Design and Analysis of Isogeny-based Static-Key Protocols

By

Andrea Basso

A thesis submitted to
the University of Birmingham
for the degree of
DOCTOR OF PHILOSOPHY

Centre for Cyber Security and Privacy
School of Computer Science
College of Engineering and Physical Sciences
University of Birmingham
January 2023

Abstract

The advent of quantum computers renders most cryptographic protocols obsolete and urges for a quick transition to post-quantum solutions. Many valid techniques for quantum-secure encryption and digital signatures have been proposed, but more advanced primitives do not yet have efficient post-quantum replacements. Among these are primitives where multiple parties have long-term secrets; we refer to these protocols as *static-key cryptography*. Few post-quantum static-key protocols have been reported in the literature, and nearly all are based on isogenies. In this work, we focus on two fundamental static-key primitives: non-interactive key exchanges (NIKE) and oblivious pseudorandom functions (OPRF). We first study the security of existing constructions with long-term static keys, and then we develop new protocols that outperform the existing ones.

In particular, we first assess the security of the isogeny-based NIKE based on k-SIDH proposed by Jao and Urbanik. Compared to the original k-SIDH, the protocol relies on non-trivial automorphisms to improve its efficiency. However, we show the protocol is vulnerable to an attack that exploits the additional automorphism structure. While the attack does not fully break the proposed parameters, it shows that the k-SIDH variant by Jao and Urbanik reduces the security level more so than it increases efficiency, making it less preferable over the standard k-SIDH.

We also analyze a validation method proposed by Fouotsa and Petit that checks the correctness of SIDH public keys. If the method were secure, it could be translated into an efficient NIKE. However, we demonstrate an efficient attack that allows a malicious party to provide dishonest public keys and satisfy the validation check. As part of this work,

we also discuss possible improvements to the countermeasures but show that they are similarly vulnerable to an extension of our attack. We then conclude with a discussion on the possibilities and the requirements of future validation techniques.

Beyond NIKEs, we analyze the security of the isogeny-based OPRF proposed by Boneh, Kogan, and Woo at Asiacrypt 2021. We introduce a polynomial-time attack on the one-more unpredictability property of the OPRF, which allows a malicious user to evaluate the PRF independently after some interactions. We show that a simple countermeasure can prevent the attack; we also propose a second attack that achieves the same goals but cannot be easily defended against. The second attack, however, has subexponential complexity. To demonstrate its feasibility, we develop a proof-of-concept implementation that can efficiently carry out the attack.

In follow-up work, we develop an efficient countermeasure against the previously introduced attack. We also integrate the countermeasures against the SIDH attacks into the OPRF protocol, and we propose a new proof of isogeny knowledge that can work with the countermeasures. Moreover, we introduce a novel zero-knowledge proof of parallel isogeny that provides non-interactive verifiability. By combining everything together, we obtain an OPRF protocol that is post-quantum secure, verifiable, round-optimal, and moderately compact.

Lastly, we study the problem of generating supersingular elliptic curves with unknown endomorphism ring. Such curves are often needed as starting parameters in many isogeny-based protocols, including the OPRF protocol we have developed. To generate curves of unknown endomorphism ring, we propose a distributed trusted-setup protocol that relies on a new statistical zero-knowledge proof; we prove its security in the general universally composable framework.

These works, both constructive and cryptanalytic, provide a better understanding of the limitations and the possibilities of isogeny-based cryptography in developing static-key protocols, and they are an important stepping stone towards fully practical post-quantum NIKEs and OPRFs.

Acknowledgments

When I started my PhD three or so years ago, I had a clear idea of how I expected my PhD experience to be: it turns out, it was nothing like I imagined. Nonetheless, and despite a global pandemic, it has been a fun, exciting, and rewarding experience, which would not have been possible without the help and support of many people.

First, I want to thank my supervisor Christophe Petit, for mentoring and encouraging me throughout, for his constant support, and for always making time.

I also want to thank my cosupervisor Sujoy Sinha Roy for his help and support, and for everything he has taught me. I learnt a lot from him.

My time as a PhD student benefitted from two wonderful internships. I want to thank Santosh Ghosh and Manoj Sastry at Intel Labs for mentoring and teaching me during my internship and for the continued collaboration over the years. I also want thank Navid Alamati and Visa Research for giving me the opportunity to learn so much and to work in person for three months. I also want to thank all the other interns at Visa Research, for some of the best months of my entire PhD.

Many thanks also go to all my coauthors throughout the years, from whom I learned a lot and with whom it has been a joy to work with.

I am also grateful to the Engineering and Physical Sciences Research Council and the University of Birmingham for funding my PhD project.

This journey would not have been the same without the people I shared it with. I am thankful to the people at the School of Computer Science, within the SRSCC, and in my research group, for the time spent together.

Thanks to all my friends, everywhere they are. Your help and support, the time spent together, the chats–deep and absurd alike–have gone a long way.

I want to deeply thank my family: my parents, my brother, my grandma. I would not have made it here without you.

And finally, thank you Jo, for *everything*.

Publications

This thesis is based on the following five papers:

1. Andrea Basso, Péter Kutas, Simon-Philipp Merz, Christophe Petit, and Charlotte Weitkämper. "On Adaptive Attacks Against Jao-Urbanik's Isogeny-Based Protocol". In: *AFRICACRYPT 20*. Ed. by Abderrahmane Nitaj and Amr M. Youssef. Vol. 12174. LNCS. Springer, Heidelberg, July 2020, pp. 195–213. DOI: 10.1007/978-3-030-51938-4_10 (see Chapter 2)

2. Andrea Basso, Tako Boris Fouotsa, Christophe Petit, and Charlotte Weitkämper. *Another Look at Adaptive Attacks on SIDH: Breaking HealSIDH*. Unpublished. 2022 (see Chapter 3)

3. Andrea Basso, Péter Kutas, Simon-Philipp Merz, Christophe Petit, and Antonio Sanso. "Cryptanalysis of an Oblivious PRF from Supersingular Isogenies". In: *ASIACRYPT 2021, Part I*. ed. by Mehdi Tibouchi and Huaxiong Wang. Vol. 13090. LNCS. Springer, Heidelberg, Dec. 2021, pp. 160–184. DOI: 10.1007/978-3-030-92062-3_6 (see Chapter 4)

4. Andrea Basso. *A Post-Quantum Round-Optimal Oblivious PRF from Isogenies*. Cryptology ePrint Archive, Paper 2023/225. 2023. URL: https://eprint.iacr.org/2023/225 (see Chapter 5)

5. 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: *EUROCRYPT 2023, Part II*. LNCS. Springer, Heidelberg, June 2023, pp. 405–437. DOI: 10.1007/978-3-031-30617-4_14 (see Chapter 6)

Throughout the PhD, I also coauthered the following publications, including peer-reviewed papers, preprints, technical reports, refereed posters, and unpublished papers.

Published papers:

6. Sujoy Sinha Roy and Andrea Basso. "High-speed Instruction-set Coprocessor for Lattice-based Key Encapsulation Mechanism: Saber in Hardware". In: *IACR TCHES* 2020.4 (2020). https://tches.iacr.org/index.php/TCHES/article/view/8690, pp. 443–466. ISSN: 2569-2925. DOI: 10.13154/tches.v2020.i4.443-466

7. Andrea Basso and Sujoy Sinha Roy. "Optimized Polynomial Multiplier Architectures for Post-Quantum KEM Saber". In: *2021 58th ACM/IEEE Design Automation Conference (DAC)*. 2021, pp. 1285–1290. DOI: 10.1109/DAC18074.2021.9586219

8. Malik Imran, Felipe Almeida, Jaan Raik, Andrea Basso, Sujoy Sinha Roy, and Samuel Pagliarini. "Design Space Exploration of SABER in 65nm ASIC". in: *Proceedings of the 5th Workshop on Attacks and Solutions in Hardware Security*. ASHES '21. Virtual Event, Republic of Korea: Association for Computing Machinery, 2021, pp. 85–90. ISBN: 9781450386623. DOI: 10.1145/3474376.3487278

9. Aikata Aikata, Andrea Basso, Gaetan Cassiers, Ahmet Can Mert, and Sujoy Sinha Roy. "Kavach: Lightweight masking techniques for polynomial arithmetic in lattice-based cryptography". In: *IACR Transactions on Cryptographic Hardware and Embedded Systems* 2023, Issue 3 (2023), pp. 366–390. DOI: 10.46586/tches.v2023.i3.366-390

10. Malik Imran, Felipe Almeida, Andrea Basso, Sujoy Sinha Roy, and Samuel Pagliarini. "High-speed SABER key encapsulation mechanism in 65nm CMOS". in: *Journal of Cryptographic Engineering* (Mar. 2023). ISSN: 2190-8516. DOI: 10.1007/s13389-023-00316-2

Preprints:

11. Andrea Basso, Furkan Aydin, Daniel Dinu, Joseph Friel, Avinash Varna, Manoj Sastry, and Santosh Ghosh. *Where Star Wars Meets Star Trek: SABER and Dilithium on the Same Polynomial Multiplier*. Cryptology ePrint Archive, Report 2021/1697. https://eprint.iacr.org/2021/1697. 2021

Technical reports and refereed posters:

12. Andrea Basso. "Poster: A Post-Quantum Oblivious PRF from Isogenies". In: ACM Press, 2022, pp. 3327–3329. DOI: 10.1145/3548606.3563542

13. Jan-Pieter D'Anvers, Angshuman Karmakar, Sujoy Sinha Roy, Frederik Vercauteren, Jose Maria Bermudo Mera, Michiel Van Beirendonck, and Andrea Basso. *SABER*. tech. rep. available at https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions. National Institute of Standards and Technology, 2020

Contents

Chapter 1

Introduction

Let the great plot commence.
— Mary Stuart, last letter to her coconsiprators

The history of cryptography is the history of a battle lasting centuries: on one side, cryptographers, trying to devise better methods to communicate securely; on the other, cryptanalysts, trying to extract secret information.

