \star E_0, \cdots, g_n \star E_0)$ where $g_{j+1}, \cdots, g_n \leftarrow G$.
4. Upon receiving the oracle query $(x_1, \cdots, x_n) \in \{0, 1\}^n$, check whether $((x_1, \cdots, x_j), X) \in L$ for some $X \in \mathcal{E}$. If so, return $\prod_{i=j+1}^{n} g_i^{x_i} \star X$. Otherwise, update

$$L \leftarrow \{((x_1, \cdots, x_{j-1}, 0), X_{\mathsf{ct}}),\ ((x_1, \cdots, x_{j-1}, 1), X_{\mathsf{ct}}')\} \cup L,$$

   and set $\mathsf{ct} \leftarrow \mathsf{ct} + 1$, and rerun this step again.

5. Output whatever $\mathcal{A}$ returns.

Note that in Step 1. if $\mathcal{B}_j$ is in the experiment $\mathsf{Exp}^{\mathsf{mcDDH}}(0)$ in the mcDDH problem (Def. 2.21 Item 1) then $\mathcal{B}_j$ generates Game_{j-1}. In contrast, if it is in the experiment $\mathsf{Exp}^{\mathsf{mcDDH}}(1)$ in the mcDDH problem (Def. 2.21 Item 2), then $\mathcal{B}_j$ generates Game_j. It follows that for $b \in \{0,1\}$,

$$\begin{aligned}\mathsf{Adv}^{\mathsf{mcDDH}}(\mathcal{B}_j) =& |\Pr[\mathcal{B}_j(\mathsf{Exp}^{\mathsf{mcDDH}}(0)) \to 1] - \Pr[\mathcal{B}_j(\mathsf{Exp}^{\mathsf{mcDDH}}(1)) \to 1]| \\ =& |\Pr[\mathcal{A}(\mathsf{Game}_{j-1}(b)) \to 1] - \Pr[\mathcal{A}(\mathsf{Game}_j(b)) \to 1]|. \end{aligned} \tag{3}$$

Therefore, we have

$$\begin{aligned}\mathsf{Adv}^{\mathsf{MDDH}}(\mathcal{A}) &= |\Pr[\mathcal{A}(\mathsf{Game}_1(b=1)) \to 1] - \Pr[\mathcal{A}(\mathsf{Game}_1(b=0)) \to 1]| && \text{(By Eq. (1))} \\ &\leq \sum_{j=2}^{n} (|\Pr[\mathcal{A}(\mathsf{Game}_{j-1}(b=1)) \to 1] - \Pr[\mathcal{A}(\mathsf{Game}_j(b=0)) \to 1]| \\ &\quad + |\Pr[\mathcal{A}(\mathsf{Game}_{j-1}(b=1)) \to 1] - \Pr[\mathcal{A}(\mathsf{Game}_1(b=0)) \to 1]|) \\ &\quad + |\Pr[\mathcal{A}(\mathsf{Game}_n(b=1)) \to 1] - \Pr[\mathcal{A}(\mathsf{Game}_{j-1}(b=1)) \to 1]| && \text{(Union bounds.)} \\ &= \sum_{j=2}^{n-1} \mathsf{Adv}^{\mathsf{mcDDH}}(\mathcal{B}_j). && \text{(By Eqs. (2) and (3))}\end{aligned}$$

The result follows. □

4 Proof Systems

4.1 The Action Factorization Relation and Its Sigma-Protocol

We consider the following action factorization relation R_{fac} for our verifiable random functions.

$$R_{\mathsf{fac}} = \left\{ \mathsf{st} = (E_0, \{E_i\}_{i\in[N]}, E), \mathsf{wt} = \{s_i\}_{i\in[N]} \middle| \begin{matrix} E_i = s_i \star E_0 \ \forall\ i \in [N] \\ E = (\Pi_{i=1}^N s_i) \star E_0 \end{matrix} \right\}.$$

Sigma Protocol for R_{fac}. We give a basic sigma protocol for R_{fac} as described in Fig. 1. Let $N \in \mathbb{N}$ and a statement $(\mathsf{st} = E_0, \{E_i\}_{i\in[N]}, E)$. Say the prover has the witness $(\mathsf{wt} = \{s_i\}_{i\in[N]})$ such that $E_i = s_i \star E_0$ for any $i \in [N]$ and $E = (\Pi_{i=1}^N s_i) \star E_0$.

To prove the knowledge, the prover firstly generates $r_1, \cdots, r_N$, computes $E_i' = r_i \star E_i$ for all $i \in [N]$ and $E' = (\Pi_{i=1}^N r_i) \star E$, and sends those $N+1$ set elements to the verifier. The verifier returns a random challenge c from $\{0,1\}$ and sends it to the prover. If the challenge is 0, the prover reveals r_i for all $i \in [N]$ to the verifier. Otherwise, the prover reveals $s_i r_i$ for every $i \in [N]$. When $c = 0$, with received $\{r_i'\}_{i\in[N]}$ the verifier checks whether $r_i' \star E_i = E_i'$ for all $i \in [N]$ and whether $E' = (\Pi_{i=1}^N r_i) \star E$. When $c = 1$, with received $\{r_i'\}_{i\in[N]}$ the verifier checks whether $r_i' \star E_0 = E_i'$ for all $i \in [N]$ and also $(\Pi_{i=1}^N r_i') \star E_0 = E'$.

In each case, if all equalities hold, the verifier returns 1 to represent the acceptance. Otherwise, the verifier returns 0 to represent the rejection.

Remark 4.1. *Constructing the same proof system in a restricted EGA model or the original CSIDH setting [CLM+ 18] with an unknown structure group is feasible. In these settings, the group elements are represented as a linear combination of a given generating set where the coefficients are chosen from a small interval $[-t, t]$. In this case, revealing the addition $s + r$ if both $s, r \in [-t, t]$ will leak the information of the secret s. Therefore, using Fiat-Shamir with aborts [Lyu09, DG19] can circumvent this by sampling r from a larger*

```
round 1: $P_1'^{\mathcal{O}}(\mathsf{st} = (E_0, \{E_i\}_{i \in [N]}, E), \mathsf{wt} = \{s_i\}_{i \in [N]})$
    $\mathsf{seed}_0 \xleftarrow{\$} \{0,1\}^\lambda$
    $(r_1, \cdots, r_N) \leftarrow \mathcal{O}(\mathsf{PRNG} \,\|\, \mathsf{seed}_0)$        ▷ Generate $r_i \in G$
    $E' \leftarrow E$
    for $i$ from 1 to $N$ do
        $E_i' \leftarrow r_i \star E_i$
        $E' \leftarrow r_i \star E'$
    $\mathsf{root} \leftarrow \mathcal{O}(\mathsf{MT} \,\|\, E_1', \cdots, E_N', E')$        ▷ Produce $\mathsf{root} \in \{0,1\}^{2\lambda}$
    Prover sends $\mathsf{com} \leftarrow \mathsf{root}$ to Verifier.

round 2: $V_1'(\mathsf{com})$
    $c \xleftarrow{\$} \{0,1\}$
    Verifier sends $\mathsf{ch} \leftarrow c$ to Prover.

round 3: $P_2'(\mathsf{st}, \mathsf{com}, \mathsf{ch})$
    $c \leftarrow \mathsf{ch}$
    if $c = 1$ then
        for $i$ from 1 to $N$ do
            $r_i' \leftarrow s_i r_i$
        $\mathsf{resp} \leftarrow \{r_i'\}_{i \in [N]}$
    else
        $\mathsf{resp} \leftarrow \mathsf{seed}_0$
    Prover sends $\mathsf{resp}$ to Verifier

Verification: $V_2'^{\mathcal{O}}(\mathsf{com}, \mathsf{ch}, \mathsf{resp})$
    $(\mathsf{root}, c) \leftarrow (\mathsf{com}, \mathsf{ch})$
    if $c = 1$ then
        $(\{r_i'\}_{i \in [N]}) \leftarrow \mathsf{resp}$
        $\widetilde{E}' \leftarrow E_0$
        for $i$ from 1 to $N$ do
            $\widetilde{E}_i' \leftarrow r_i' \star E_0$
            $\widetilde{E}' \leftarrow r_i' \star \widetilde{E}'$
        $\widetilde{\mathsf{root}} \leftarrow \mathcal{O}(\mathsf{MT} \,\|\, \widetilde{E}_1', \cdots, \widetilde{E}_N', \widetilde{E}')$
        return $\top$ if $\widetilde{\mathsf{root}} = \mathsf{root}$; otherwise, return $\perp$.
    else
        Repeat round 1 with $\mathsf{seed}_0 \leftarrow \mathsf{resp}$.
        return $\top$ if results in $\mathsf{root}$; otherwise, return $\perp$.
```

Figure 1: Construction of the base sigma protocol $\Pi_\Sigma^{\mathsf{base}} = (P' = (P_1', P_2'), V' = (V_1', V_2'))$ for the relation R where $O(\mathsf{PRNG}\|\cdot)$ and $O(\mathsf{Com}\|\cdot)$ are a PRNG and a commitment scheme instantiated by the random oracle, respectively.

$[-(T+1)t, (T+1)t]$ *for some* $T \in \mathbb{N}$ *and, then, aborting the session while required to reveal* $r + s$ *and* $r + s \notin [-Tt, Tt]$*. With a straightforward application to our case of* $s_i r_i$ *and* $\Pi(s_i r_i)$ *for* $i \in [N]$ *and* $T = 2\lambda^2$*, the abort rate will be bounded above by* $1/3$ *(see [DG19, Lemma 2.]). The rejection sampling method can also be improved using [DPV19].*

To reduce the size of the overall response, the prover uses a pseudorandom number generator to generate $r_1, \cdots, r_N \in G$ with a seed, seed_0, picked uniformly at random from $\{0,1\}^\lambda$. Also, the prover uses the Merkle tree to reduce the communication cost of the first message by producing a root of $\{\{E_i'\}_{i \in [N]}, E'\}$ over $\{0,1\}^{2\lambda}$.

Theorem 4.2. *The sigma protocol* $\Pi_\Sigma^{\mathsf{base}}$ *described in Fig. 1 has correctness.*

Proof. When the challenge is $c = 0$, the prover sends the seed, seed_0, to the verifier. The computation of the verifier will result in the same Merkle root in this case.

When $c = 1$, the prover sends $r_i' = s_i r_i$ for every $i \in [N]$ to the verifier. Recall that for any $i \in [N]$, we have $E_i = s_i \star E_0$, $E_i' = r_i \star E_i$, $E = (\Pi_{i=1}^N s_i) \star E_0$, and $E' = (\Pi_{i=1}^N r_i) \star E$. Also, $E_i' = r_i \star E_i$. Hence, due to commutative G, we have

$$\begin{aligned}(E_1', \cdots, E_N', E') &= (r_1 s_1 \star E_0, \cdots, r_N s_N \star E_0, (\Pi_{i=1}^N r_i s_i) \star E_0) \\ &= (r_1' \star E_0, r_N' \star E_N, (\Pi_{i=1}^N r_i') \star E_0).\end{aligned}$$

The Merkle tree will result in the same root and correctness follows. □

Theorem 4.3. *Let* $|G| \geq 2^\lambda$ *(see Rem. 2.16). The sigma protocol* $\Pi_\Sigma^{\mathsf{base}}$ *described in Fig. 1 has 2-special soundness for the relation* R_{fac} *if the Merkle tree hash function* $\mathcal{O}(\mathsf{MT} \parallel \cdot)$ *is collision-resistant. Concretely, for a fixed statement* st*, there exists an extractor* Ext *on input two valid transcripts returning either a valid witness* wt *or a pair* $(\mathsf{wt}_1, \mathsf{wt}_2)$ *such that* $(\mathsf{st}, \mathsf{wt}) \in R_{\mathsf{fac}}$ *or* $\mathcal{O}(\mathsf{MT} \parallel \mathsf{wt}_1) = \mathcal{O}(\mathsf{MT} \parallel \mathsf{wt}_2)$*, respectively.*

Proof. Let $\{\mathsf{root}, 0, \mathsf{resp}_0\}$ and $\{\mathsf{root}, 1, \mathsf{resp}_1\}$ be the two valid transcripts for the same first-message root. Write $r_1, \cdots, r_N \leftarrow \mathcal{O}(\mathsf{PRNG} \parallel \mathsf{resp}_0)$ and $\{r'_1, \cdots, r'_N\} = \mathsf{resp}_1$, the extractor Ext proceeds as follows.

1. Compute $\mathsf{wt}_1 = (r_1 \star E_1, \cdots, r_N \star E_N, (\Pi_{i=1}^N r_i) \star E)$.
2. Compute $\mathsf{wt}_2 = (r'_1 \star E_0, \cdots, r'_N \star E_0, (\Pi_{i=1}^N r'_i) \star E_0)$.
3. If $\mathsf{wt}_1 \neq \mathsf{wt}_2$, then return $(\mathsf{wt}_1, \mathsf{wt}_2)$.
4. Else, return $(r_1^{-1} r'_1, \cdots, r_N^{-1} r'_N)$.