This long-raging war dates back to the birth of cryptography. For centuries, codemakers were in the lead: substitution ciphers, where one letter is replaced by another, were thought to be secure against anybody who did not know the mapping between letters. This changed around AD 800 when Al-Kindi, an Arab philosopher and mathematician, proposed a simple yet brilliant method to break substitution ciphers: he noted letters do not appear with the same frequency, and thus even if a letter is replaced by another, its frequency is unchanged and can reveal the original value. This breakthrough had significant consequences: a famous example is the sentencing of Mary Stuart, who plotted to overthrow Queen Elizabeth I of England in 1586. After the plot was discovered, she believed she was safe because she had encrypted her letters to the other conspirators with a substitution cipher. But a crown cryptanalyst had intercepted, decrypted, and modified Mary Stuart's communications. When she sent the letter that would commence the plot,

she signed her execution orders.

Over the centuries that followed, the competition continued with alternating fortunes for codemakers and codebreakers. In modern times, World War II has been a major battlefield between cryptographers on both sides of the war. The German military apparatus relied on Enigma, a machine to encrypt and decrypt information that was believed exceptionally secure. But the work of Polish mathematicians at the Polish Cipher Bureau and British researchers, including Alan Turing, at Bletchley Park eventually led to a breakthrough that allowed Allied operators to decrypt German secret information. Besides contributing to the Allied victory, the decryption of Enigma is estimated to have shortened the war and saved lives.

After the world war, the battle between cryptographers and cryptanalysts that had lasted millennia came to a sudden end, with the resounding and definitive win of cryptographers. This started with the advent of public-key cryptography in 1976. Before then, all encryption methods required two people to have already established a shared secret key, but such an approach could not work in an increasingly more connected world where people who have never interacted before needed to communicate securely. A new, radically different approach was developed by Whitfield Diffie, Martin Hellman, and Ralph Merkle. In 1974, Merkle developed the concept of public-key cryptography: a method that enabled any two people to establish a shared secret without the need for any previous interaction. While it was originally thought to be impossible, only two years later, Diffie and Hellman develop the first efficient public-key protocol [DH76], which is based on exponentiation in finite groups. Shortly afterwards, Ron Rivest, Adi Shamir, and Leonard Adleman developed a different protocol [RSA78], named RSA after its creators, that achieved the same goals.

Over the decades that followed, several other protocols were developed, but to this day the most used public-key algorithms are based on the Diffie-Hellman and RSA protocols. Their security has been thoroughly analyzed, and they are widely believed to be secure. Their security relies on two hard problems: the discrete logarithm problem for the

Diffie-Hellman protocol, and the integer factorization problem for RSA. The amount of computations needed to solve these problems is believed to be astronomical. Even the most powerful supercomputers, running for the entire lifespan of the universe, could not break these protocols. Nonetheless, a Damocles sword hangs onto them: in 1994, Peter Shor proposed a quantum algorithm [Sho94] that can efficiently solve both problems. The reason why these protocols are still widely used and believed secure is that a quantum algorithm requires a quantum computer to run. These are machines that exploit the principles of quantum mechanics to compute in a radically different way, which enables certain classes of computations to be carried out much faster. Prototypes of quantum computers exist, but they cannot run Shor's algorithm yet because they are still very limited in their computational power. They can be compared to the state of computers in the 1950s. Like those machines, however, quantum computers are expected to become more and more powerful. Many companies, universities, and governments are heavily investing in their development, and they might become powerful enough to break RSA and Diffie-Hellman soon. A recent survey of 46 experts [MP21] reports that the majority of them agree that quantum computers are more likely than not to break RSA-2048 in a day within the next 15 years.

Some alternatives that are resistant against attacks on quantum computers have been proposed, and they are known as post-quantum or quantum-secure. The transition to post-quantum solutions cannot wait until quantum computers become sufficiently powerful. Analyzing and developing quantum-resistant protocols requires several years, and several more are necessary to widely deploy such protocols. Moreover, encrypted communication can be stored for years; private information that is encrypted now can be decrypted later when quantum computers will be sufficiently powerful. This can be devastating for important data, such as military and diplomatic communications, even if they are revealed decades after their initial communication. Hence, it is critical to urgently transition to post-quantum solutions as soon as possible.

To spearhead such a transition, the American National Institute of Standards and

Technology (NIST) initiated a standardization process in 2016 aimed at identifying new solutions for post-quantum key-encapsulation methods (KEMs) and digital signatures. Initially, 69 submissions were received. These relied on different approaches, including constructions based on lattices, isogenies between supersingular curves, error-correcting codes, multivariate systems of equations, and hash functions. While some were broken within hours of publication, the other submissions underwent several years of scrutiny by the wider cryptographic community. The majority of the submissions were based on lattice-related problems, which are believed to be quantum resistant. Other protocols were based on codes, multivariate systems of equations, isogenies, and hash functions. NIST selected 26 candidates for round two of the competition, out of which only four KEMs and three digital signature protocols were chosen as finalists in round three. Among the KEM finalists, there was Saber, a key-encapsulation mechanism that we developed based on the learning with rounding problem.

In March 2022, NIST announced Kyber [SABDK+22] as the KEM protocol to be standardized, as well as Dilithium [LDKLS+22], Falcon [PFHKL+22], and SPHINCS+ [HBDEF+22] as the upcoming standards for digital signatures. Three of the four protocols (all except SPHINCS+) base their security on the hardness of lattice-related problems. The three protocols are very efficient and have a moderate bandwidth requirement, making them well-suited to be deployed in most scenarios.

1.1 — Static-Key Cryptography

With the standardization of Kyber, we can consider post-quantum encryption as a solved problem; several better protocols may be proposed in the future, but Kyber, as well as many other protocols, currently offers a valid alternative to classical encryption protocols. The same is also mostly true for digital signatures: while SPHINCS+ is designed for high-assurance scenarios, Dilithium and Falcon can replace their classical correspondents in most instances, although the larger signatures and the longer execution times pose an

issue in specific use cases, such as in the TLS protocol [SSW20]. However, the transition to post-quantum solutions becomes significantly harder when more advanced primitives are considered. In particular, primitives that rely on multiple parties having long-term static keys are often hard to implement in a quantum-secure manner. We refer to the set of such primitives as *static-key cryptography*. In this thesis, we mainly focus on two static-key primitives: non-interactive key exchanges and oblivious pseudorandom functions.

1.1.1 – Non-interactive key exchange

A non-interactive key exchange (NIKE) is a primitive where any two parties, having previously published their public key, can agree on a shared secret without any further interaction. First formalized by Freire, Hofheinz, Kiltz and Paterson [FHKP13], it is a useful construct that plays a role as a building block in more advanced protocols, but it also has direct applications.

NIKEs are widely employed to guarantee implicit and deniable authentication. If two parties communicate over a channel encrypted with a shared secret generated by a NIKE, they are implicitly authenticating each other because no other party can derive their shared secret. Nonetheless, this authentication is deniable: since the protocol is non-interactive, the authentication is derived exclusively from the shared secret, which is only known to the two people engaging in the protocol. To any external party, there is no guarantee that the people engaging in the protocol are who they claim to be. Deniable authentication is at the core of messaging protocols such as the X3DH layer [MP16] used in Signal and WhatsApp. Another common application of NIKEs is encryption in resource-constrained devices, such as microcontrollers and IoT appliances. For low-power devices, communication is often very costly; thus, reusing preshared public keys of a non-interactive key exchange can significantly reduce the bandwidth consumption.

In the classical setting, it is easy to build a NIKE since the original Diffie-Hellman construction [DH76] is already non-interactive. Given two public keys g^a and g^b, both parties can obtain the shared secret g^{ab} without any further interaction. This simple

construction is surprisingly hard to replicate in the post-quantum domain: NIKEs based on the learning-with-error (LWE) problem face significant challenges [GKRS22], and no other construction has been proposed based on other lattice-based or code-based problems. Some quantum-secure solutions based on isogenies have been proposed, but they are far from being an efficient replacement for Diffie-Hellman.

1.1.2 – Oblivious pseudorandom function

The second primitive on which we focus is oblivious pseudorandom functions (OPRF). Such a primitive allows a user and a server to jointly evaluate a pseudorandom function (PRF) on a user's input and a server's long-term key. The evaluation must be oblivious, which means the server should not learn anything about the client's input. If the OPRF protocol is *verifiable*, the server provides a proof that its computations were honest and used a previously-committed secret key during each execution of the protocol.

OPRFs are an important building block in many cryptographic applications, and a wide range of internet protocols depend on them. OPRFs can be used for private set intersection [JL09], which in turn has many applications ranging from privacy-preserving contact discovery for messaging services [DRRT18] to checking for compromised credentials [LPASC+19]. For instance, the web browser Microsoft Edge uses an OPRF-based protocol to detect compromised passwords [LKLM21]. A second practical use case of OPRFs is the privacy-preserving authorisation mechanism known as Privacy Pass [DRRT18]. Developed and currently deployed by Cloudflare, Privacy Pass reduces the number of CAPTCHAs that users need to solve by issuing a number of tokens, which users can later spend to avoid solving a second CAPTCHA. To prevent the server (i.e., Cloudflare, in this case) from tracking users across websites, the users' queries must be oblivious, and they are protected by an OPRF protocol. An OPRF is also a key component in OPAQUE [JKX18], a strong asymmetrical password-authenticated key exchange that enables a client to authenticate to a server via a password, without the need to communicate the password explicitly. The strong asymmetrical property guarantees the security of

the password even if the server is compromised; this is made possible by the evaluation of an OPRF during the authentication phase. Further applications of (verifiable) OPRFs include password-management systems [ECSJR15], adaptive oblivious transfer [JL09], and password-protected secret sharing [JKK14].

It is possible to build an OPRF from generic two-party computation techniques, but the resulting protocols are inefficient and highly interactive since actively secure two-party computations require at least five rounds of interaction [KO04]. A simple and highly-efficient OPRF can be obtained by obliviously evaluating a PRF based on the Diffie-Hellman assumption. The PRF consists of mapping an input x to a group element via a hash function $H : \mathcal{X} \to \mathbb{G}$ and exponentiating by the secret key, i.e. $F(k, x) = H(x)^k$. This evaluation can be made oblivious by introducing a blinding exponent b: the client sends $H(x)^b$ to the server, which returns $g = H(x)^{bk}$, and the client can compute the output $g^{-b} = H(x)^k$. Several other Diffie-Hellman-based OPRFs are reported in the literature; for a comprehensive analysis, we refer to a survey by Casacuberta, Hesse, and Lehmann [CHL22].

In the post-quantum setting, very few solutions have been proposed. Albrecht, Davidson, Deo, and Smart [ADDS21] introduced a verifiable and round-optimal OPRF based on lattices, but the result is mainly a demonstration of feasibility: the protocol requires the client and the server to exchange more than 128 GB for each execution of the OPRF. Boneh, Kogan, and Woo [BKW20] proposed two OPRF protocols based on isogenies. The first is verifiable but highly interactive, and we proposed two attacks on the one-more unpredictability property of the protocol: after a number of interactions, a malicious user can extract enough information to evaluate the PRF independently [BKMPS21, or Chapter 4]. The second construction was unaffected by the attack, but the protocol is not verifiable and requires three rounds of interaction.

1.1.3 – Commonalities

Both non-interactive key exchanges and oblivious pseudorandom functions lack good post-quantum constructions. This is not accidental since both primitives rely on long-term static keys. This is also the case for generic encryption algorithms—such as those standardized by NIST—which allow one party to have a static key. However, those algorithms protect against adaptive attacks by having the other party reveal their ephemeral key. This cannot be done either in NIKEs, where both parties are using static keys, or in OPRFs, where the user's input needs to remain oblivious.

But the connection between the two primitives is deeper: OPRFs require a client to blind their input so that the server cannot distinguish it. The output of the OPRF needs to be deterministic and independent of the blinding: this means that the server needs to output some value that allows the client to undo the effects of the blinding. The blinding, acting on blinded data, and unblinding strategy is reminiscent of non-interactive key exchanges, where one party computes their public key (blinding), the second party completes the exchange (acting on blinded data), and the first party can generally undo their secret to obtain the other party's public key (unblinding). Thus, to construct either a NIKE or an OPRF, it is necessary to have a mechanism that enables such computations, and it can do so in the presence of long-term secret keys.

The relationship between the two primitives is further evidenced by the few post-quantum constructions that have been reported. With only one exception, all constructions of post-quantum NIKEs and OPRFs are based on isogenies and share similar techniques. Indeed, isogeny-based cryptography offers techniques to securely work with static keys, and it appears to be the only solution to develop post-quantum NIKEs and OPRFs.

1.2 — Isogeny-based cryptography

Elliptic curves played a fundamental role in the history of cryptography: many common protocols rely on elliptic curves, and they enabled the development of powerful pairing-

based tools. Classical protocols generally consider only operations on a single elliptic curve: the points on a curve form a commutative group, and scalar multiplication is hard to invert on a classical computers. The problem is however easy on a quantum computer due to Shor's algorithm [Sho94], which makes operations on an elliptic curve unsuitable for quantum-secure protocols. However, elliptic curves can still play a role in post-quantum cryptography. Isogenies are maps between elliptic curves, and given two curves, finding a connecting isogeny is believed to be hard even for quantum computers. Relying on the hardness of this problem, it is possible to develop powerful protocols.

The first isogeny-based protocol dates back to 1996; it is due to Couveignes, who only made it public in 2006 [Cou06]. Couveignes proposed a framework called Hard Homogenous Space (HHS), which consists of a set with a free and transitive group action that satisfies certain properties. These properties ensure that it is possible to work with group elements and to efficiently compute the group action, while also guaranteeing the hardness of the inverse operation. An HHS leads to a key exchange protocol, as well as an identification protocol. The HHS framework captures the concept of discrete logarithms, but Couveignes also proposed an instantiation based on isogenies between ordinary elliptic curves. The fundamental assumption it relied on is that, given two elliptic curves, it is hard to find an isogeny between them: this is often referred to as the *pure isogeny problem*.

Independently, Rostovtsev and Stolbunov [RS06] proposed a public-key encryption scheme based on the same group action. Stolbunov [Sto10] further developed the concept by proposing an interactive key agreement protocol. The resulting protocols were impractical and inefficient, but they were an early attempt at developing post-quantum solutions. The claims of quantum resistance prompted Childs, Jao, and Soukharev to adapt Kuperberg's algorithm [Kup05] to obtain a quantum sub-exponential attack on the protocol [CJS14]. More recently, De Feo, Kieffer, and Smith [DKS18] proposed an improved version of the Couveignes-Rostovtsev-Stolbunov protocol based on an optimised choice of parameters that reduces the computation costs. The improvements significantly

reduced the latency of the protocol, which still remains impractical for most use cases.

The same year that Rostovtsev and Stolbunov proposed an isogeny-based encryption protocol, Charles, Goren, and Lauter proposed a provably-secure hash function based on expander graphs [CLG09] with an instantiation based on supersingular elliptic curves. This is the first isogeny-based protocol to work with supersingular curves. Indeed, the set of supersingular elliptic curves forms a connected graph, where the nodes are supersingular curves (up to isomorphism) and the edges are the isogenies between them. This graph is Ramanujan, which means it has good expanding properties: the distribution of ending curves after a short random walk in the graph quickly converges to the stationary distribution. This property is useful for cryptographic applications, and hash functions in particular; the CGL hash function maps an input message to a walk in the path and outputs the ending curve. The expanding property of the supersingular graph is used to prove the function is hard to invert.

Subsequent work [PL17; EHLMP18] showed the hash function can be backdoored: the function is not collision resistance if the endomorphism ring of the starting curve is known; thus, particular care is needed when picking the starting parameters. The implication, however, goes both ways: if the endomorphism ring of the starting curve is unknown, the CGL hash function can be shown to be collision-resistant, under the assumption that computing endomorphism rings of a given curve is a hard problem. Besides the direct applications to the CGL hash function, its security analysis showed the importance and the risks associated with curves of known endomorphism ring in isogeny-based protocols.

1.2.1 – The SIDH key exchange

The modern era of isogeny-based cryptography started in 2011, when Jao and De Feo proposed SIDH, or Supersingular Isogeny Diffie Hellman [JD11]. The protocol relied on isogenies between supersingular elliptic curves: this change allowed better control of the number of rational points on the elliptic curve. Moreover, supersingular curves avoid the

subexponential attack on isogenies between ordinary elliptic curves [CJS14] by having a non-commutative endomorphism ring, which allows using smaller parameters. However, since the endomorphism ring is non-commutative, isogenies between supersingular curves do not give rise to a group action: the key contribution of Jao and De Feo was a method to overcome this by revealing torsion point images.

The SIDH protocol, depicted in Fig. 1.1, relies on a prime p of the form $p = ABf - 1$, where A and B are smooth numbers and f a small cofactor. The main parameter is a supersingular elliptic curve E_0 defined over $\mathbb{F}_{p^2}$. One party computes the isogeny $\phi_A : E_0 \to E_A$ of degree A and reveals the curve E_A together with the torsion images $\phi_A(P_B), \phi_A(Q_B)$, where P_B, Q_B form a basis of $E_0[B]$. The other party proceeds similarly by computing an isogeny $\phi_B : E_0 \to E_B$ and revealing the curve E_B, together with torsion images. Then, the first party can compute the pushforward $\phi'_A = [\phi_A]_*\phi_B$ by expressing any generator K of the kernel of ϕ_B as $[x]P_B + [y]Q_B$ and computing the isogeny with kernel $\langle [x]\phi_A(P_B) + [y]\phi_A(Q_B)\rangle$. The second party can similarly compute the pushforward $\phi'_B[\phi_B]_*\phi_A$. The codomain of both pushforwards is the same curve E_{AB} [Leo20], which then forms the shared secret.