Since $V_2'^{\mathcal{O}}(\{\mathsf{root}, b, \mathsf{resp}_b\}) \to 1$ for $i \in \{0, 1\}$, we know have

$$\mathsf{root} = \mathcal{O}(\mathsf{MT} \parallel r_1 \star E_1, \cdots, r_N \star E_N, (\Pi_{i=1}^N r_i) \star E),$$
$$\mathsf{root} = \mathcal{O}(\mathsf{MT} \parallel r'_1 \star E_0, \cdots, r'_N \star E_0, (\Pi_{i=1}^N r'_i) \star E_0)$$

where $r_1, \cdots, r_N \leftarrow \mathcal{O}(\mathsf{PRNG} \parallel \mathsf{resp}_0)$ and $\{r'_1, \cdots, r'_N\} = \mathsf{resp}_1$. If $\mathsf{wt}_1 \neq \mathsf{wt}_2$, then they form a collision for the Merkle tree hash function.

If $\mathsf{wt}_1 = \mathsf{wt}_2$, we have $r_i \star E_1 = r'_i \star E_0$ for all $i \in [N]$ and $(\Pi_{i=1}^N r_i) \star E = (\Pi_{i=1}^N r'_i) \star E_0$. If follows that $E_i = (r_i^{-1} r'_i) \star E_0$ for all $i \in [N]$. Moreover, since the group is commutative and $(\Pi_{i=1}^N r_i)^{-1} (\Pi_{i=1}^N r'_i) \star E_0 = E$, we have $(\Pi_{i=1}^N (r_i^{-1} r'_i)) \star E_0 = E$. ☐

Theorem 4.4. *The sigma protocol* $\Pi_\Sigma^{\mathsf{base}}$ *described in Fig. 1 is statistically HVZK where the pseudorandom number generator and the Merkle tree hash function are modeled as random oracles* $\mathcal{O}(\mathsf{PRNG} \| \cdot)$ *and* $\mathcal{O}(\mathsf{MT} \| \cdot)$*, resp. Concretely, for any* $(\mathsf{st}, \mathsf{wt}) \in R_{\mathsf{fac}}$ *and a computationally-unbounded adversary* $\mathcal{A}$ *with at most* q_H *queries of* $\mathcal{O}(\mathsf{PRNG} \parallel \cdot)$*, there exists a simulator* Sim *such that*

$$\left| \Pr[\mathcal{A}^{\mathcal{O}}(P^{\mathcal{O}}(\mathsf{st}, \mathsf{wt}, c)) = 1] - \Pr[\mathcal{A}^{\mathcal{O}}(\mathsf{Sim}^{\mathcal{O}}(\mathsf{st}, c)) = 1] \right| \leq q_H / 2^\lambda.$$

Proof. Let $(\mathsf{st} = (E_0, \{E_i\}_{i \in [N]}, E), \mathsf{wt} = \{s_i\}_{i \in [N]}) \in R_{\mathsf{fac}}$. Given a st and $c \in \{0, 1\}$, the simulator $\mathsf{Sim}^{\mathcal{O}}(\mathsf{st}, \mathsf{wt}, c)$ proceeds as follows.

1. If $c = 0$, then execute P'_1 and generate $(\mathsf{root}, 0, \mathsf{seed}_0)$ where the witness is not required in this process.
2. If $c = 1$, then
   - (1.) Generate $r'_1, \cdots, r'_N \leftarrow G$ and let $\mathsf{resp} \leftarrow \{r'_1, \cdots, r'_N\}$.
   - (2.) Compute $E'_i = r'_i \star E_0$ for every $i \in [N]$.
   - (3.) Compute $E' = (\Pi_{i=1}^N r'_i) \star E_0$.
   - (4.) Compute $\mathsf{root} \leftarrow \mathcal{O}(\mathsf{MT} \parallel E'_1, \cdots, E'_N, E')$.
   - (5.) Return $(\mathsf{root}, c, \mathsf{resp})$.

The simulated transcripts are identical to ones produced by the prover with the witness executing the protocol $\Pi_\Sigma^{\mathsf{base}}$. For the case $c = 0$, the procedure is the same since the witness is not involved.

For the case $c = 1$, one can observe that the simulator returns a valid transcript and each element in the response follows the uniform distribution over G. The distribution is the same as the uniform distribution over the coset $(s_i)^{-1}G$ for any $i \in [N]$ used by the prover, since $\mathcal{O}(\mathsf{PRNG} \,\|\, \cdot)$ is modeled as a random oracle, except for those queries has been made before. Concretely, the difference of two distribution is

$$\begin{aligned} &1/2\sum \left|\Pr[(\mathsf{com},\mathsf{ch},\mathsf{resp}) \leftarrow \widetilde{P}^{\mathcal{O}}(\mathsf{st},\mathsf{wt},c))] - \Pr[(\mathsf{com},\mathsf{ch},\mathsf{resp}) \leftarrow \mathsf{Sim}^{\mathcal{O}}(\mathsf{st},c)]\right| \\ =&1/2\sum \left|\Pr[(\mathsf{com},1,\mathsf{resp}) \leftarrow \widetilde{P}^{\mathcal{O}}(\mathsf{st},\mathsf{wt},c))] - \Pr[(\mathsf{com},1,\mathsf{resp}) \leftarrow \mathsf{Sim}^{\mathcal{O}}(\mathsf{st},c)]\right| \\ =&\frac{q_H}{2}(1/2^{\lambda} - 1/|G|^N) \\ \leq&\frac{q_H}{2^{\lambda}}, \end{aligned}$$

so is the advantage of the adversary $\mathcal{A}$. ☐

Theorem 4.5. *Let $|G| \geq 2^{\lambda}$ (see Rem. 2.16). The sigma protocol $\Pi_{\Sigma}^{\mathsf{base}}$ in Fig. 1 has λ min-entropy where $\mathcal{O}(\mathsf{PRNG} \,\|\, \cdot)$ and $\mathcal{O}(\mathsf{MT} \,\|\, \cdot)$ are modeled by a random oracle.*

Proof. When the challenge $\mathsf{ch} = 0$, the seed is drawn uniformly at random from $\{0,1\}^{\lambda}$, and then r_i are drawn uniformly at random from G for any $i \in [N]$. Note that $|G| \geq 2^{\lambda}$. Since the action is free and transitive, $r_i \star E_i$ follows the uniform distribution over $\mathcal{E}$ for every i. Then, $\mathsf{com} \in \{0,1\}^{2\lambda}$ is produced by $\mathcal{O}(\mathsf{MT} \,\|\, \cdot)$. Throughout the procedure, every random element is drawn from a set larger than 2^{λ}. Therefore, we have $\Pr\left[\mathsf{com} = \mathsf{com}' \middle| \mathsf{com} \leftarrow P_1^{\mathcal{O}}(\mathsf{st},\mathsf{wt}), \mathsf{com}' \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{st},\mathsf{wt})\right] \leq 2^{-\lambda}$. ☐

4.2 Online-extractable NIZK

By λ times repetitions and using the Fiat-Shamir transform, we turn the sigma protocol Fig. 1 into a proof system for the relation R_{fac}. The description is displayed in Fig. 2.

```
$\mathsf{Prove}^{\mathcal{O}}(\mathsf{st} = (E_0, \{E_i\}_{i \in [N]}, E), \mathsf{wt} = \{s_i\}_{i \in [N]})$
for $i \in [\lambda]$ do
    $\mathsf{com}_i \leftarrow P_1'^{\mathcal{O}}(\mathsf{st},\mathsf{wt})$
$\mathsf{com} \leftarrow (\mathsf{com}_1, \cdots, \mathsf{com}_{\lambda})$
$\mathsf{ch} = (c_1, \cdots, c_{\lambda}) \leftarrow \mathcal{O}(\mathsf{FS} \,\|\, \mathsf{st} \,\|\, \mathsf{com})$
for $i \in [\lambda]$ do
    $\mathsf{resp}_i \leftarrow P_2'^{\mathcal{O}}(\mathsf{st}, \mathsf{com}_i, c_i)$
$\mathsf{resp} \leftarrow (\mathsf{resp}_1, \cdots, \mathsf{resp}_{\lambda})$
return $\pi \leftarrow (\mathsf{com}, \mathsf{ch}, \mathsf{resp})$
```

```
$\mathsf{Verify}^{\mathcal{O}}(\mathsf{st} = (E_0, \{E_i\}_{i \in [N]}, E), \pi)$
$(\mathsf{com} = (\mathsf{com}_1, \cdots, \mathsf{com}_{\lambda}), \mathsf{ch} = (c_1, \cdots, c_{\lambda}), \mathsf{resp} = (\mathsf{resp}_1, \cdots, \mathsf{resp}_{\lambda})) \leftarrow \pi$
$\mathsf{output} = 1$
for $i \in [\lambda]$ do
    $r \leftarrow V_2'(\mathsf{com}_i, c_i, \mathsf{resp}_i)$
    $\mathsf{output} \leftarrow \mathsf{output} \cdot r$
$\mathsf{output} \leftarrow \mathsf{output} \cdot (\mathsf{ch} == \mathcal{O}(\mathsf{FS} \,\|\, \mathsf{st} \,\|\, \mathsf{com}))$
return $\mathsf{output}$
```

Figure 2: NIZK for the relation $\mathsf{R_{fac}}$ by applying the Fiat-Shamir transform to $\Pi_{\Sigma}^{\mathsf{base}} = (P' = (P_1', P_2'), V' = (V_1', V_2'))$ with λ repetitions.

Theorem 4.6 (Completeness)**.** *The proof system Π_{NIZK} for the relation R_{fac} in Fig. 2 is complete.*

Proof. In each iteration of $i \in [\lambda]$ in Fig. 2, the prover and the verifier execute P' and V' in $\Pi_{\Sigma}^{\mathsf{base}} = (P', V')$ respectively. By Def. 2.2, each execution of $\Pi_{\Sigma}^{\mathsf{base}} = (P', V')$ has correctness, and the completeness of Π_{NIZK} follows. ☐

Theorem 4.7 (Zero-knowledge)**.** *Let $|G| \geq 2^{\lambda}$ (see Rem. 2.16). The proof system Π_{NIZK} for the relation R_{fac} in Fig. 2 is zero-knowledge in the random oracle model. Concretely, for any $(\mathsf{st},\mathsf{wt}) \in R_{\mathsf{fac}}$ and a computationally-unbounded adversary $\mathcal{A}$ with at most q_{PRNG} queries of $\mathcal{O}(\mathsf{PRNG} \,\|\, \cdot)$ and q_{FS} queries of $\mathcal{O}(\mathsf{FS} \,\|\, \cdot)$, there exists a simulator Sim such that*

$$\left|\Pr\left[\mathcal{A}^{\mathcal{O}}(P^{\mathcal{O}}(\mathsf{st},\mathsf{wt})) = 1\right] - \Pr\left[\mathcal{A}^{\mathcal{O}}(\mathsf{Sim}^{\mathcal{O}}(\mathsf{st})) = 1\right]\right| \leq \frac{q_{\mathsf{PRNG}}}{2^{\lambda}} + \frac{q_{\mathsf{FS}}}{2^{\lambda N}},$$

Proof. Let Sim' be the simulator in Thm. 4.4. The simulator Sim firstly simulates the oracle of $\mathcal{O}(\mathsf{FS} \parallel \cdot)$, $\mathcal{O}(\mathsf{FS} \parallel \mathsf{MT})$ and $\mathcal{O}(\mathsf{PRNG} \parallel \cdot)$ by keeping lists $L_{\mathsf{FS}}, L_{\mathsf{MT}}$, and L_{PRNG} respectively using the straight-line and on-the-fly method. Sim also keeps a list L to simulate the oracle queries. Take $\mathcal{O}(\mathsf{FS} \parallel \cdot)$ for instance, upon receiving an oracle query as $\mathcal{O}(\mathsf{FS} \parallel x)$, the Sim simulates the oracle as follows.

1. Check whether there exists a pair $(x, y) \in L_{\mathsf{FS}}$ for some y. If so, return y.
2. Otherwise draw $y \leftarrow \{0,1\}^\lambda$ uniformly at random. Add y to the list (x, y) and return y.

Given a statement st in the language of R_{fac}, the simulator Sim simulates the transcripts as follows.