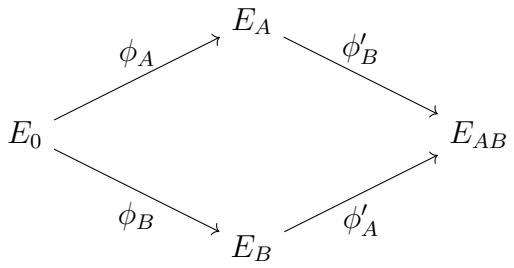

Figure 1.1: The SIDH key exchange.

From a security perspective, SIDH differs in two fundamental ways from the more generic setting of the CGL hash function: all the isogenies have a fixed and short degree[1], and their action on a torsion basis of coprime order is revealed. For a long time, these changes were thought not to affect the security of the protocol, although they did have some limited implications.

[1]The degree of both ϕ_A and ϕ_B is $\approx \sqrt{p}$; however, given two random curves, the shortest isogeny between them is expected to have degree $\approx p$.

In 2016, Galbraith, Petit, Shani, and Ti [GPST16] showed that the shortness of the isogenies made it easy to recover them if the endomorphism ring of E_0 and E_A are known. This did not lead to an attack since the endomorphism ring of a curve is generally hard to compute. In the same work, the authors also proposed an adaptive attack against long-term static keys: by revealing maliciously-generated torsion points and observing whether the key exchange still succeeds, it is possible to recover one bit of the other party's secret key. Repeating the procedure multiple times leads to an efficient key-recovery attack. A second, albeit less efficient, adaptive attack that also exploits maliciously generated public keys was later proposed by Fouotsa and Petit [FP22]. A different passive attack that exploits the presence of torsion point images was proposed by Petit [Pet17], but it only applied to variants of SIDH with unbalanced parameters, i.e. the degree of ϕ_A is much larger than that of ϕ_B (or vice versa). This line of attacks was later improved [QKLMP+21] to reduce the imbalance between parameters and to obtain better asymptotic quantum attacks, but they did not affect the security of SIDH with balanced parameters. These attacks showed that the torsion point images made isogeny recovery easier, at least in some circumstances.

Since the GPST adaptive attack showed that SIDH could only be used with ephemeral keys, SIKE (supersingular isogeny key-encapsulation) was proposed [JACCD+17]. SIKE is a version of SIDH with the Fujisaki-Okamoto transform [FO99; FO13], which allows one party to have a long-term secret key at the cost of repeating the SIDH computations twice. This is similar to most other post-quantum key exchanges, where static-static key exchanges are hard to achieve. However, unlike other protocols, the issue is not inherent to the protocol itself, but it is the result of the possibility of revealing maliciously computed torsion points. Thus, if it were possible to validate public keys, SIDH could have been used to build an efficient post-quantum authenticated key exchange (AKE) and a non-interactive key exchange (NIKE), as discussed by Galbraith [Gal18].

Validating public keys without any additional information, i.e. distinguishing whether two points are the correct images under a secret isogeny, was considered a hard problem,

and it had been conjectured to be as hard as breaking the SIDH protocol itself [UJ18]. To avoid adaptive attacks, the two parties can provide a zero-knowledge proof that the public keys are honest, but such proofs are large and inefficient. An alternative approach was proposed in k-SIDH [AJL17], which consisted of running k^2 instances of SIDH in parallel. However, the protocol was also very inefficient, as it required a large k since Dobson et al. [DGLTZ20] proposed a polynomial-time attack on k-SIDH for small values of k. Urbanik and Jao [UJ20] attempted to improve the efficiency of k-SIDH by exploiting the non-trivial automorphisms of special curves to reduce the number of computations while maintaining the same security level. However, we later showed that the introduction of automorphisms does reduce security [BKMPW20, or Chapter 2] more so than it improves efficiency: thus, in most situations, k-SIDH was preferable over the Jao-Urbanik variant.

A different approach was introduced by Fouotsa and Petit [FP21], who proposed a validation technique to be used in interactive key exchange with long-term secrets based on SIDH. In the protocol, each party reveals additional torsion images on the shared curve, which helps the other party validate the correctness of the torsion point. The same validation process could be used to build a non-interactive key exchange. Subsequent work [GL22] showed the protocol is still vulnerable to more sophisticated adaptive attacks. In parallel and independent work, we also obtained similar results by proposing a different attack (see Chapter 3).

Overall, the relative efficiency of SIDH, its small public keys, and its partial support for non-interactive modes, when paired with an appropriate zero-knowledge proof, made SIDH the most practical and well-known isogeny-based protocol. Several protocols have been built on top of it. These included proofs of knowledge [JD11; DDGZ22; BCCDF+23] and digital signatures [YAJJS17; CMP22], some of which with more advanced functionalities [SC18; SGP19]. These signature protocols lacked in efficiency and compactness, however: the underlying sigma protocol they relied on had a soundness of $1/2$, and thus it needed to be repeated a high number of times to achieve sufficient security. On the

encryption side, Eaton et al. [EJKM22] proposed a key-updatable public-key encryption protocol based on SIDH, while Dobson and Galbraith [DG22] introduced SI-X3DH, an SIDH-based version of the Signal authentication protocol [MP16]. Since SIDH supported non-interactive key exchanges, when public keys are authenticated, the resulting protocol was far more efficient than the alternatives based on other post-quantum primitives.

1.2.2 – The SIDH attacks

The landscape of isogeny-based protocols radically changed when Castryck and Decru [CD23], Maino and Martindale [MMPPW23] (later joined by Panny, Pope, and Wesolowski), and Robert [Rob23] proposed a series of attacks on SIDH that completely broke the security of the protocol. Not all isogeny-based protocols are affected: the most common among these is CSIDH [CLMPR18], a key exchange protocol based on a commutative group action. It can be interpreted as the translation of the protocols by Couveignes [Cou06] and Rostovtsev and Stolbunov [RS06] to the supersingular curve setting, and as such is vulnerable to the subexponential attack by Childs, Jao, and Soukharev. While identifying the correct parameters that provide sufficient security may be hard [BS20; Pei20; CCJR22], the protocol is compact and very flexible: since it relies on a group action, it enables a simple translation of most protocols based discrete logarithms (that do not require group multiplication, only exponentiation) to the post-quantum setting. Another prominent isogeny-based protocol is SQISign, a signature scheme that proves knowledge of endomorphism rings. The protocol is very compact, resulting in the smallest known post-quantum signature while offering nearly practical signing and verification speeds. These protocols, together with newer ones [Ler22], remain unaffected by the SIDH attacks and bring useful features for the post-quantum transition.

The three attacks on SIDH follow a similar approach and are all based on isogenies between varieties in genus two. A generalization of elliptic curves to higher dimensions is a principally polarized abelian variety, which is a variety equipped with an isomorphism

from the variety to its dual called a polarization. Principally polarized abelian surfaces (PPAS) can be divided into two categories: those that are the product of two elliptic curves, and those that are the Jacobian of a superspecial genus-2 curve. These latter ones are exponentially more common: over $\mathbb{F}_{p^2}$, there are $p^3/2880$ such curves, compared to the $p^2/288$ that arise as the product of two elliptic curves. Thus, a random walk in a PPAS isogeny-graph is expected to end on a Jacobian with overwhelming probability. However, this is not the case for a specific isogeny whose kernel is related to the image points revealed in SIDH, as shown by Kani's theorem [Kan97]; this property lies at the core of the SIDH attacks.

Let us consider the case where an attacker aims at recovering a secret A-isogeny $\phi : E_0 \to E_1$, given the curves E_0, E_1, the basis P_0, Q_0 of $E_0[B]$ for some B coprime with A, and its image $P_1 = \phi(P_0), Q_1 = \phi(Q_0)$. Assume that an attacker can also evaluate an isogeny $\gamma : E_0 \to E_2$ of degree $c = B - A$, which means they can build the following SIDH square:

$$\begin{array}{ccc} E_0 & \xrightarrow{\phi} & E_1 \\ {\scriptstyle\gamma}\downarrow & & \downarrow{\scriptstyle\gamma'} \\ E_2 & \xrightarrow[\phi']{} & E_3 \end{array}$$

Kani's theorem implies that there exists a dimension-two isogeny $\rho : E_2 \times E_1 \to E_0 \times E_3$ given by

$$\rho(X, Y) = (\hat{\gamma}(X) + \hat{\phi}(Y), \phi'(X) - \gamma'(Y)),$$

whose kernel is $\langle(\gamma(P_0), \phi(P_0)), (\gamma(P_0), \phi(Q_1))\rangle$. This means that an attacker can compute the isogeny ρ from the torsion images and the evaluations of γ. However, the isogeny ρ contains the dual of the secret isogeny $\hat{\phi}$ in its first component: thus, computing $\rho(\mathcal{O}_{E_2}, \cdot)$, where $\mathcal{O}_{E_2}$ is the neutral point on E_2, allows the attacker to evaluate $\hat{\phi}$ on any point on E_1. This includes points of order A, from whose image it is possible to recover the kernel of ϕ and $\hat{\phi}$. Alternatively, Kani's theorem can also be exploited to obtain a decisional

oracle by checking whether the codomain of the isogeny ρ is a product of elliptic curves or a Jacobian. Such a method similarly leads to a full key-recovery due to the search-to-decision reduction between the Decisional and the Computational SIDH problem by Galbraith and Vercauteren [GV18]. The attacker can guess one step of the isogeny ϕ, use Kani's theorem to check if the guess was correct, and repeat the procedure until all the steps in ϕ are recovered.