1. Generate $\mathsf{ch} = (c_1, \cdots, c_\lambda) \leftarrow \{0,1\}^\lambda$ uniformly at random.
2. For each $i \in [\lambda]$, run $(\mathsf{com}_i, c_i, \mathsf{resp}_i) \leftarrow \mathsf{Sim}'(\mathsf{st}, c_i)$.
3. Concatenate $\mathsf{com} \leftarrow (\mathsf{com}_1, \cdots, \mathsf{com}_\lambda)$, $\mathsf{resp} \leftarrow (\mathsf{resp}_1, \cdots, \mathsf{resp}_\lambda)$.
4. Add $(\mathsf{com}, \mathsf{ch})$ to the list L_{FS}. If com has been queried before, abort and return $\perp$.
5. Output the transcript $(\mathsf{com}, \mathsf{ch}, \mathsf{resp})$.

By Thm. 4.5, we know each generation com_i has λ min-entropy. Therefore, the abort in Item 4 occurs with a negligible probability $q_{\mathsf{FS}}/2^{\lambda N}$.

Given such a distinguisher $\mathcal{A}$, one can construct an HVZK adversary $\mathcal{B}$ against the sigma-protocol $\Pi_\Sigma^{\mathsf{base}}$ using $\mathcal{A}$. Recall that when the challenge is 0, the simulation of $\mathsf{Sim}'(\cdot, 0)$ is perfect. The reduction $\mathcal{B}$ using $\mathcal{A}$ proceeds as follows. Upon receiving the statement st and the transcript ensemble $X = \{\mathsf{com}_i, 1, \mathsf{resp}_i\}_i$ for the challenge 1, $\mathcal{B}$ simulates as what Sim does except that the transcripts from $\mathsf{Sim}'(\mathsf{st}, 1)$ is replace by those taken from the ensemble X. $\mathcal{B}$ invokes $\mathcal{A}$ with st and the simulated transcripts. When the ensemble is generated by a real prover, then $\mathcal{B}$ generates the transcripts as a real prover in Π_{NIZK} except for the occurrence of aborts. When the ensemble is generated by a simulator, then $\mathcal{B}$ generates the transcripts as Sim in Π_{NIZK}. Hence, $\mathsf{Adv}^{\mathsf{ZK}}_{\Pi_{\mathsf{NIZK}}}(\mathcal{A}) \leq \mathsf{Adv}^{\mathsf{HVZK}}_{\Pi_\Sigma^{\mathsf{base}}}(\mathcal{B}) + q_{\mathsf{FS}}/2^{\lambda N}$.

Therefore, we have

$$\left| \Pr\left[\mathcal{A}^{\mathcal{O}}(P^{\mathcal{O}}(\mathsf{st}, \mathsf{wt})) = 1\right] - \Pr\left[\mathcal{A}^{\mathcal{O}}(\mathsf{Sim}^{\mathcal{O}}(\mathsf{st})) = 1\right] \right| \leq \frac{q_{\mathsf{PRNG}}}{2^\lambda} + \frac{q_{\mathsf{FS}}}{2^{\lambda N}}.$$

□

Theorem 4.8 (Online-extractable)**.** *Assume $\mathcal{O}(\cdot)$ is collision resistant, $|G| \geq 2^\lambda$ (see Rem. 2.16), and $N \in \mathbb{N}$. The proof system Π_{NIZK} in Fig. 2 is online-extractable. Concretely, for any adversary $\mathcal{A}$ with q_{FS} queries to $\mathcal{O}(\mathsf{FS} \parallel \cdot)$ and q_{PRNG} queries to $\mathcal{O}(\mathsf{PRNG} \parallel \cdot)$,*

$$\mathsf{Adv}^{\mathsf{OE}}_{\Pi_{\mathsf{VRF}^\star}}(\mathcal{A}) \leq \frac{q_{\mathsf{FS}} + 1}{2^\lambda} + \frac{q_{\mathsf{FS}} q_{\mathsf{PRNG}}}{2^{N\lambda}}.$$

Proof. With the extractability access to the oracle, the extractor Ext observes the queries to $\mathcal{O}$ of the form $(\mathsf{PRNG} \parallel \cdot)$, and record (x, y) to the list L_{PRNG} where x is the input and y is the oracle output. Also, Ext does the same for the queries of the form $(\mathsf{FS} \parallel \cdot)$, and keeps a list L_{FS}. We say x is in the list L_{PRNG} if there exists some y such that $(x, y) \in L_{\mathsf{PRNG}}$.

Upon receiving a statement $\mathsf{st} = (E_0, (E_1, \cdots, E_N), E')$, possibly not in the language of $\mathsf{R}_{\mathsf{fac}}$, and a valid proof $(\mathsf{com}, \mathsf{ch}, \mathsf{resp})$, the extractor Ext proceeds as follows.

1. Parse $\mathsf{ch} = (c_1, \cdots, c_\lambda)$ where $c_k \in \{0,1\}$ for $i \in [\lambda]$. Also, parse $\mathsf{com} = (\mathsf{root}_1, \cdots, \mathsf{root}_\lambda)$ and $\mathsf{resp} = (\mathsf{resp}_1, \cdots, \mathsf{resp}_\lambda)$.
2. Collect $K \subseteq [\lambda]$ where $c_k = 1$ for any $k \in K$.

3. Collect the queries $S = \{\mathsf{seed}_j\}_{j \in [q_{\mathsf{PRNG}}]}$ recorded in list L_{PRNG}.
4. Find one $(k, j) \in K \times [q_{\mathsf{PRNG}}]$ such that $\mathsf{root}_k, \mathsf{seed}_j$ satisfy $\mathsf{root}_k = \mathcal{O}(\mathsf{MT} \,\|\, (r_1, \cdots, r_N) \star (E_1, \cdots, E_N))$ where $(r_1, \cdots, r_N) \leftarrow \mathcal{O}(\mathsf{PRNG} \,\|\, \mathsf{seed}_j)$. If no such pairs found, return $\perp$.
5. Execute the extractor Ext' described in Thm. 4.3 on input two valid transcripts $(\mathsf{com}_k, 0, \mathsf{seed}_j)$, $(\mathsf{com}_k, 1, \mathsf{resp}_k)$ to extract $\mathsf{wt} \in G^N$ and return wt.

We have to argue the pair (k, j) in Item 4 exists with an overwhelming probability.
For simplicity, we say a seed seed can *serve as a 0-response* for root if $(r_1, \cdots, r_N) \leftarrow \mathcal{O}(\mathsf{PRNG} \,\|\, \mathsf{seed})$ and $\mathsf{root} = \mathcal{O}(\mathsf{MT} \,\|\, (r_1, \cdots, r_N) \star \mathsf{st}, (\Pi r_i) \star E_0)$. For example, one can interpret Item 4 as finding a 0-response for root_k for some $k \in K$.

Case I: $\mathcal{O}(\mathsf{FS} \,\|\, \mathsf{st} \,\|\, \mathsf{com})$ has not been queried before the verification. This implies that $\mathcal{A}$ produces com and resp without knowing the challenge. However, it requires ch equals $\mathcal{O}(\mathsf{FS} \| \mathsf{st} \| \mathsf{com})$ in the verification process. This occurs with a probability not greater than $1/2^\lambda$.

Analysis. We analyze the advantage of $\mathcal{A}$ against Ext by aiming at each FS challenge query made by the adversary to $\mathcal{O}(\mathsf{FS} \,\|\, \mathsf{st}' \,\|\, \cdot)$ for some st'. We analyze when $\mathcal{A}$ submits a new $\mathsf{com}' = (\mathsf{root}'_1, \cdots, \mathsf{root}'_\lambda)$ to the FS oracle, whether there exist 0-responses in the query list L_{PRNG}.

For $K' \subseteq [\lambda]$, we define the $\mathsf{E}_{K'}$ that when $\mathcal{A}$ submitting com to the FS oracle of the form $(\mathsf{FS} \,\|\, \mathsf{st}' \,\|\, \mathsf{root}_1, \cdots, \mathsf{root}_\lambda)$ to the random oracle, there exist no 0-responses in the query list L_{PRNG} for root_k for any $k \in [K']$. We also define event $\mathsf{F}_{K'}$ that the FS oracle returns the challenge $(c'_1, \cdots, c'_\lambda)$ where $c'_k = 1$ for all $k \in K'$ and $c_k = 0$ otherwise. Obviously, $\Pr[\mathsf{F}_{K'}] = 1/2^\lambda$ for every new FS query. Denote the event that $\mathcal{A}$ outputs a transcript containing com' by $\mathsf{O}_{\mathsf{com}'}$ (e.g. $(\mathsf{com}', \mathsf{ch}', \mathsf{resp}')$) and the output is extractable for Ext by $\mathsf{L}_{\mathsf{com}'}$. The latter case implies $\mathcal{A}$ fails.

Note that $\mathsf{E}_{K'}$ forms a partition. Therefore, if $\mathcal{A}$ returns $(\mathsf{com}', \mathsf{ch}', \mathsf{resp}')$ we have

$$\begin{aligned}
\Pr[\mathsf{O}_{\mathsf{com}'}] &= \sum_{K' \subseteq [\lambda]} \Pr[\mathsf{O}_{\mathsf{com}'} \cap \mathsf{E}_{K'}] \\
&= \Pr[\mathsf{O}_{\mathsf{com}'} \cap \mathsf{E}_{K'}], \text{ for some } K' \\
&= \Pr[\mathsf{O}_{\mathsf{com}'} \cap \mathsf{E}_{K'} \cap \mathsf{F}_{K'}] + \Pr[\mathsf{O}_{\mathsf{com}'} \cap \mathsf{E}_{K'} \cap \neg \mathsf{F}_{K'}] \\
&\leq 1/2^\lambda + \Pr[\mathcal{A} \text{ wins using } \mathsf{com}' \cap \mathsf{E}_{K'} \cap \neg \mathsf{F}_{K'}] + \Pr[\mathsf{L}_{\mathsf{com}'} \cap \mathsf{E}_{K'} \cap \neg \mathsf{F}_{K'}],
\end{aligned}$$

where $\Pr[\mathsf{O}_{\mathsf{com}'} \cap \mathsf{E}_{K'} \cap \mathsf{F}_{K'}] \leq 1/2^\lambda$ since $\Pr[\mathsf{F}_{K'}] = 1/2^\lambda$. We partition the event that $\mathsf{O}_{\mathsf{com}'} \cap \mathsf{E}_{K'} \cap \neg \mathsf{F}_{K'}$ into two cases: $\mathcal{A}$ wins or not (i.e. whether the tuple $(\mathsf{com}', \mathsf{ch}', \mathsf{resp}')$ is extractable).

Case II: $\mathcal{A}$ wins with a tuple using $\mathsf{com}' \cap \mathsf{E}_{K'} \cap \neg \mathsf{F}_{K'}$. Recall that if there exists $k \in [\lambda] - K'$ such that $c'_k = 1$, then one can invoke Ext to extract the witness using resp'_k and the list of $\mathcal{O}(\mathsf{PRNG} \,\|\, \cdot)$. Therefore, the case that $\mathcal{A}$ wins implies that $c'_k = 0$ for all $k \in [\lambda] - K'$ and $\mathcal{A}$ produces a seed seed_k for some $c'_k = 0, k \in K'$ such that $\mathsf{com}'_k = (r_1, \cdots, r_N) \star E_0$ where $(r_1, \cdots, r_N) \leftarrow \mathcal{O}(\mathsf{PRNG} \,\|\, \mathsf{seed}_k)$. Note that such seed_k is generated after the FS query. Since the protocol has the unique response property [3] and the group elements are generated uniformly from G by $\mathcal{O}(\mathsf{PRNG} \| \cdot)$, the adversary can generate such a seed with chance not greater than $q_{\mathsf{PRNG}}/|G|^N$.

Therefore,

$$|\Pr[\mathsf{O}_{\mathsf{com}'}] - \Pr[\mathsf{L}_{\mathsf{com}'}]| \leq 1/2^\lambda + q_{\mathsf{PRNG}}/|G|^N.$$

Wrapping up, given an adversary with q_{FS} FS queries and q_{PRNG} PRNG queries, by taking a union bound

---

[3] Given $\mathbf{E} \in \mathcal{E}^N$ there exist two unique group elements $\mathbf{g} \in G^N$ and $\mathbf{g} \in G'^N$ such that $\mathbf{E} = \mathbf{g} \star (E_1, \cdots, E_N)$ and $\mathbf{E} = \mathbf{g}' \star E_0$.

over all FS queries we know the advantage of the adversary:

$$\begin{aligned}\mathsf{Adv}^{\mathsf{OnlineExtract}}_{\Pi_{\mathsf{VRF}^\star}}(\mathcal{A}) &\leq \Pr[\text{Case I}] + \sum_{\mathsf{com}\text{ in } L_{\mathsf{FS}}} \Pr[\text{Case II wrt } \mathsf{com}] \\ &\leq \frac{1}{2^\lambda} + \sum_{\mathsf{com}\text{ in } L_{\mathsf{FS}}} |\Pr[\mathsf{O}_{\mathsf{com}'}] - \Pr[\mathsf{L}_{\mathsf{com}'}]| \\ &\leq \frac{q_{\mathsf{FS}}+1}{2^\lambda} + \frac{q_{\mathsf{FS}} q_{\mathsf{PRNG}}}{|G|^N}.\end{aligned}$$

□

5 Verifiable Random Functions from Effective Group Actions

In this section, we present our first VRF construction from an effective group action – CAPYBARA (Compact Action factorization Proofs Yielded By A RAndom function):

Construction. $\Pi_{\mathsf{VRF}} = \{\mathsf{ParGen}, \mathsf{KeyGen}, \mathsf{VRFEval}, \mathsf{Ver}\}$ using $\Pi^{\mathsf{fac}}_{\mathsf{NIZK}} = (P, V)$, H where:

- $\mathsf{ParGen}(1^\lambda)$: on input a security parameter 1^λ, it returns $\mathsf{pp} = (G, \mathcal{E}, \star, E_0)$, which is a free, transitive and effective group action.
- $\mathsf{KeyGen}(\mathsf{pp})$: On input public parameter $\mathsf{pp} = (G, \star, E_0, \mathcal{E})$, it returns a secret key $\mathsf{sk} = (c_0, c_1, s_1, \cdots, s_\lambda)$ and a public key $\mathsf{vk} = (c_0 \star E_0, c_1 \star E_0, s_1 \star E_0, \cdots, s_\lambda \star E_0)$.
- $\mathsf{VRFEval}(\mathsf{sk}, x)$ [4]: On input a secret key sk and an input $x = (x_i) \in \{0,1\}^\lambda$, this algorithm outputs (v, π) for the VRF value where $v = (c_0 c_1 \Pi_{i=1}^{\lambda} s_i^{x_i}) \star E_0$ together with the corresponding proof π where $I = \{1, 2\} \cup \{i + 2 | x_i = 1 \wedge i \in [\lambda]\}$ and $\pi \leftarrow P(\mathsf{st} = (E_0, \mathsf{vk}_I, v), \mathsf{wt} = \mathsf{sk}_I)$ of Π_{NIZK}.
- $\mathsf{Ver}(\mathsf{vk}, v, x, \pi)$: On input (vk, v, x, π), this algorithm computes $b \leftarrow V(\mathsf{st} = (E_0, \mathsf{vk}_I, v), \pi)$ using Π_{NIZK} where $I = \{1, 2\} \cup \{i + 2 | x_i = 1 \wedge i \in [\lambda]\}$, and returns b.

Theorem 5.1. *The VRF construction Π_{VRF} in Fig. 3 has provability.*

Proof. Let $(E, \pi) \leftarrow \mathsf{VRFEval}(\mathsf{sk}, x)$ and $v = (c_0 c_1 \Pi_{i=1}^{\lambda} s_i^{x_i}) \star E_0$. The proof π is generated by $P(\mathsf{st} = (E_0, \mathsf{vk}_I, v), \mathsf{wt} = \mathsf{sk}_I)$ and $I = \{1, 2\} \cup \{i + 2 | x_i = 1 \wedge i \in [\lambda]\}$. Since $(\mathsf{st} = (E_0, \mathsf{vk}_I, v), \mathsf{wt} = \mathsf{sk}_I) \in R$ and Π_{NIZK} has correctness, we have $\mathsf{VRFVer}(\mathsf{vk}, v, x, \pi) = 1$. □

Theorem 5.2. *If Π_{NIZK} is extractable, the VRF construction Π_{VRF} in Fig. 3 has computational full uniqueness in the random oracle model. Concretely, for any full uniqueness adversary $\mathcal{A}$ against Π_{VRF}, there exists an extractable adversary $\mathcal{B}$ against Π_{NIZK} such that*

$$\mathsf{Adv}^{\mathsf{UP}}_{\Pi_{\mathsf{VRF}}}(\mathcal{A}) \leq 2\mathsf{Adv}^{\mathsf{OE}}_{\Pi_{\mathsf{NIZK}}}(\mathcal{B}).$$

Proof. Given $(\mathsf{vk}, x, v_1, v_2, \pi_1, \pi_2) \leftarrow \mathcal{A}$ where both

$$\mathsf{VRFVer}((E_0, \mathsf{vk}_I, v_1), \pi_1) = \mathsf{VRFVer}((E_0, \mathsf{vk}_I, v_2), \pi_2) = 1$$

, $v_1 \neq v_2$, and $I = \{1, 2\} \cup \{i + 2 | x_i = 1 \wedge i \in [\lambda]\}$.

By invoking the extractor Ext of Π_{NIZK} in Thm. 4.8 twice, we have $\mathbf{s}_1 \leftarrow \mathsf{Ext}((E_0, \mathsf{vk}_I, v_1), \pi_1)$, $\mathbf{s}_2 \leftarrow \mathsf{Ext}((E_0, \mathsf{vk}_I, v_2), \pi_2)$ such that $v_1 = (\Pi_i(\mathbf{s}_1)_i) \star E_0$ and $v_2 = (\Pi_i(\mathbf{s}_2)_i) \star E_0$. Also, $\mathsf{vk}_I = \mathbf{s}_1 \star E_0$ and $\mathsf{vk}_I = \mathbf{s}_2 \star E_0$. Since the action is free and transitive, we have $\mathbf{s}_1 = \mathbf{s}_2$, which contradicts $v_1 \neq v_2$.

---

[4] In the formal syntax of VRF, vk is not inclueded in the $\mathsf{VRFEval}$. One can also include vk as part of the secret key. In our case, the user can recover vk from sk. Both justify the notation here.

```
$\mathsf{ParGen}(1^\lambda)$
Generate $\mathsf{pp} = (G, \star, E_0, \mathcal{E})$
return $\mathsf{pp}$
```

```
$\mathsf{KeyGen}(\mathsf{pp})$
$(G, \star, E_0, \mathcal{E}) \leftarrow \mathsf{pp}$
$\mathsf{sk} \leftarrow G^{\lambda+2}$
$\mathsf{vk} = \mathsf{sk} \star E_0$
return $(\mathsf{vk}, \mathsf{sk})$
```

```
$\mathsf{VRFEval}(\mathsf{sk}, x)$
$(G, \star, E_0, \mathcal{E}) \leftarrow \mathsf{pp}$
$v = E_0$
$I \leftarrow \{1, 2\}$
for $i \in [\lambda]$ do
    if $x_i = 1$ then
        $I \leftarrow I \cup \{i + 2\}$
for $s \in \mathsf{sk}_I$ do
    $v \leftarrow s \star v$
$\pi \leftarrow P(\mathsf{st} = (E_0, \mathsf{vk}_I, v), \mathsf{wt} = \mathsf{sk}_I)$
return $(v, \pi)$
```

```
$\mathsf{VRFVer}(\mathsf{vk}, v, x, \pi)$
$(G, \star, E_0, \mathcal{E}) \leftarrow \mathsf{pp}$
for $i \in [\lambda]$ do
    if $x_i = 1$ then
        $I \leftarrow I \cup \{i + 2\}$
return $V(\mathsf{st} = (E_0, \mathsf{vk}_I, v), \pi)$
```

Figure 3: The verifiable random function scheme Π_{VRF} based an effective group action and on the DDH problem where $\Pi^{\mathsf{fac}}_{\mathsf{NIZK}} = (P, V)$ is an NIZK for the relation R_{fac} described in Sec. 4.2.

In other words, if $\mathcal{A}$ wins, then the extractor $\mathcal{E}$ shall fail among two extractions. We can therefore tranform $\mathcal{A}$ into an extractabililty adversary $\mathcal{B}$ against Π_{NIZK}. Concretely, if $\mathcal{A}$ returns $(\mathsf{vk}, x, v_1, v_2, \pi_1, \pi_2)$, then $\mathcal{B}$ randomly outputs one of $((E_0, \mathsf{vk}_I, v_1), \pi_1)$ or $((E_0, \mathsf{vk}_I, v_2), \pi_2)$ where $I = \{1, 2\} \cup \{i + 2 | x_i = 1 \wedge i \in [\lambda]\}$. Therefore, we have

$$\mathsf{Adv}^{\mathsf{UP}}_{\Pi_{\mathsf{VRF}}}(\mathcal{A}) \leq 2\mathsf{Adv}^{\mathsf{OE}}_{\Pi_{\mathsf{NIZK}}}(\mathcal{B}).$$

□

Theorem 5.3. *If the decisional master DDH problem is hard, then the VRF construction Π_{VRF} in Fig. 3, with a subroutine $\Pi_{\mathsf{NIZK}} = (P, V)$ in Fig. 2, has (residual) pseudorandomness. Concretely, for any residual pseudorandomness adversary $\mathcal{A}$ against Π_{VRF} with at most q_{PRNG} queries of $\mathcal{O}(\mathsf{PRNG} \,\|\, \cdot)$ and q_{FS} queries of $\mathcal{O}(\mathsf{FS} \,\|\, \cdot)$, there exists a MDDH adversary $\mathcal{B}$ such that*

$$\mathsf{Adv}^{\mathsf{PR}}_{\Pi_{\mathsf{VRF}}}(\mathcal{A}) \leq \frac{q_{\mathsf{PRNG}}}{2^\lambda} + \frac{q_{\mathsf{FS}}}{2^{2\lambda}} + \mathsf{Adv}^{\mathsf{MDDH}}(\mathcal{B}).$$

Proof. We show by using a hybrid argument that such an adversary $\mathcal{A}$ can be transformed into a MDDH adversary $\mathcal{B}_2$. Let Game_0 be the original residual pseudorandomness experiment and Game_1 be the modified experiment. For $i \in \{0, 1\}$, we denote the advantage of $\mathcal{A}$ in Game_i by $\mathsf{Adv}_i(\mathcal{A}) = |\Pr[\mathcal{A}(\mathsf{Game}_i(b = 1)) \rightarrow 1] - \Pr[\mathcal{A}(\mathsf{Game}_i(b = 0)) \rightarrow 1]|$, where $b \in \{0, 1\}$ represents the random coin chosen by the challenger (Def. 2.10 Item 7). Since Game_0 is the original experiment, we know $\mathsf{Adv}_0(\mathcal{A}) = \mathsf{Adv}^{\mathsf{PR}}_{\Pi_{\mathsf{VRF}}}(\mathcal{A})$ by definition.

We introduce Game_1 which is the same as Game_0 except for the way of evaluating x for a query. Rather than generated via Prove from the subroutine Π_{NIZK}, the proof is generated using the simulator Sim for Π_{NIZK} in Thm. 4.7. Here, we have the parameter $N \geq 2$ in Thm. 4.7 since the prover uses at least two elements ($\mathsf{sk}_{\{1,2\}} \subseteq \mathsf{sk}_I$) to generate a proof. By Thm. 4.7, since the simulator Sim is statistically indistinguishable from a real prover, the change in Game_1 results in a negligible loss. Concretely, $|\mathsf{Adv}_0(\mathcal{A}) - \mathsf{Adv}_1(\mathcal{A})| \leq \frac{q_{\mathsf{PRNG}}}{2^\lambda} + \frac{q_{\mathsf{FS}}}{2^{2\lambda}}$.

We now transform an adversary in Game_1 into a MDDH problem adversary $\mathcal{B}$. The reduction $\mathcal{B}$ starts the MDDH problem with parameter $n = \lambda \in \mathbb{N}$, receives $(E_1, \cdots, E_\lambda)$, and proceeds as follows.

1. First, $\mathcal{B}$ simulates the oracle of $\mathcal{O}(\mathsf{FS} \,\|\, \cdot)$, $\mathcal{O}(\mathsf{FS} \,\|\, \mathsf{MT})$ and $\mathcal{O}(\mathsf{PRNG} \,\|\, \cdot)$ by keeping lists $L_{\mathsf{FS}}, L_{\mathsf{MT}}$, and L_{PRNG} respectively using the straight-line and on-the-fly method. $\mathcal{B}$ also keeps a list L to simulate the

oracle queries. Take $\mathcal{O}(\mathsf{FS}\,\|\,\cdot)$ for instance; upon receiving an oracle query as $\mathcal{O}(\mathsf{FS}\,\|\,x)$, the $\mathcal{B}$ simulates the oracle as follows.

   (a) Check whether there exists a pair $(x, y) \in L_{\mathsf{FS}}$ for some y. If so, return y.
   (b) Otherwise draw $y \leftarrow \{0,1\}^\lambda$ uniformly at random. Add y to the list (x, y) and return y.