Castryck and Decru, and Maino and Martindale independently discovered this attack. The two works differ in their key-recovery technique and their approaches in computing the isogeny γ of degree $B - A$. The former work relies on the decisional oracle to recover the secret key *bit by bit*, while the latter attack uses the dual isogeny computations to directly recover a kernel generator. Moreover, Castryck and Decru showed that if the endomorphism ring of the starting curve is known, the isogeny γ can be an endomorphism of the correct degree, and it can be efficiently evaluated on the 2^a-torsion since it is represented as a combination of basis endomorphisms. Maino and Martindale, instead, analyzed the complexity of evaluating an isogeny of degree c, based on the distribution of its smooth factors, and derived a subexponential attack that works with curves of unknown endomorphism ring. This suggested that it may be possible to avoid these attacks by increasing the parameters and by using a starting curve of unknown endomorphism. This may be generated by one party, in the case of SIKE, where the other party already reveals their secret isogenies, or through a trusted setup protocol [BCCDF+23, or Chapter 6]. However, Robert [Rob22] showed that evaluating any isogeny, even of prime degree, can be computed in polynomial time by embedding it into a smooth isogeny of higher dimensions. When this method is applied to the SIDH attacks [Rob23], it gives a polynomial-time algorithm to compute the isogeny γ in all cases; this leads to a polynomial-time attack on SIDH with any starting curve.

1.2.3 – Bringing SIDH back

The SIDH attacks fully break SIDH, the most efficient and well-known isogeny-based protocol, and one of the few NIST round four candidates. The attack easily extends to the wide range of protocols based on SIDH and, more generally, to any protocol that reveals torsion images of sufficiently large order. A natural question is whether it is possible to modify the SIDH protocol to incorporate some countermeasures that prevent the SIDH attacks.

A first approach may reduce the amount of torsion information that is revealed. The initial attacks required the degree of the secret isogeny to be comparable to the order of the torsion images revealed, but Robert [Rob23] extended the attack to recover an isogeny of degree d when the torsion points have order $\sqrt{d}$. This is, in some sense, optimal: more than one isogeny of degree d may map a torsion basis to the same basis of order $\ll \sqrt{d}$ [MP19]. This suggests that revealing torsion images of small order preserves the secrecy of the isogeny. However, an attacker can brute-force part of the secret isogeny: if the degree of the isogeny is d, the torsion points need to have order smaller than $\sqrt{d/2^\lambda}$ to achieve a security of λ bits. While this approach appears to be secure[2], it can only be used in some asymmetrical constructions, but not in SIDH. Since the order of the torsion points corresponds to the degree of the other party's isogeny, this method may be used to protect one party's isogeny but not both. A more sophisticated approach is thus needed.

Recently, Fouotsa, Moriya, and Petit [FMP23b] proposed two countermeasures. The first relies on hiding the degree of the secret isogenies: since the attacks rely on the degree of the secret isogeny to be known, the change to a variable-degree SIDH prevents the attacks. However, the space of possible degrees needs to be exponentially large, which means the isogeny degrees need to be the product of several prime powers, rather than a single prime power as in the original SIDH protocol. Moreover, the torsion point images reveal some information about the isogeny degree. To avoid this, the torsion images need

[2]Previous attacks similarly do not apply. Even if the endomorphism ring of the starting curve is known, the secret isogeny is sufficiently long that the torsion point attacks proposed in [QKLMP+21] do not apply.

to be scaled by a value coprime with their order; this does not affect the correctness of the protocol since scaling kernel generators produces another generator of the same kernel, but it hides the degree of the isogeny. Nonetheless, since both points need to be scaled by the same value, the pairing between the torsion images still reveals the quadratic residuosity of the degree. Hence, the protocol requires a 13810-bit prime to provide 128 bits of security. This is significantly less efficient and less compact than the original SIDH, making it impractical for most applications. Moreover, since the degree of the isogeny is secret, it appears to be hard to translate the existing proofs of public key correctness to work with a hidden-degree SIDH, because these proofs all rely on revealing a parallel isogeny of the same degree as the secret one.

Fouotsa, Moriya, and Petit also propose a second countermeasure based on a similar technique: masking the torsion points by the same scalar appears to be sufficient to protect against the attacks. As in the previous case, however, the attacker can extract the square of the scaling value through the pairing of the torsion images. Hence, for the protocol to be secure, it requires that the scaling values have exponentially-many square roots, which implies that $p + 1$ is the product of at least 2λ distinct primes, where λ is the security parameter. To protect against a guessing attack, this countermeasure uses a 5911-bit prime for 128 bits of security. While this is also impractical for most applications, the fixed-degree nature of the protocol makes it better suited to work with proofs of public key correctness. The first such proof that works with these countermeasures is presented in Chapter 5.

1.3 — Objectives

In this thesis, we study how isogeny-based techniques can lead to post-quantum alternatives to non-interactive key exchanges and oblivious pseudorandom functions.

For both primitives, CSIDH offers some solutions: the CSIDH key exchange is inherently non-interactive, and Boneh, Kogan, and Woo [BKW20] introduced a CSIDH-based

OPRF. However, in both cases, there are significant limitations. The quantum attacks on CSIDH make it harder to estimate the requirements of CSIDH-based protocols, and they suggest the running times of a single CSIDH exchange might be beyond practical. Furthermore, the CSIDH group action is generally well-understood, which means there is little room for improvement for a CSIDH-based NIKE. On the OPRF front, the CSIDH protocol partially relies on generic two-party computations, which make it inherently interactive and not verifiable.

For these reasons, SIDH-based protocols appeared to be more promising, especially before the SIDH attacks. Several constructions have been considered for an SIDH-based NIKE [Gal18; AJL17; UJ20; FP21], while the OPRF protocol based on SIDH [BKW20] is verifiable or—alternatively—non-interactive in its non-verifiable form.

In this thesis, we thus focus on SIDH-based solutions. In the first part, we analyze the existing SIDH-based constructions to assess their security. In particular, we show the non-interactive key exchange protocol proposed by Urbanik and Jao [UJ20] is less secure than originally thought; we also propose an efficient attack against the validation method for SIDH public key proposed by Fouotsa and Petit [FP21], which—if secure—could have been used to build an efficient NIKE; lastly, we develop two attacks against the one-more unpredictability property of the OPRF proposed by Boneh, Kogan, and Woo [BKW20]. Overall, the analysis develops our understanding of the possibilities and the limitations of SIDH-based solutions to develop static-key protocols.

In the second part, we build upon the previous results to propose new protocols. In particular, we introduce an OPRF protocol that is post-quantum secure, verifiable, round-optimal, and moderately compact. The construction is based on the protocol by Boneh, Kogan, and Woo [BKW20], but it incorporates an efficient countermeasure against the one-more unpredictability attack, a proof of knowledge that works with the SIDH countermeasures, and a novel proof of parallelness to obtain round-optimal verifiability. We also propose a trusted setup protocol that enables participants to obtain a supersingular elliptic curve whose endomorphism ring is unknown. Such a curve is

necessary as a starting parameter for many other protocols, including the OPRF protocol we are proposing. The new OPRF construction, together with the trusted setup protocol to pick its starting curve, is an important stepping stone towards practical post-quantum static-key protocols.

1.4 — Analysis of existing protocols

1.4.1 – The Jao-Urbanik k-SIDH variant

The first protocol we analyze is the k-SIDH variant proposed by Urbanik and Jao [UJ20]. The original k-SIDH protocol [AJL17] is a modified version of SIDH that avoids adaptive attacks by relying on multiple curves. Each party relies on k distinct secret isogenies (hence the name), which lead to k^2 SIDH exchanges. The shared secret is then a hash of all the k^2 shared secrets. This protocol protects against adaptive attacks because they rely on modifying the torsion points based on the partially recovered secret key: since multiple secret keys are present, it seems hard to modify torsion points to work with all the k partial keys. One may be tempted to think that $k = 2$ is sufficient to guarantee security, but Dobson, Galbraith, LeGrow, Ti, and Zobernig [DGLTZ20] showed an attack on k-SIDH that is exponential in k. For sufficiently large values of k (≈ 90), the protocol is secure against adaptive attacks, but its efficiency is heavily impacted by running k^2 SIDH instances. Urbanik and Jao [UJ20] proposed a more efficient variant of k-SIDH by considering starting curves with special automorphism. Due to such automorphism, the same public key can be interpreted as three distinct ones: this means that both parties need to compute and exchange only a third of the public keys.