2. Generates $c_0, c_1 \leftarrow G$.
3. Invoke $\mathcal{A}$ with $\mathsf{vk} = (c_0 \star E_0, c_1 \star E_0, E_1, \cdots, E_\lambda)$.
4. Upon receiving the evaluation query $x \in \{0,1\}^\lambda$, forward the query x to the MDDH problem oracle and recieve E. Run the simulator in Thm. 4.7 to produce a proof $\pi \leftarrow \mathsf{Sim}(E_0, \mathsf{vk}_I, (c_0c_1) \star E)$ where $I = \{1,2\} \cup \{i + 2 | x_i = 1 \wedge i \in [\lambda]\}$. Return (x, π) to $\mathcal{A}$.
5. Upon receiving the challenge $\widetilde{x}$, forward the challenge to $\widetilde{x}$ to the MDDH problem challenger and obtains v_b. Forward v_b to $\mathcal{A}$ and output whatever $\mathcal{A}$ returns.

When the MDDH problem challenger using the random coin $b \in \{0,1\}$ in the experiment (Def. 2.22 Item 4). $\mathcal{B}$ creates Game_1 using the same random coin b. Therefore,

$$\begin{aligned}\mathsf{Adv}_1(\mathcal{A}) &= |\Pr[\mathcal{A}(\mathsf{Game}_i(b=1)) \to 1] - \Pr[\mathcal{A}(\mathsf{Game}_i(b=0)) \to 1]| \\ &= \left|\Pr[\mathcal{B}_2(\mathsf{Exp}^{\mathsf{MDDH}}(\lambda, 1)) \to 1] - \Pr[\mathcal{B}_2(\mathsf{Exp}^{\mathsf{MDDH}}(\lambda, 0)) \to 1]\right| \\ &= \mathsf{Adv}^{\mathsf{MDDH}}(\mathcal{B}).\end{aligned}$$

Hence,

$$\mathsf{Adv}^{\mathsf{PR}}_{\Pi_{\mathsf{VRF}}}(\mathcal{A}) \leq \frac{q_{\mathsf{PRNG}}}{2^\lambda} + \frac{q_{\mathsf{FS}}}{2^{2\lambda}} + \mathsf{Adv}^{\mathsf{MDDH}}(\mathcal{B}).$$

□

6 TSUBAKI - Twist-Square-Based Tweak for Isogenies

This section presents the variant using the CSIDH-based action with quadratic twists. Let $(G, \mathcal{E}, \star, E_0)$ denote the group action where $E_0 \in \mathcal{E}$ denotes the element that has the property that $E_0^t = E_0$. Also, for any $(g, E) \in G \times \mathcal{E}$, we have $(g \star E)^t = g^{-1} \star E^t$.

The high-level idea of the optimization is to consider "somewhat" generalized version of Naor-Reingold PRF of form: $c_0c_1s_1^{x_1} \cdots s_\kappa^{x_\kappa} \star h_0$ where $x_i \in \{\pm 1, 0\}$ for every i. Hence, by taking a twist on the public keys $(g_0 \star E_0, g_1 \star E_0, s_1 \star E_0, \cdots, s_\kappa \star E_0)$, an additional public key is obtained, with the corresponding secret key being the inverse of the original one.

This results in a reduction in the sizes of the secret key, public key, and proof size without incurring additional overhead. As illustrated in App. A, we can show that the underlying assumption is as hard as the square DDH problem. We believe this problem is trustworthy as the hardness is justified in the generic model introduced in a recent work [DHK+23]. Moreover, the computational version has been shown to be equivalent to the standard CDH problem [LGD21].

It is important to note that the Naor-Reingold PRF in the group action setting for a larger space of x_i than $\{\pm 1, 0\}$ may not be secure due to the underlying group structure, which contains small subgroups.

A variant of CAPYBARA is described as follows.

Construction.

The complete probability and the unique provability hold naturall by embedding $\Pi_{\mathsf{VRF}^\star}$ in Fig. 4 back to Π_{VRF} in Fig. 3. We therefore skip the proofs here. We only show the residual pseudorandomness of $\Pi_{\mathsf{VRF}^\star}$.

Theorem 6.1. *The VRF construction $\Pi_{\mathsf{VRF}^\star}$ in Fig. 4 has complete provability.*

```
$\mathsf{ParGen}(1^\lambda)$
Generate $\mathsf{pp} = (G, \star, E_0, \mathcal{E})$
return $\mathsf{pp}$
```

```
$\mathsf{KeyGen}(\mathsf{pp})$
$(G, \star, E_0, \mathcal{E}) \leftarrow \mathsf{pp}$
$\mathsf{sk} \leftarrow G^{\kappa+2}$
$\mathsf{vk} = \mathsf{sk} \star E_0$
return $(\mathsf{vk}, \mathsf{sk})$
```

```
$\mathsf{Expand}_s(\mathsf{sk})$
$(c_0, c_1, s_1, \cdots, s_\kappa) \leftarrow \mathsf{sk}$
return $(c_0, c_1, s_1, \cdots, s_\kappa, -s_1, \cdots, -s_\kappa)$
```

```
$\mathsf{Expand}_v(\mathsf{vk})$
$(X_0, X_1, E_1, \cdots, E_\kappa) \leftarrow \mathsf{vk}$
return $(X_0, X_1, E_1, \cdots, E_\kappa, E_1^t, \cdots, E_\kappa^t)$
```

```
$\mathsf{VRFEval}(\mathsf{sk}, x)$
$(G, \star, E_0, \mathcal{E}) \leftarrow \mathsf{pp}$
$v = E_0$
$I \leftarrow \{1, 2\}$
for $i \in [\kappa]$ do
    if $x_i = 1$ then
        $I \leftarrow I \cup \{i + 2\}$
    if $x_i = -1$ then
        $I \leftarrow I \cup \{i + \kappa + 2\}$
$\mathsf{sk}', \mathsf{vk}' \leftarrow \mathsf{Expand}_s(\mathsf{sk}), \mathsf{Expand}_v(\mathsf{vk})$
for $s \in \mathsf{sk}'_I$ do
    $v \leftarrow s \star v$
$\pi \leftarrow (P(\mathsf{st} = (E_0, \mathsf{vk}'_I, v), \mathsf{wt} = \mathsf{sk}'_I))$
return $(v, \pi)$
```

```
$\mathsf{VRFVer}(\mathsf{vk}, v, x, \pi)$
$(G, \star, E_0, \mathcal{E}) \leftarrow \mathsf{pp}$
for $i \in [\kappa]$ do
    if $x_i = 1$ then
        $I \leftarrow I \cup \{i + 2\}$
    if $x_i = -1$ then
        $I \leftarrow I \cup \{i + \kappa + 2\}$
$\mathsf{vk}' \leftarrow \mathsf{Expand}_v(\mathsf{vk})$
return $V(\mathsf{st} = (E_0, \mathsf{vk}'_I, v), \pi)$
```

Figure 4: Our verifiable random function scheme $\Pi_{\mathsf{VRF}^\star}$ based on the sDDH problem where $\Pi_{\mathsf{NIZK}}^{\mathsf{fac}} = (P, V)$ is an NIZK for the relation R_{fac} described in Sec. 4.2. The input x is ternary of length $\kappa = \lceil \lambda / \log_2(3) \rceil$.

Theorem 6.2. *If* Π_{NIZK} *is extractable, the VRF construction* $\Pi_{\mathsf{VRF}^\star}$ *in Fig. 4 has unique provability in the random oracle model. Concretely, for any unique provability adversary* $\mathcal{A}$ *against* $\Pi_{\mathsf{VRF}^\star}$*, there exists an extractable adversary* $\mathcal{B}$ *against* Π_{NIZK} *such that*

$$\mathsf{Adv}_{\Pi_{\mathsf{VRF}^\star}}^{\mathsf{UP}}(\mathcal{A}) \leq 2\mathsf{Adv}_{\Pi_{\mathsf{NIZK}}}^{\mathsf{OE}}(\mathcal{B}).$$

Theorem 6.3. *If the twist decisional master DDH problem is hard, then the VRF construction* $\Pi_{\mathsf{VRF}^\star}$ *in Fig. 4, with a subroutine* $\Pi_{\mathsf{NIZK}} = (P, V)$ *in Fig. 2, has (residual) pseudorandomness. Concretely, for any residual pseudorandomness adversary* $\mathcal{A}$ *against* $\Pi_{\mathsf{VRF}^\star}$ *with at most* q_{PRNG} *queries of* $\mathcal{O}(\mathsf{PRNG} \parallel \cdot)$ *and* q_{FS} *queries of* $\mathcal{O}(\mathsf{FS} \parallel \cdot)$*, there exists a* tMDDH *adversary* $\mathcal{B}$ *such that*

$$\mathsf{Adv}_{\Pi_{\mathsf{VRF}^\star}}^{\mathsf{PR}}(\mathcal{A}) \leq \frac{q_{\mathsf{PRNG}}}{2^\lambda} + \frac{q_{\mathsf{FS}}}{2^{2\kappa}} + \mathsf{Adv}^{\mathsf{tMDDH}}(\mathcal{B}).$$

Proof. We show by using a hybrid argument that such an adversary $\mathcal{A}$ can be transformed into a tMDDH adversary $\mathcal{B}_2$. Let Game_0 be the original residual pseudorandomness experiment and Game_1 be the modified experiment. For $i \in \{0, 1\}$, we denote the advantage of $\mathcal{A}$ in Game_i by $\mathsf{Adv}_i(\mathcal{A}) = |\Pr[\mathcal{A}(\mathsf{Game}_i(b = 1)) \to 1] - \Pr[\mathcal{A}(\mathsf{Game}_i(b = 0)) \to 1]|$, where $b \in \{0, 1\}$ represents the random coin chosen by the challenger (Def. 2.10 Item 7). Since Game_0 be the original experiment, we know $\mathsf{Adv}_0(\mathcal{A}) = \mathsf{Adv}_{\Pi_{\mathsf{VRF}}}^{\mathsf{PR}}(\mathcal{A})$ by definition.

We introduce Game_1, which is the same as Game_0 except for the way to respond to an evaluation query. Rather than generated via Prove from the subroutine Π_{NIZK}, the proof is generated using the simulator Sim for Π_{NIZK} in Thm. 4.7. Here, we have the parameter $N \geq 2$ in Thm. 4.7 since the prover uses at least two elements ($\mathsf{sk}_{\{1,2\}} \subseteq \mathsf{sk}_I$) to generate a proof. By Thm. 4.7, since the simulator Sim is statistically indistinguishable from a real prover, the change in Game_1 results in a negligible loss. Concretely, $|\mathsf{Adv}_0(\mathcal{A})-\mathsf{Adv}_1(\mathcal{A})| \leq \frac{q_{\mathsf{PRNG}}}{2^{\lambda}}+\frac{q_{\mathsf{FS}}}{2^{2\kappa}}$.

We now transform an adversary in Game_1 into a tMDDH problem adversary $\mathcal{B}$. The reduction $\mathcal{B}$ starts the tMDDH problem with parameter $n=\kappa \in \mathbb{N}$, receives $(E,(E_1,\cdots,E_\kappa))$, and proceeds as follows.

1. Firstly, $\mathcal{B}$ simulates the oracle of $\mathcal{O}(\mathsf{FS}\parallel\cdot)$, $\mathcal{O}(\mathsf{FS}\parallel\mathsf{MT})$ and $\mathcal{O}(\mathsf{PRNG}\parallel\cdot)$ by keeping lists $L_{\mathsf{FS}}, L_{\mathsf{MT}}$, and L_{PRNG} respectively using the straight-line and on-the-fly method. $\mathcal{B}$ also keeps a list L to simulate the oracle queries. Take $\mathcal{O}(\mathsf{FS}\parallel\cdot)$ for instance; upon receiving an oracle query as $\mathcal{O}(\mathsf{FS}\parallel x)$, the $\mathcal{B}$ simulates the oracle as follows.
    (a) Check whether there exists a pair $(x,y)\in L_{\mathsf{FS}}$ for some y. If so, return y.
    (b) Otherwise draw $y \leftarrow \{0,1\}^\kappa$ uniformly at random. Add y to the list (x,y) and return y.
2. Generates $c_0, c_1 \leftarrow G$.
3. Invoke $\mathcal{A}$ with $\mathsf{vk}=(c_0\star E_0, E, E_1,\cdots,E_\kappa)$.
4. Upon receiving the evaluation query $x\in\{0,\pm1\}^\kappa$, forward the query x to the tMDDH problem oracle and recieve E. Write $\mathsf{vk}'=(c_0\star E_0, c_0\star E, E_1,\cdots,E_\kappa,E_1^t,\cdots,E_\kappa^t)$ and $I=\{1,2\}\cup\{i+2|x_i=1\wedge i\in[\kappa]\}\cup\{i+2+N|x_i=-1\wedge i\in[\kappa]\}$. Run the simulator in Thm. 4.7 to produce a proof $\pi\leftarrow\mathsf{Sim}(E_0,\mathsf{vk}'_I,c_0\star E)$. Return (x,π) to $\mathcal{A}$.
5. Upon receiving the challenge $\widetilde{x}$, forward the challenge to $\widetilde{x}$ to the tMDDH problem challenger and obtains v_b. Forward v_b to $\mathcal{A}$ and output whatever $\mathcal{A}$ returns.