In Chapter 2, we propose a new attack on the Jao-Urbanik variant that exploits the presence of the automorphisms. In particular, the attack by Dobson, Galbraith, LeGrow, Ti, and Zobernig [DGLTZ20] requires the attacker to interact with the target with modified torsion points and guess the possible shared secret recomputed by the target. We show that the shared secrets that correspond to the same public key are correlated, which

allows the attacker to reduce the guesses to about a third. In other words, the structure that enables a more efficient protocol also enables a more efficient attack. This attack does not break the protocol for the proposed parameters, but it significantly reduces its security. When comparing the original k-SIDH protocol with its Jao-Urbanik variant re-parametrized to guarantee the same security level, we see that the Jao-Urbanik scheme offers moderately smaller ($\approx 30\%$) public keys at the cost of twice as many computations. Hence, in most instances, the original k-SIDH construction offers better trade-offs.

1.4.2 – The HealSIDH validation method

The second protocol we analyze is a novel validation technique proposed by Fouotsa and Petit [FP21]. The technique ensures the correctness of an SIDH public key by revealing additional torsion images on the shared secret curve. The authors propose an interactive static-static key exchange called HealSIDH: in other words, the two parties have long-term public keys, but the validation requires them to interact. This was a first step toward an efficient SIDH-based non-interactive key exchange. Indeed, we demonstrated that it is possible to obtain a NIKE based on HealSIDH by considering the data used for validation as part of the secret key. While proving the security of the resulting protocol, we identified an efficient attack that equally applied to the original construction. Chapter 3 presents the attack, together with a more general discussion of countermeasures and corresponding attacks. Note that Galbraith and Lai [GL22] developed a different attack on the same protocol in parallel and independent work.

In Chapter 3, we first propose a matrix-based notation that makes it easier to describe and analyze the HealSIDH validation method. To showcase the utility of the new notation, we apply it to the GPST attack [GPST16], and we show that it is possible to generalize it to a wide range of query points. Then, relying on the proposed notation, we study the specific requirements imposed by the validation technique. We demonstrate that the protocol considers many pairs of public keys and validation data to be correct, even if they are not honestly generated. By crafting specific interactions, we propose an adaptive

attack on the validation method, HealSIDH, and the related NIKE. Since the validation check relies on the private key, our attack leads to a full key recovery. Interestingly, our attack relies on honestly-generated public keys and malicious validation data; this showcases the risks associated with additional validation checks and their potential for further attacks.

Then, we propose an alternative validation method based on a similar strategy as HealSIDH: unlike HealSIDH, however, the proposed method introduced an additional dual isogeny computation and reveals torsion points on the public key curve of the other party. This strategy avoids both the proposed attack as well as the attack by Galbraith and Lai; nonetheless, it is still vulnerable to the more sophisticated attack that we present. This attack is more involved and requires several more interactions with the target to recover the secret key. Several experiments show that the number of interactions remains polynomial in the security parameter, making the new validation method also insecure. Lastly, we introduce dimension-two variants of the proposed protocols. We formalize dimension-two HealSIDH, and we discuss the difficulties in adapting the proposed attack to the dimension-two setting. We then conclude with a perspective on validation methods for SIDH and the issues that would need to be solved for an efficient solution to be developed.

1.4.3 – Cryptanalysis of the OPRF protocol by Boneh, Kogan, and Woo

We then focus on the SIDH-based verifiable OPRF protocol proposed by Boneh, Kogan, and Woo [BKW20] at Asiacrypt'20. To evaluate the PRF, the user maps the input m to an isogeny $\phi_m : E_0 \to E_m$. Then, the user blinds the curve E_m by computing a second isogeny $\phi_b : E_m \to E_{mb}$. The user sends the curve E_{mb} with the relevant torsion images to the server, which computes the isogeny $\phi_k : E_{mb} \to E_{mbk}$ based on its secret key and replies with the curve E_{mbk}, together with some torsion information. The user finally uses the provided torsion information to undo the blinding isogeny, i.e. to compute the

translation of the dual of the blinding isogeny, to obtain the curve E_{mk}, which is hashed together with the input and the public information to form the PRF output. To prevent adaptive attacks, the user provides a non-interactive zero-knowledge proof that torsion information was honestly computed; similarly, the server provides an analogous proof to guarantee the correctness of its computations. Furthermore, the server also engages in an interactive protocol with the client to demonstrate the computations have used a previously committed key.

In Chapter 4, we propose two attacks against the one-more unpredictability of the by OPRF Boneh, Kogan, and Woo [BKW20]. In the first attack, a malicious user engages in the OPRF with a message isogeny ϕ_m with kernel generator a point M, of order ℓ^e. The attacker repeats the process with the isogenies along the same path, but with shorter and shorter degrees. Thus, the output curves form an isogeny path parallel to the message isogeny. The attacker is then able to compute a generator of the kernel of such isogeny. By repeating this process three times with linearly independent points, the attacker obtains enough information to generate a torsion basis that allows them to evaluate the PRF on *any* input of their choice, without further interacting with the server. The attack is polynomial time, but it crucially relies on using message isogenies ϕ_m of varying degrees. The attack can be thwarted by the server checking the order of the isogeny ϕ_m, which is possible because of the proof of knowledge provided by the user. We also propose a second attack that cannot be easily prevented. The attack proceeds in a similar way to the previous one, but the malicious user uses only isogenies of full degree. To obtain the curves on the path of ϕ_m, the attacker needs to find the middle curve between two PRF outputs. This introduces a trade-off between the complexity of the attack and the number of queries. Minimizing both yields a subexponential attack on the one-more unpredictability of the protocol. To demonstrate the practicality of the attack, we also propose a proof-of-concept implementation of the attack that can break 64 bits of security in a weekend. We also estimate the running times of the attack for higher security levels, which demonstrates its feasibility.

1.5 — New protocols

1.5.1 – A new OPRF protocol

In Chapter 5, we build upon the results of Chapter 4 to derive a novel OPRF protocol. The high-level structure of the protocol is similar to that of the SIDH-based OPRF proposed by Boneh, Kogan, and Woo [BKW20], but we propose three significant changes that make the protocol secure and more efficient.

First, we propose an efficient countermeasure against the attacks proposed in Chapter 4. The main difficulty in avoiding the attacks is due to the ability of a malicious user to always recompute some secret information. Our proposed countermeasure avoids the attacks by changing how the user's input is mapped to an isogeny, which does not prevent the attacker from recovering the secret information, but it makes that information insufficient to independently evaluate the PRF. Moreover, the proposed changes result in a smaller prime and a more efficient protocol.

Secondly, the original SIDH-based construction is vulnerable to the SIDH attacks. We discuss several possible countermeasures in the context of the OPRF protocol, and we choose masked SIDH as the most suitable. The change requires longer isogenies and a larger prime, which makes the protocol less efficient, but a series of widespread optimizations allow us to maintain a moderate communication cost. Furthermore, to integrate masked SIDH within the OPRF, we also propose the first zero-knowledge proof of knowledge that can guarantee the correctness of a masked SIDH public key. The protocol proves the correctness of masked torsion images by building an SIDH square, as in previous proofs, and revealing masked torsion points on each side of the SIDH square so that the composition of the masking values along the SIDH square gives the secret masking value.

Lastly, we develop the first proof of knowledge that guarantees parallelness of two isogenies. In construction by Boneh, Kogan, and Woo, the server engages with the user

in a costly and highly interactive protocol to prove verifiability. With the proposed proof of knowledge, the proof of verifiability is non-interactive, which reduces the rounds of communications to the theoretically-optimal two. The proof is obtained by evaluating two proofs of isogeny knowledge in parallel and with correlated randomness; this means that verifiability can be proved almost *for free*, without a dedicated proof. Hence, we obtain a verifiability proof that is more compact than the original one and, most importantly, non-interactive.

Putting everything together, we obtain an OPRF protocol that is post-quantum secure, verifiable, round-optimal, and moderately compact, with a security proof in the UC framework [Can01] in the random-oracle model.

1.5.2 – Generating curves with unknown endomorphism

The OPRF protocol we developed requires to start from a Supersingular Elliptic Curve of Unknown Endomorphism Ring (SECUER), because the knowledge of its endomorphism ring can make certain computations easy. This is not uncommon, and several other protocols require such a curve, such as the CGL hash function [CLG09], dual-mode PKE [ADMP20], oblivious transfer [LGD21], and commitment schemes [Ste22]. However, it is currently unknown how to generate a curve such that nobody knows its endomorphism ring. Several techniques have been studied to sample a SECUER, or in other words to hash onto the supersingular graph, but no method has been found so far [BBDFG+22; MMP22].

The problem can be avoided using a *trusted setup*: a trusted party can generate a random curve by taking a random walk over the isogeny graph and discarding the information about the path. This, however, requires trusting that the party is indeed honest. Alternatively, this trust can be split among several participants: each one computes a segment of the random walk so that nobody entirely knows the path.