When the tMDDH problem challenger using the random coin $b\in\{0,1\}$ in the experiment (Def. 2.22 Item 4). $\mathcal{B}$ creates Game_1 using the same random coin b. Therefore,

$$\begin{aligned}\mathsf{Adv}_1(\mathcal{A}) &= |\Pr[\mathcal{A}(\mathsf{Game}_i(b=1))\to 1]-\Pr[\mathcal{A}(\mathsf{Game}_i(b=0))\to 1]|\\ &= \left|\Pr[\mathcal{B}_2(\mathsf{Exp}^{\mathsf{tMDDH}}(\kappa,1))\to 1]-\Pr[\mathcal{B}_2(\mathsf{Exp}^{\mathsf{tMDDH}}(\kappa,0))\to 1]\right|\\ &= \mathsf{Adv}^{\mathsf{tMDDH}}(\mathcal{B}).\end{aligned}$$

Hence,

$$\mathsf{Adv}^{\mathsf{PR}}_{\Pi_{\mathsf{VRF}*}}(\mathcal{A}) \leq \frac{q_{\mathsf{PRNG}}}{2^{\lambda}}+\frac{q_{\mathsf{FS}}}{2^{2\kappa}}+\mathsf{Adv}^{\mathsf{tMDDH}}(\mathcal{B}).$$

□

7 Optimization and Performance

We ameliorate the proof size by utilizing the two techniques presented in [BKP20], briefly summarised as follows.

Unbalanced Challenge Space. One can observe the response of a prover in the proof system Fig. 2 for the challenge 0 is a seed, which is much shorter than that for challenge one. By introducing the unbalanced challenge space $C_{M,K}=\{\mathsf{ch}\in\{0,1\}^M \mid |\mathsf{ch}|=K\}$, where $|\cdot|$ is the ℓ_1-norm and $2^\lambda \leq \frac{M!}{K!(M-K)!}$. We thereby obtain a much smaller proof size and the online-extractability and zero-knowledge remain the same.

Seed Trees. The seed tree technique allows the prover to produce a large amount of the seeds using PRNG and iteratively generating binary subtrees. The leaves of the tree are the seeds to be used. The prover can later reveal the generating nodes while not disclosing the information of those unrevealed leaves. The method reduces the size of responses for the challenge 0 in our case. Though the proof size regarding this technique is not fixed, we will calculate the worst case for the proof size estimation.

The performances of CAPYBARA and TSUBAKI are given in Tab. 1 for the input space $\{0,1\}^{128}$. CAPYBARA is based on the standard DDH assumption while TSUBAKI is based on the stronger square DDH (Def. 2.23). We use the group action from CSIDH-512, as specified in [BKV19], with $M = 855$ and $K = 19$ as the unbalanced challenge space in our implementation. Our proof sizes are flexible and depend on the input length, with lengths of approximately $79|x|/128$ for CAPYBARA and $51|x|/81$ for TSUBAKI. The group action from CSIDH-512 has been estimated to have 128 bits of classical security and over 63 bits of quantum security [CSCJR22]. As stated in [CSCJR22], the caculation does not take the cost of group action evaluatings into account and is like to understimate security. As explained in Sec. 2.4, we are not able to provide the numbers for a stronger security level.

We also compare our VRFs to other existing post-quantum VRFs, including LB-VRF [EKS+21], X-VRF, SL-VRF [BDE+22], LaV [ESLR23], and iVRF [EEK+22], all aiming to meet the NIST II security level. LB-VRF, X-VRF, and iVRF have limited residual pseudorandomness, while SL-VRF, LaV, and our VRFs are full VRFs. iVRF, tailored to their applications, also relaxes the unique provability by imposing one more restriction on the adversary (see CFU of [EEK+22] on P7). iVRF's evaluation size is viewed as zero since one can recover the evaluation from the proof (see Table I of [EEK+22]). The security of X-VRF, SL-VRF, and iVRF is based on XMSS, LowMC, and SHA-256, respectively, and LB-VRF and LaV rely on a hybrid lattice assumption MSIS/MLWE and MSIS/MLWR respectively. We add the new isogeny-based VRF by Leroux [Ler23] to the table, as introduced in Rem. 1.1.

	$\vert\mathsf{sk}\vert$	$\vert\mathsf{vk}\vert$	$\vert v\vert$	$\vert\pi\vert$	Assumption	Relaxation	Security Level
CAPYBARA [Fig. 3]	32B	8.3KB	64B	39 KB	DDH		≥63
TSUBAKI [Fig. 4]	32B	5.3KB	64B	34 KB	sDDH		≥63
LB-VRF I [EKS+21]		3.3KB	84B	4.9KB	MSIS/MLWE	1-Time	NIST II
LB-VRF III [EKS+21]		3.4KB	84B	7.3KB	MSIS/MLWE	5-Time	NIST II
X-VRF [BDE+22]	132B	64B	32B	2.6 KB	XMSS	2^{15}-Time	NIST II
SL-VRF [BDE+22]	24B	48B	32B	40 KB	LowMC		NIST II
LaV [ESLR23]	6.4KB	3.4KB	124B	12 KB	MSIS/MLWR		NIST II
iVRF [EEK+22]		32B	0B	608 B	SHA-256	2^{18}/ CFU	NIST II
$\mathsf{DeuringVRF}_{2,1}$ [Ler23]		944B	32B	224 B	$\mathsf{OMIP}_{\mathsf{2dim}}$		NIST I
$\mathsf{DeuringVRF}_{4,2}$ [Ler23]		192B	32B	112 B	$\mathsf{OMIP}_{\mathsf{2dim}}$		NIST I

Table 1: CAPYBARA and TSUBAKI (Figs. 3 and 4, resp) using the group action setting CSIDH-512 instantiated in [BKV19]. The unbalanced challenge space $C_{M,K}$ where $M = 855, K = 19$ is used in the proof system Fig. 2. Note that our original secret key sizes are 2KB, 1.3KB, respectively, and one can use a 32B seed to generate the entire secret key sk using a PRNG. Our proof sizes are $\approx 79|x|/128$ and $\leq 51|x|/81$ respectively and vary with the density $|x|/\kappa$ of the input x where $|\cdot|$ is the ℓ_1-norm. The notations $|\mathsf{sk}|, |\mathsf{vk}|, |v|, |\pi|$ represent the length of the secret key, verification key, output, and proof, respectively. The security of X-VRF, SL-VRF, and iVRF is based on XMSS, LowMC, and SHA-256, respectively, and LB-VRF and LaV rely on a hybrid lattice assumption MSIS/MLWE and MSIS/MLWR respectively.

Acknowledgement

Yi-Fu Lai was supported by the Ministry for Business, Innovation and Employment in New Zealand. We would like to express our gratitude to Steven Galbraith, Muhammed Esgin, and the anonymous reviewers for their valuable editorial suggestions that have helped to enhance the presentation of this work.

References

[ACF14] Michel Abdalla, Dario Catalano, and Dario Fiore. Verifiable random functions: Relations to identity-based key encapsulation and new constructions. *Journal of Cryptology*, 27(3):544–593, July 2014.

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

[AEK$^+$22] Michel Abdalla, Thorsten Eisenhofer, Eike Kiltz, Sabrina Kunzweiler, and Doreen Riepel. Password-authenticated key exchange from group actions. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part II*, volume 13508 of *LNCS*, pages 699–728. Springer, Heidelberg, August 2022.

[BBD$^+$22] Jeremy Booher, Ross Bowden, Javad Doliskani, Tako Boris Fouotsa, Steven D. Galbraith, Sabrina Kunzweiler, Simon-Philipp Merz, Christophe Petit, Benjamin Smith, Katherine E. Stange, Yan Bo Ti, Christelle Vincent, José Felipe Voloch, Charlotte Weitkämper, and Lukas Zobernig. Failing to hash into supersingular isogeny graphs. Cryptology ePrint Archive, Report 2022/518, 2022. `https://eprint.iacr.org/2022/518`.

[BCC$^+$23] Andrea Basso, Giulio Codogni, Deirdre Connolly, Luca De Feo, Tako Boris Fouotsa, Guido Maria Lido, Travis Morrison, Lorenz Panny, Sikhar Patranabis, and Benjamin Wesolowski. Supersingular curves you can trust. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part II*, volume 14005 of *LNCS*, pages 405–437. Springer, Heidelberg, April 2023.

[BDE$^+$22] Maxime Buser, Rafael Dowsley, Muhammed F. Esgin, Shabnam Kasra Kermanshahi, Veronika Kuchta, Joseph K. Liu, Raphaël C.-W. Phan, and Zhenfei Zhang. Post-quantum verifiable random function from symmetric primitives in PoS blockchain. In Vijayalakshmi Atluri, Roberto Di Pietro, Christian Damsgaard Jensen, and Weizhi Meng, editors, *ESORICS 2022, Part I*, volume 13554 of *LNCS*, pages 25–45. Springer, Heidelberg, September 2022.

[BDK$^+$22] Ward Beullens, Samuel Dobson, Shuichi Katsumata, Yi-Fu Lai, and Federico Pintore. Group signatures and more from isogenies and lattices: Generic, simple, and efficient. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part II*, volume 13276 of *LNCS*, pages 95–126. Springer, Heidelberg, May / June 2022.

[BGLS03] Dan Boneh, Craig Gentry, Ben Lynn, and Hovav Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In Eli Biham, editor, *EUROCRYPT 2003*, volume 2656 of *LNCS*, pages 416–432. Springer, Heidelberg, May 2003.

[BKM$^+$21] Andrea Basso, Péter Kutas, Simon-Philipp Merz, Christophe Petit, and Antonio Sanso. Cryptanalysis of an oblivious PRF from supersingular isogenies. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part I*, volume 13090 of *LNCS*, pages 160–184. Springer, Heidelberg, December 2021.

[BKP20] Ward Beullens, Shuichi Katsumata, and Federico Pintore. Calamari and Falafl: Logarithmic (linkable) ring signatures from isogenies and lattices. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part II*, volume 12492 of *LNCS*, pages 464–492. Springer, Heidelberg, December 2020.

[BKV19] 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, *ASIACRYPT 2019, Part I*, volume 11921 of *LNCS*, pages 227–247. Springer, Heidelberg, December 2019.

[BKW20] Dan Boneh, Dmitry Kogan, and Katharine Woo. Oblivious pseudorandom functions from isogenies. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part II*, volume 12492 of *LNCS*, pages 520–550. Springer, Heidelberg, December 2020.

[BLMW07] Emmanuel Bresson, Yassine Lakhnech, Laurent Mazaré, and Bogdan Warinschi. A generalization of DDH with applications to protocol analysis and computational soundness. In Alfred Menezes, editor, *CRYPTO 2007*, volume 4622 of *LNCS*, pages 482–499. Springer, Heidelberg, August 2007.

[BLS01] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the Weil pairing. In Colin Boyd, editor, *ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 514–532. Springer, Heidelberg, December 2001.

[BMM$^+$22] Saikrishna Badrinarayanan, Daniel Masny, Pratyay Mukherjee, Sikhar Patranabis, Srinivasan Raghuraman, and Pratik Sarkar. Round-optimal oblivious transfer and mpc from computational csidh. Cryptology ePrint Archive, Paper 2022/1511, 2022. https://eprint.iacr.org/2022/1511.

[BMR10] Dan Boneh, Hart William Montgomery, and Ananth Raghunathan. Algebraic pseudorandom functions with improved efficiency from the augmented cascade. In Ehab Al-Shaer, Angelos D. Keromytis, and Vitaly Shmatikov, editors, *ACM CCS 2010*, pages 131–140. ACM Press, October 2010.

[Boy08] Xavier Boyen. The uber-assumption family (invited talk). In Steven D. Galbraith and Kenneth G. Paterson, editors, *PAIRING 2008*, volume 5209 of *LNCS*, pages 39–56. Springer, Heidelberg, September 2008.

[BPR12] Abhishek Banerjee, Chris Peikert, and Alon Rosen. Pseudorandom functions and lattices. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 719–737. Springer, Heidelberg, April 2012.

[BS20] Xavier Bonnetain and André Schrottenloher. Quantum security analysis of CSIDH. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part II*, volume 12106 of *LNCS*, pages 493–522. Springer, Heidelberg, May 2020.

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

[CHVW22] Wouter Castryck, Marc Houben, Frederik Vercauteren, and Benjamin Wesolowski. On the decisional diffie–hellman problem for class group actions on oriented elliptic curves. *Research in Number Theory*, 8(4):99, 2022.