In Chapter 6, we formalize this process to obtain a trusted setup protocol that produces a SECUER. First, we propose the first statistically zero-knowledge proof of isogeny

knowledge. To do so, we study the supersingular isogeny graph *with level structure*, i.e. where each node is a curve with an associated subgroup. We prove the graph is Ramanujan: that means that if you fix a curve E with a subgroup G and you consider a sufficiently long isogeny $\phi : E \to E'$, the output curve E' and the subgroup image $\phi(G)$ are statistically close to uniformly random[3]. Building on this result, we prove the statistical zero-knowledge property of the proposed proof.

Second, we analyze the trusted setup protocol with n parties: each individual computes a random walk starting from the previous end curve, and they output the new end curve together with a proof of knowledge of a path between the previous and the current end curve. Under standard isogeny assumptions, we show that the protocol is secure in the Generalized Universally Composable framework. The reliance on fundamental assumptions and a statistical zero-knowledge proof make the protocol particularly reliable: if the protocol is insecure, then so are the protocols that require a SECUER.

Lastly, we develop an optimized implementation of the proposed zero-knowledge proof. The results show that the proof is sufficiently efficient to practically deploy a trusted-setup ceremony. We conclude with a discussion on how to concretely do so.

[3]This is strictly true only when $p \equiv 1 \bmod 12$. Otherwise, the distribution of the output curves with subgroups is statistically close to the stationary distribution, which differs from the uniform one due to the presence of special curves $j = 0$ and $j = 1728$.

Part I

Analysis of Existing Protocols

Chapter 2

On Adaptive Attacks against Jao-Urbanik's Isogeny-Based Protocol

How beautiful the world would be if there were a procedure for moving through labyrinths.

— U. Eco, Il nome della rosa

This chapter is a verbatim reproduction of the following paper:

Andrea Basso, Péter Kutas, Simon-Philipp Merz, Christophe Petit, and Charlotte Weitkämper. "On Adaptive Attacks Against Jao-Urbanik's Isogeny-Based Protocol". In: *AFRICACRYPT 20*. Ed. by Abderrahmane Nitaj and Amr M. Youssef. Vol. 12174. LNCS. Springer, Heidelberg, July 2020, pp. 195–213. DOI: 10.1007/978-3-030-51938-4_10

I contributed to the development of the proposed attack.

Abstract: The k-SIDH protocol is a static-static isogeny-based key agreement protocol. At Mathcrypt 2018, Jao and Urbanik introduced a variant of this protocol which uses non-scalar automorphisms of special elliptic curves to improve its efficiency.

In this paper, we provide a new adaptive attack on Jao-Urbanik's protocol. The attack is a non-trivial adaptation of Galbraith-Petit-Shani-Ti's attack on SIDH (Asiacrypt 2016) and its extension to k-SIDH by Dobson-Galbraith-LeGrow-Ti-Zobernig (IACR eprint 2019).

Our attack provides a speedup compared to a naïve application of Dobson et al.'s attack to Jao-Urbanik's scheme, exploiting its inherent structure. Estimating the security of k-SIDH and Jao-Urbanik's variant with respect to these attacks, k-SIDH provides better efficiency.

2.1 — Introduction

With the expected advent of quantum computers, current public key cryptography algorithms based on discrete logarithm and factorization problems will have to be replaced by stronger, so-called post-quantum cryptography algorithms. Isogeny-based cryptography is among the leading approaches currently considered for post-quantum cryptography. A major protocol in isogeny-based cryptography is the SIDH key exchange protocol [JD11], whose principles underlie the SIKE algorithm recently submitted to the NIST post-quantum standardization process [JACCD+20].

In internet communication contexts, key exchange protocols are often used in a semi-static mode, where the server uses the same *static* secret key to establish any new session key with a client. Galbraith et al. have shown that the basic SIDH protocol is vulnerable to adaptive attacks in these contexts [GPST16]. In SIKE the attacks are defeated by using a variant of the Fujisaki-Okamoto transform.

The k-SIDH protocol is an alternative countermeasure to Galbraith et al.'s attack suggested by Azarderakhsh et al. [AJL17]. The protocol has the additional advantage to allow for static-static key exchange (where both parties use static keys), but it comes at the cost of a significant efficiency loss as it essentially involves running k^2 instances of the SIDH protocol in parallel for an integer $k > 1$, with $k = 92$ suggested by the authors. Very recently, Dobson et al. described an adaptive attack against the 2-SIDH protocol [DGLTZ20]. Their attack also generalizes to the k-SIDH protocol with $k > 2$,

though the required number of instances of the protocol with the server is exponential in k.

Our contributions. In this paper, we provide a new adaptive attack on a variant of the k-SIDH protocol suggested by Jao and Urbanik [UJ20]. The Jao-Urbanik protocol introduces some redundancy in k-SIDH's secret keys using the non-trivial automorphisms of curves with j-invariants 0 or 1728 to increase efficiency. While the authors of the protocol conjectured that the inherent structure could be exploited in attacks and chose larger security parameters to account for this, we provide a concrete attack.

Our attack borrows from Galbraith et al. and Dobson et al.'s attacks, but it crucially differs from them in the following ways:

- We use the underlying relationship between the kernel generators of corresponding curves to match up triples of candidate curves instead of exhaustively searching over all possibilities when querying for the first key bits.

- Instead of separately computing the key bits and pullbacks at any step of the attack, we combine these stages by guessing the key bits and computing candidate pullbacks first to then validate any possible combination using the oracle.

- Contrasting to the attack in [DGLTZ20], we manage to compute precise pullbacks at each step instead of having to keep track of multiple candidates which are indistinguishable to the attacker.

- Overall, we significantly reduce the number of oracle queries by exploiting the structure underlying the Jao-Urbanik protocol.

We show that our attack requires to run $\mathcal{O}(32^{k/3})$ instances of the protocol with the server, if the Jao-Urbanik protocol is instantiated with secret isogenies of degree a power of two. This is almost a cube root speedup compared to Dobson et al.'s attack on the same instantiation.

While our attack does not break the security level for the parameter sets recommended by Jao and Urbanik, we give estimated attack costs for their parameters. Under consideration of currently known attacks against k-SIDH and Jao-Urbanik's protocol, we conclude that the former provides a better efficiency-security trade-off.

Outline. The remaining of this paper is organized as follows. To begin with, we give some background on isogenies and supersingular isogeny protocols in Section 2.2. We then recall the Dobson et al. attack on k-SIDH in Section 2.3 and the Jao-Urbanik protocol in Section 2.4. We continue by describing our attack on Jao-Urbanik's scheme in Section 2.5, and conclude the paper in Section 2.6. The Appendix includes an extension of our attack.

2.2 — Preliminaries

For a full treatment of background information on elliptic curves we refer to Silverman [Sil86].

2.2.1 – Isogenies

Let $\mathbb{F}_q$ be a finite field of characteristic p. In the following we assume $p > 3$ and therefore an elliptic curve E over $\mathbb{F}_q$ can be defined by its short Weierstrass form

$$E(\mathbb{F}_q) = \{(x, y) \in \mathbb{F}_q^2 \mid y^2 = x^3 + Ax + B\} \cup \{\mathcal{O}_E\},$$

where $A, B \in \mathbb{F}_q$ and $\mathcal{O}_E$ is the point $(X : Y : Z) = (0 : 1 : 0)$ on the projective curve $Y^2Z = X^3 + AXZ^2 + BZ^3$. The set of points on an elliptic curve forms an abelian group with $\mathcal{O}_E$ being the identity element. The *j-invariant* of an elliptic curve is

$$j(E) = 1728\frac{4A^3}{4A^3 + 27B^2},$$

and there is an isomorphism $f : E \to E'$ between the curves E and E' if and only if $j(E) = j(E')$.

Given two elliptic curves E_1 and E_2 over a finite field $\mathbb{F}_q$, an *isogeny* is a morphism $\phi : E_1 \to E_2$ such that $\phi(\mathcal{O}_{E_1}) = \mathcal{O}_{E_2}$. The condition implies that isogenies are also group homomorphisms. If there exists an isogeny $\phi : E_1 \to E_2$, then there exists a unique isogeny $\hat{\phi} : E_2 \to E_1$, called the *dual* isogeny, such that $\phi \circ \hat{\phi} = [n]$ (where $[n]$ denotes the multiplication-by-n map on E_2). If there exists a non-constant isogeny between two curves, then they are called *isogenous*. The *degree* of an isogeny ϕ is its degree when treated as an algebraic map. If the isogeny is separable (which is always the case in this work), the degree is equal to the size of the kernel of ϕ. An isogeny from E to itself is called an endomorphism. Endomorphisms of an elliptic curve form a ring under addition and composition. If E is defined over a finite field then the endomorphism ring is either an order in an imaginary quadratic number field (such curves are called *ordinary*) or an order in the quaternion algebra ramified at p (the characteristic of the finite field) and at infinity. The latter curves are called *supersingular*. In this paper we will only consider supersingular elliptic curves.