[CL23] Mingjie Chen and Antonin Leroux. Scallop-hd: group action from 2-dimensional isogenies. Cryptology ePrint Archive, Paper 2023/1488, 2023. https://eprint.iacr.org/2023/1488.

[CLG09] Denis Xavier Charles, Kristin E. Lauter, and Eyal Z. Goren. Cryptographic hash functions from expander graphs. *Journal of Cryptology*, 22(1):93–113, January 2009.

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

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

[CSV20] Wouter Castryck, Jana Sotáková, and Frederik Vercauteren. Breaking the decisional Diffie-Hellman problem for class group actions using genus theory. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part II*, volume 12171 of *LNCS*, pages 92–120. Springer, Heidelberg, August 2020.

[DDF$^+$21] Luca De Feo, Cyprien Delpech de Saint Guilhem, Tako Boris Fouotsa, Péter Kutas, Antonin Leroux, Christophe Petit, Javier Silva, and Benjamin Wesolowski. Séta: Supersingular encryption from torsion attacks. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part IV*, volume 13093 of *LNCS*, pages 249–278. Springer, Heidelberg, December 2021.

[DF19] Luca De Feo. Seasign: Compact isogeny signatures from class group actions, 2019. Talk at Eurocrypt 2019.

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

[DG19] Luca De Feo and Steven D. Galbraith. SeaSign: Compact isogeny signatures from class group actions. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part III*, volume 11478 of *LNCS*, pages 759–789. Springer, Heidelberg, May 2019.

[DGKR18] Bernardo David, Peter Gazi, Aggelos Kiayias, and Alexander Russell. Ouroboros praos: An adaptively-secure, semi-synchronous proof-of-stake blockchain. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part II*, volume 10821 of *LNCS*, pages 66–98. Springer, Heidelberg, April / May 2018.

[DHK$^+$23] Julien Duman, Dominik Hartmann, Eike Kiltz, Sabrina Kunzweiler, Jonas Lehmann, and Doreen Riepel. Generic models for group actions. In Alexandra Boldyreva and Vladimir Kolesnikov, editors, *PKC 2023, Part I*, volume 13940 of *LNCS*, pages 406–435. Springer, Heidelberg, May 2023.

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

[DM20] Luca De Feo and Michael Meyer. Threshold schemes from isogeny assumptions. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020, Part II*, volume 12111 of *LNCS*, pages 187–212. Springer, Heidelberg, May 2020.

[DMPS19] Luca De Feo, Simon Masson, Christophe Petit, and Antonio Sanso. Verifiable delay functions from supersingular isogenies and pairings. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part I*, volume 11921 of *LNCS*, pages 248–277. Springer, Heidelberg, December 2019.

[DPV19] Thomas Decru, Lorenz Panny, and Frederik Vercauteren. Faster SeaSign signatures through improved rejection sampling. In Jintai Ding and Rainer Steinwandt, editors, *Post-Quantum Cryptography - 10th International Conference, PQCrypto 2019*, pages 271–285. Springer, Heidelberg, 2019.

[EEK$^+$22] Muhammed F. Esgin, Oguzhan Ersoy, Veronika Kuchta, Julian Loss, Amin Sakzad, Ron Steinfeld, Wayne Yang, and Raymond K. Zhao. A new look at blockchain leader election: Simple, efficient, sustainable and post-quantum. Cryptology ePrint Archive, Report 2022/993, 2022. `https://eprint.iacr.org/2022/993`.

[EKP20] Ali El Kaafarani, Shuichi Katsumata, and Federico Pintore. Lossy CSI-FiSh: Efficient signature scheme with tight reduction to decisional CSIDH-512. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020, Part II*, volume 12111 of *LNCS*, pages 157–186. Springer, Heidelberg, May 2020.

[EKS$^+$21] Muhammed F. Esgin, Veronika Kuchta, Amin Sakzad, Ron Steinfeld, Zhenfei Zhang, Shifeng Sun, and Shumo Chu. Practical post-quantum few-time verifiable random function with applications to algorand. In Nikita Borisov and Claudia Diaz, editors, *FC 2021, Virtual Event, March 1-5, 2021, Revised Selected Papers, Part II*, volume 12675 of *Lecture Notes in Computer Science*, pages 560–578. Springer, 2021.

[ESLR23] Muhammed F. Esgin, Ron Steinfeld, Dongxi Liu, and Sushmita Ruj. Efficient hybrid exact/relaxed lattice proofs and applications to rounding and VRFs. LNCS, pages 484–517. Springer, Heidelberg, August 2023.

[GHM$^+$17] Yossi Gilad, Rotem Hemo, Silvio Micali, Georgios Vlachos, and Nickolai Zeldovich. Algorand: Scaling byzantine agreements for cryptocurrencies. In *Proceedings of the 26th Symposium on Operating Systems Principles, Shanghai, China, October 28-31, 2017*, pages 51–68. ACM, 2017.

[GNP$^+$15] Sharon Goldberg, Moni Naor, Dimitrios Papadopoulos, Leonid Reyzin, Sachin Vasant, and Asaf Ziv. NSEC5: Provably preventing DNSSEC zone enumeration. In *NDSS 2015*. The Internet Society, February 2015.

[GPSV21] Steven Galbraith, Lorenz Panny, Benjamin Smith, and Frederik Vercauteren. Quantum equivalence of the DLP and CDHP for group actions. *Mathematical Cryptology*, 1(1):40–44, Jun. 2021.

[HMW18] Timo Hanke, Mahnush Movahedi, and Dominic Williams. DFINITY technology overview series, consensus system. *CoRR*, abs/1805.04548, 2018.

[Jag15] Tibor Jager. Verifiable random functions from weaker assumptions. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015, Part II*, volume 9015 of *LNCS*, pages 121–143. Springer, Heidelberg, March 2015.

[JF11] David Jao and Luca De Feo. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In Bo-Yin Yang, editor, *Post-Quantum Cryptography - 4th International Workshop, PQCrypto 2011, Taipei, Taiwan, November 29 - December 2, 2011. Proceedings*, volume 7071 of *Lecture Notes in Computer Science*, pages 19–34. Springer, 2011.

[KLLQ23] Shuichi Katsumata, Yi-Fu Lai, Jason T. LeGrow, and Ling Qin. CSI -otter: Isogeny-based (partially) blind signatures from the class group action with a twist. In *CRYPTO 2023, Part III*, LNCS, pages 729–761. Springer, Heidelberg, August 2023.

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

[Kup11] Greg Kuperberg. Another subexponential-time quantum algorithm for the dihedral hidden subgroup problem. *arXiv preprint arXiv:1112.3333*, 2011.

[Lai23] Yi-Fu Lai. *Advanced Isogeny-based Cryptosystems*. Phd thesis, The University of Auckland, 2023.

[Ler23] Antonin Leroux. Verifiable random function from the deuring correspondence and higher dimensional isogenies. Cryptology ePrint Archive, Paper 2023/1251, 2023. `https://eprint.iacr.org/2023/1251`.

[LGD21] Yi-Fu Lai, Steven D. Galbraith, and Cyprien Delpech de Saint Guilhem. Compact, efficient and UC-secure isogeny-based oblivious transfer. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part I*, volume 12696 of *LNCS*, pages 213–241. Springer, Heidelberg, October 2021.

[Lyu09] Vadim Lyubashevsky. Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures. In Mitsuru Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 598–616. Springer, Heidelberg, December 2009.

[MMP22] Marzio Mula, Nadir Murru, and Federico Pintore. Random sampling of supersingular elliptic curves. Cryptology ePrint Archive, Report 2022/528, 2022. https://eprint.iacr.org/2022/528.

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

[Mon18] Hart Montgomery. More efficient lattice PRFs from keyed pseudorandom synthesizers. In Debrup Chakraborty and Tetsu Iwata, editors, *INDOCRYPT 2018*, volume 11356 of *LNCS*, pages 190–211. Springer, Heidelberg, December 2018.

[MOT20] Tomoki Moriya, Hiroshi Onuki, and Tsuyoshi Takagi. SiGamal: A supersingular isogeny-based PKE and its application to a PRF. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part II*, volume 12492 of *LNCS*, pages 551–580. Springer, Heidelberg, December 2020.

[MRV99] Silvio Micali, Michael O. Rabin, and Salil P. Vadhan. Verifiable random functions. In *40th FOCS*, pages 120–130. IEEE Computer Society Press, October 1999.

[MZ22] Hart Montgomery and Mark Zhandry. Full quantum equivalence of group action DLog and CDH, and more. In Shweta Agrawal and Dongdai Lin, editors, *ASIACRYPT 2022, Part I*, volume 13791 of *LNCS*, pages 3–32. Springer, Heidelberg, December 2022.

[NP01] Moni Naor and Benny Pinkas. Efficient oblivious transfer protocols. In S. Rao Kosaraju, editor, *12th SODA*, pages 448–457. ACM-SIAM, January 2001.

[NR99] Moni Naor and Omer Reingold. Synthesizers and their application to the parallel construction of pseudo-random functions. *Journal of Computer and System Sciences*, 58(2):336–375, 1999.

[Pan23] Lorenz Panny. Csi-fish really isn't polynomial-time, 2023. https://yx7.cc/blah/ 2023-04-14.html.

[Pas03] Rafael Pass. On deniability in the common reference string and random oracle model. In Dan Boneh, editor, *CRYPTO 2003*, volume 2729 of *LNCS*, pages 316–337. Springer, Heidelberg, August 2003.

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

[PR23] Aurel Page and Damien Robert. Introducing clapoti(s): Evaluating the isogeny class group action in polynomial time. Cryptology ePrint Archive, Paper 2023/1766, 2023. https://eprint.iacr.org/2023/1766.

[PWH$^+$17] Dimitrios Papadopoulos, Duane Wessels, Shumon Huque, Moni Naor, Jan Včelák, Leonid Reyzin, and Sharon Goldberg. Making NSEC5 practical for DNSSEC. Cryptology ePrint Archive, Report 2017/099, 2017. https://eprint.iacr.org/2017/099.

[Reg04] Oded Regev. A subexponential time algorithm for the dihedral hidden subgroup problem with polynomial space. *arXiv preprint quant-ph/0406151*, 2004.

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

[Sho99] Peter W Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41(2):303–332, 1999.

[Unr15] Dominique Unruh. Non-interactive zero-knowledge proofs in the quantum random oracle model. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 755–784. Springer, Heidelberg, April 2015.

[YAJ+17] Youngho Yoo, Reza Azarderakhsh, Amir Jalali, David Jao, and Vladimir Soukharev. A post-quantum digital signature scheme based on supersingular isogenies. In Aggelos Kiayias, editor, *FC 2017*, volume 10322 of *LNCS*, pages 163–181. Springer, Heidelberg, April 2017.

A Hardness of Twisted Master Decisional Problem

We start from a quick recap of the assumptions in Sec. 2.6.

Definition A.1 (Decisional Square CSIDH (sDDH) Problem)**.** *Let* $(G, \mathcal{E}, \star, E_0)$ *be a group action. The decisional square CSIDH problem is that the adversary* $\mathcal{A}$ *is given* $T_b = (g_1 \star E_0, h_b \star E_0)$ *where* $h_0 = g_1^2, h_1 = g_2$ *and* $(g_1, g_2, b) \leftarrow G^2 \times \{0,1\}$ *and return* $b \in \{0,1\}$.

Definition A.2 (Decisional Reciprocal CSIDH (rDDH) Problem)**.** *Let* $(G, \mathcal{E}, \star, E_0)$ *be a group action. The decisional reciprocal CSIDH problem is that the adversary* $\mathcal{A}$ *is given* $T_b = (g_1 \star E_0, g_2 \star E_0, h_b \star E_0, h'_b \star E_0)$ *where* $h_0 = g_1 g_2, h_1 = g_3, h'_0 = g_1 g_2^{-1}, h'_1 = g_4$ *and* $(g_1, g_2, g_3, g_4, b) \leftarrow G^4 \times \{0,1\}$, *and return* $b' \in \{0,1\}$.

Definition A.3 (Multi-challenge Decisional Reciprocal CSIDH (mcrDDH) Problem)**.** *Let* $(G, \mathcal{E}, \star, E_0)$ *be a group action and* $b \in \{0,1\}$. *The multi-challenge decisional reciprocal Diffie-Hellman experiment* $\mathsf{Exp}^{\mathsf{mcrDDH}}(b)$ *on input* b *proceeds as follows. The adversary* $\mathcal{A}$ *is given* $(g_1 \star E_0)$ *where* $g_1 \leftarrow G$ *together with access to* $\mathcal{O}_b^{\mathsf{mcrDDH}}$ *defined as follows:*

1. $\mathcal{O}_0^{\mathsf{mcrDDH}}$: $(g_2 \star E_0, (g_1 g_2) \star E_0, (g_1 g_2^{-1}) \star E_0)$ *where* $g_2 \leftarrow G$,
2. $\mathcal{O}_1^{\mathsf{mcrDDH}}$: $(g_2 \star E_0, g_3 \star E_0, g_4 \star E_0)$ *where* $g_2, g_3, g_4 \leftarrow G$,

and outputs $b' \in \{0,1\}$.

We denote the advantage of a mcrDDH problem adversary $\mathcal{A}$ problem by

$$\mathsf{Adv}^{\mathsf{mcrDDH}}(\mathcal{A}) = \left| \Pr[\mathcal{A}(\mathsf{Exp}^{\mathsf{mcrDDH}}(b=0)) \to 1] - \Pr[\mathcal{A}(\mathsf{Exp}^{\mathsf{mcrDDH}}(b=1)) \to 1] \right|,$$

where b is the randomness in the experiment, and the probability is taken over the randomness used by $\mathcal{A}$ and the randomness used in the experiment.
The group action $(G, \mathcal{E}, \star, E_0)$ is implicitly parameterized in the experiment. We say the mcrDDH problem is hard, if for any PPT adversary $\mathcal{A}$, there exists a negligible function negl such that $\mathsf{Adv}^{\mathsf{mcrDDH}}(\mathcal{A}) \leq \mathsf{negl}(\lambda)$. One can use a standard hybrid argument to give a reduction from the rDDH problem to the mcrDDH problem. We skip the proof here.

Definition A.4 (Twisted Master Decisional CSIDH (tMDDH) Problem)**.** *Let* $(G, \mathcal{E}, \star, E_0)$ *be a group action,* $n \in \mathbb{N}$, *and* $b \in \{0,1\}$. *The twisted master DDH problem experiment* $\mathsf{Exp}^{\mathsf{tMDDH}}(n, b)$ *on input* (n, b) *proceeds as follows.*

1. *The challenger $\mathcal{C}$ computes $E = g \star E_0$ where $g \leftarrow G$.*
2. *$\mathcal{C}$ generates a tuple $(g_1 \star E, \cdots, g_n \star E)$ where $g_1, \cdots, g_n \leftarrow G$, and sends the tuple to the adversary $\mathcal{A}$.*
3. *$\mathcal{A}$ is given access to a Diffie-Hellman (DH) oracle on input $(x_1, \cdots, x_n) \in \{0, \pm 1\}^n$ returning $\prod_i^n g_i^{x_i} \star E$.*
4. *$\mathcal{A}$ sends a string $v = (v_1, \cdots, v_n) \in \{0, \pm 1\}^n$ to $\prod_i^n g_i^{v_i} \star E$ to the challenge oracle $\mathcal{C}$.*
5. *$\mathcal{C}$ ignores if v has been queried before or is of Hamming weight less than 2. Otherwise, $\mathcal{C}$, depending on b, computes $X_0 = \prod_i^n g_i^{v_i} \star E$ or $X_1 = r \star E$ for some $r \leftarrow G$, and send X_b to $\mathcal{A}$. This process will only output for one time.*
6. *$\mathcal{A}$ outputs $b' \in \{0, 1\}$.*

We denote the advantage of the decisional problem adversary $\mathcal{A}$ by

$$\mathsf{Adv}^{\mathsf{tMDDH}}(\mathcal{A}) = \left| \Pr[\mathcal{A}(\mathsf{Exp}^{\mathsf{tMDDH}}(n, b = 0)) \to 1] - \Pr[\mathcal{A}(\mathsf{Exp}^{\mathsf{tMDDH}}(n, b = 1)) \to 1] \right|,$$

where b is the randomness in the experiment, and the probability is taken over the randomness used by $\mathcal{A}$ and the randomness used in the experiment. The group action $(G, \mathcal{E}, \star, E_0)$ is implicitly parameterized in the experiment. We say the tMDDH problem is hard, if for any PPT adversary $\mathcal{A}$, there exists a negligible function negl such that $\mathsf{Adv}^{\mathsf{tMDDH}}(\mathcal{A}) \leq \mathsf{negl}(\lambda)$.

Theorem A.5. *The* tMDDH *problem is not easier than the* mcrDDH *problem. Concretely, let $\mathcal{A}$ be an adversary against the* mcrDDH *problem with the parameter $(G, \mathcal{E}, \star, E_0)$ and $n \in \mathbb{N}$. If at most $q_{\mathsf{DH}} = poly(\lambda)$ queries are made in the* mcrDDH *experiment, then there exists* tMDDH *problem adversaries $\mathcal{B}_2, \cdots \mathcal{B}_n$ such that*

$$\mathsf{Adv}^{\mathsf{tMDDH}}(\mathcal{A}) \leq \sum_{i=2}^{n} \mathsf{Adv}^{\mathsf{mcrDDH}}(\mathcal{B}_i).$$

Proof. We prove the theorem via a hybrid argument by introducing two series of games $\mathsf{Game}_1, \cdots, \mathsf{Game}_n$ by modifying the responses of the DH oracle and the challenge oracle in the tMDDH experiment gradually. Among the games, Game_1 be the original tMDDH experiment, We will modify the response of the challenge oracle and the DH oracle together, which will be explained later. For $i \in [n]$ where $b \in \{0, 1\}$, let $\mathcal{A}(\mathsf{Game}_i(b))$ represent $\mathcal{A}$ running the Game_i, the modified tMDDH experiment with the random coin b used in the experiment, and $\mathcal{A}$ will return 0 or 1. Therefore, by definition,

$$\mathsf{Adv}^{\mathsf{tMDDH}}(\mathcal{A}) = |\Pr[\mathcal{A}(\mathsf{Game}_1(b = 1)) \to 1] - \Pr[\mathcal{A}(\mathsf{Game}_1(b = 0)) \to 1]|. \tag{4}$$

Looking ahead, Game_n is the modified tMDDH experiment where both the DH oracle and the challenger reply with random elements in $\mathcal{E}$. Therefore, since b is information theoretically hidden from $\mathcal{A}$,

$$|\Pr[\mathcal{A}(\mathsf{Game}_n(b = 1)) \to 1] - \Pr[\mathcal{A}(\mathsf{Game}_n(b = 0)) \to 1]| = 0. \tag{5}$$

Game_1 : the original tMDDH experiment starting with a tuple $(g_1 \star E, \cdots, g_n \star E)$ where $g_1, \cdots, g_n \leftarrow G$ and the oracle response as specified.

Game_2 to Game_n: for $j \in \{2, \cdots, n\}$, Game_j is the same as Game_{j-1} except that the response of the DH oracle and the challenge oracle is modified as follows. The modification starts with a list L which is initially

$$\{(\mathbf{0}, E), (\mathbf{e}_1, g_1 \star E), \cdots, (\mathbf{e}_j, g_j \star E)\} \subseteq \{0, \pm 1\}^j \times \mathcal{E}.$$

On the query $x = (x_1, \cdots, x_n) \in \{0, \pm 1\}^n$, if $((x_1, \cdots, x_j), X) \in L$ for some $X \in \mathcal{E}$, the oracle returns $(\prod_{i=j+1}^n g_i^{x_i}) \star X$; otherwise, it draws $g' \leftarrow G$, computes $X = g' \star E$, adds $((x_1, \cdots, x_j), X)$ to the list L, and returns $(\prod_{i=j+1}^n g_i^{x_i}) \star X$ to $\mathcal{A}$. The reply for the challenge query is modified in the same way if the random coin $b = 0$.

Claim that $\mathsf{Game}_{j-1} \approx_c \mathsf{Game}_j$ for $\mathcal{A}$ for any $2 \le j \le n$ by assuming the mcrDDH problem. Concretely, a reduction $\mathcal{B}_j$ to the mcrDDH problem proceeds as follows

1. Obtain $T = (g' \star E_0, \{(X_i, X'_i, X''_i)\}_{i \in [q_{\mathsf{DH}}+j-1]})$ from the mcrDDH oracle.
2. Overwrite the notations of X_i, X'_i, X''_i by $\big(g' \star E, \{X_i, X'_i, X''_i\}_{i \in [q_{\mathsf{DH}}+j-1]}\big) \leftarrow g \star T$ where $g \leftarrow G$.
3. Then, $\mathcal{B}_j$ initializes with a list

$$L = \left\{\begin{matrix}(\mathbf{e}_1, X_1), \cdots, (\mathbf{e}_{j-1}, X_{j-1}), (\mathbf{e}_1 + \mathbf{e}_j, X'_1), \cdots, (\mathbf{e}_{j-1} + \mathbf{e}_j, X'_{j-1}), (\mathbf{0}, E), \\ \hfill (\mathbf{e}_1 - \mathbf{e}_j, X''_1), \cdots, (\mathbf{e}_{j-1} - \mathbf{e}_j, X''_{j-1}), (\mathbf{e}_j, g' \star E)\end{matrix}\right\}$$

   where $\subset \{0, \pm 1\}^j \times \mathcal{E}$, $\mathbf{e}_i$ is the i-th elementary vector in $\{0, \pm 1\}^j$, and set a counter $\mathsf{ct} = j$ to record the number of the pairs (X_i, X'_i, X''_i) taken into the list L.
4. Invoke $\mathcal{A}$ on input $(E, X_1, \cdots, X_{j-1}, g' \star E_0, g_{j+1} \star E_0, \cdots, g_n \star E_0)$ where $g_{j+1}, \cdots, g_n \leftarrow G$.
5. Upon receiving the oracle query $(x_1, \cdots, x_n) \in \{0, \pm 1\}^n$, check whether $((x_1, \cdots, x_j), X) \in L$ for some $X \in \mathcal{E}$. If so, return $\prod_{i=j+1}^{n} g_i^{x_i} \star X$. Otherwise, update

$$L \leftarrow \{((x_1, \cdots, x_{j-1}, 0), X_{\mathsf{ct}}),\ ((x_1, \cdots, x_{j-1}, 1), X'_{\mathsf{ct}}), ((x_1, \cdots, x_{j-1}, -1), X''_{\mathsf{ct}})\} \cup L,$$

   and set $\mathsf{ct} \leftarrow \mathsf{ct} + 1$, and rerun this step again.
6. Output whatever $\mathcal{A}$ returns.

Note that in Step 1. if $\mathcal{B}_j$ is in $\mathsf{Exp}^{\mathsf{mcrDDH}}(0)$ (Def. A.3 Item 1) then $\mathcal{B}_j$ generates Game_{j-1} because $g' \star X_i = X'_i$ and $g'^{-1} \star X_i = X''_i$. In contrast, if it is in $\mathsf{Exp}^{\mathsf{mcrDDH}}(1)$ (Def. A.3 Item 2), then $\mathcal{B}_j$ generates Game_j. It follows that for $b \in \{0, 1\}$,

$$\begin{aligned}\mathsf{Adv}^{\mathsf{mcrDDH}}(\mathcal{B}_j) &= |\Pr[\mathcal{B}_j(\mathsf{Exp}^{\mathsf{mcrDDH}}(0)) \to 1] - \Pr[\mathcal{B}_j(\mathsf{Exp}^{\mathsf{mcrDDH}}(1)) \to 1]| \\ &= |\Pr[\mathcal{A}(\mathsf{Game}_{j-1}(b)) \to 1] - \Pr[\mathcal{A}(\mathsf{Game}_j(b)) \to 1]|. \end{aligned} \tag{6}$$

Therefore, we have

$$\begin{aligned}\mathsf{Adv}^{\mathsf{tMDDH}}(\mathcal{A}) &= |\Pr[\mathcal{A}(\mathsf{Game}_1(b=1)) \to 1] - \Pr[\mathcal{A}(\mathsf{Game}_1(b=0)) \to 1]| && \text{(By Eq. (4))} \\ &\le \sum_{j=2}^{n} (|\Pr[\mathcal{A}(\mathsf{Game}_{j-1}(b=1)) \to 1] - \Pr[\mathcal{A}(\mathsf{Game}_j(b=0)) \to 1]| \\ &\quad + |\Pr[\mathcal{A}(\mathsf{Game}_{j-1}(b=1)) \to 1] - \Pr[\mathcal{A}(\mathsf{Game}_1(b=0)) \to 1]|) \\ &\quad + |\Pr[\mathcal{A}(\mathsf{Game}_n(b=1)) \to 1] - \Pr[\mathcal{A}(\mathsf{Game}_{j-1}(b=1)) \to 1]| && \text{(Union bounds.)} \\ &= \sum_{j=2}^{n-1} \mathsf{Adv}^{\mathsf{mcrDDH}}(\mathcal{B}_j). && \text{(By Eqs. (5) and (6))}\end{aligned}$$

The result follows. □