Since an isogeny defines a group homomorphism $E_1 \to E_2$, its kernel is a subgroup of E_1. Conversely, any subgroup $S \subset E_1$ determines a (separable) isogeny $\phi : E_1 \to E_2$ with $\ker(\phi) = S$ and $E_2 = E_1/S$. Furthermore, if the degree of the isogeny is smooth, Vélu's formulae [Vél71] provide a polynomial time algorithm for computing the isogeny (as a rational map) from its kernel.

The following lemma [Sil86, Chapter III, Corollary 4.11] describes how the isogenies corresponding to two subgroups can be related if one subgroup contains the other:

Lemma 1. *Let E_i, $i = 1, 2, 3$ be elliptic curves and let $\phi : E_1 \to E_2$ and $\psi : E_1 \to E_3$ be two isogenies such that $ker(\phi) \subseteq ker(\psi)$. Then there exists an isogeny $\lambda : E_2 \to E_3$ such that $\psi = \lambda \circ \phi$ which is unique up to isomorphism.*

2.2.2 – SIDH

In this subsection, we recall Jao and De Feo's original scheme [JD11].

Let E be a supersingular elliptic curve. In the setup, one chooses two small primes ℓ_A and ℓ_B and a prime p which is of the form $p = \ell_A^{e_A}\ell_B^{e_B}f - 1$, where f is a small cofactor and e_A and e_B are large integers. Let P_A, Q_A be generators of the $\ell_A^{e_A}$-torsion and let P_B, Q_B be generators of the $\ell_B^{e_B}$-torsion of E. Then the protocol is as follows:

1. Alice chooses a random cyclic subgroup of $E[\ell_A^{e_A}]$ of order $\ell_A^{e_A}$. As P_A, Q_A form a basis of the $\ell_A^{e_A}$-torsion, there exist integers x_A, y_A such that $A = [x_A]P_A + [y_A]Q_A$ generates this subgroup. Similarly, Bob chooses a random cyclic subgroup of $E[\ell_B^{e_B}]$ of order $\ell_B^{e_B}$ generated by $B = [x_B]P_B + [y_B]Q_B$ for some x_B, y_B.

2. Alice computes the isogeny $\phi_A : E \to E/\langle A\rangle$ and Bob computes the isogeny $\phi_B : E \to E/\langle B\rangle$.

3. Alice sends the curve $E/\langle A\rangle$ and the points $\phi_A(P_B)$ and $\phi_A(Q_B)$ to Bob and Bob similarly sends ($E/\langle B\rangle$, $\phi_B(P_A)$, $\phi_B(Q_A)$) to Alice.

4. Alice and Bob both use the images of the torsion points to compute the shared secret which is the curve $E/\langle A, B\rangle$ (e.g. Alice can compute $\phi_B(A) = [x_A]\phi_B(P_A) + [y_A]\phi_B(Q_A)$ and $E/\langle A, B\rangle = E_B/\langle\phi_B(A)\rangle$).

Due to efficiency reasons in [JD11], the authors suggested the use of $\ell_A = 2$ and $\ell_B = 3$. They also suggested to use the starting curve E with j-invariant 1728. In [AJL17], the authors use a variant of the Fujisaki-Okamoto transform [HHK17] to obtain an IND-CCA secure key encapsulation mechanism. For concrete parameters of the scheme the reader is referred to [AJL17].

Note that by [GPST16, Lemma 2.1], it is possible for Alice (and analogously for Bob) to always choose the secret integers x_A, y_A such that one of them equals 1 given that the generators P_A, Q_A of the 2^{e_A}-torsion are independent. Hence it suffices to choose a single

secret instead of two integers. In practice, this is usually done for efficiency reasons, and we will also use the convention in the following.

In [GPST16] Galbraith et al. propose an adaptive attack against SIDH, showing that SIDH is not suitable for static-static key exchange; see Section 2.2.4 for a description of the GPST attack.

2.2.3 – k-SIDH

Now we recall the k-SIDH scheme of Azarderakhsh et al. [AJL17]. This protocol is a modification of the original SIDH which is potentially secure against active attacks. The protocol is as follows. Both parties agree on a curve E as well as a basis of the 2^{e_A}-torsion and a basis of the 3^{e_B}-torsion. Alice chooses k different secret integers $\alpha^{(1)}, \ldots, \alpha^{(k)}$ modulo 2^{e_A} and Bob chooses k different secret integers $\beta^{(1)}, \ldots, \beta^{(k)}$ modulo 3^{e_B}. Let h be a preimage resistant hash function. The steps of the protocol are the following:

1. Alice computes the curves $E_A^{(r)} = E/\langle P_A + [\alpha^{(r)}]Q_A \rangle$ and the corresponding isogenies $\phi_{A,r}$.

2. Bob computes the curves $E_B^{(r)} = E/\langle P_B + [\beta^{(r)}]Q_B \rangle$ and the corresponding isogenies $\phi_{B,r}$.

3. Alice sends $E_A^{(r)}$, $\phi_{A,r}(P_B)$, $\phi_{A,r}(Q_B)$ to Bob and Bob sends $E_B^{(r)}$, $\phi_{B,r}(P_A)$, $\phi_{B,r}(Q_A)$ to Alice.

4. Alice and Bob perform the SIDH key exchange for every pair $E_A^{(r)}, E_B^{(s)}$ and compute the corresponding j-invariant $j_{r,s}$.

5. The shared secret is the hash $h(j_{1,1}||j_{1,2}||\ldots||j_{k,k})$ of all the j-invariants.

2.2.4 – The GPST attack on static SIDH

The adaptive GPST attack actively recovers the static SIDH key α of a party, say Alice, where $\langle P_A + [\alpha]Q_A \rangle$ is the subgroup corresponding to her secret isogeny. An attacker

uses the key exchange protocol as an oracle to recover Alice's static key bit-wise. For simplicity, we set $n := e_A$ in the following.

Definition 1 (Oracle in static SIDH)**.** *Upon receipt of an elliptic curve E, two linearly independent points $R, S \in E[2^n]$ of order 2^n and another elliptic curve E', the oracle responds 1 if $j(E/\langle R + [\alpha]S\rangle) = j(E')$ and 0 otherwise.*

To recover Alice's secret key, an attacker first generates the ephemeral key $(E_B, R := \phi_B(P_A), S := \phi_B(Q_A))$ honestly as specified by the SIDH key exchange. Then, they query the oracle on $(E_B, R, S+[2^{n-1}]R, E_{AB})$, which reveals whether $E_B/\langle R+[\alpha](S+[2^{n-1}]R)\rangle$ is isomorphic to $E_B/\langle R + [\alpha]S\rangle$. By the following lemma, this reveals the least significant bit of the static secret α.

Lemma 2. *[GPST16, Lemma 2] For linearly independent $R, S \in E[2^n]$ of order 2^n, α is even if and only if $\langle R + [\alpha](S + [2^{n-1}]R)\rangle = \langle R + [\alpha]S\rangle$.*

Afterwards, the attacker can proceed iteratively for all but the last two bits. Assume the attacker has recovered the i least significant bits of α, i.e. the partial key $K_i := \sum_{k=0}^{i-1} \alpha_k 2^k$ such that $\alpha = K_i + \alpha_i 2^i + \alpha' 2^{i+1}$. To learn the next bit $\alpha_i \in \{0, 1\}$, the attacker queries the oracle on

$$\big(E_B, [\theta](R - [2^{n-i-1}][K_i]S), [\theta]([1 + 2^{n-i-1}]S), E_{AB}\big). \tag{2.1}$$

Here, θ is a suitable scaling parameter to avoid detection of the attack by Weil pairing validation. We omit further details as this has no relevance to the methods presented in this paper, and we refer to the original paper [GPST16] for the computational details. In this exposition we omit such factors for simplicity.

The bit α_i is deduced from the oracle's answer using the following lemma.

Lemma 3 ([GPST16])**.** *The oracle call* (2.1) *returns 1 if and only if $\alpha_i = 0$.*

Proof. The curve computed by Alice is E_B/G' where $G' = \langle R' + [\alpha]S'\rangle = \langle (R - [2^{n-i-1}][K_i]S) + [\alpha]([1 + 2^{n-i-1}]S)\rangle = \langle R + [\alpha]S + [\alpha - K_i][2^{n-i-1}]S)\rangle$. This is equal to G if and only if $\alpha_i = 0$. □

The last two bits α_{n-2}, α_{n-1} should be brute-forced, as there is no suitable scaling parameter θ to avoid detection by Weil pairing validation. Note that this does not require any oracle query.

2.3 — The DGLTZ attack

The DGLTZ attack [DGLTZ20] follows roughly the same methodology as the GPST one. In this section, let $\alpha^{(r)}$ denote Alice's k secret keys associated to the kernel generators $A^{(r)} = R + [\alpha^{(r)}]S$ for some points R, S spanning $E[2^n]$. For simplicity we will largely only use two secret keys α, β with corresponding kernel generators A, B. Then we denote by α_i the i-th bit of $\alpha = K_i^{(a)} + \alpha_i 2^i + \alpha' 2^{i+1}$, where $K_i^{(a)}$ is the i-th partial key, and analogously for β. Dobson et al. first justify the existence of the following oracle.

Definition 2 (Oracle in k-SIDH)**.** *Let H be some public hash function. Upon receipt of an elliptic curve E, two points R, S spanning $E[2^n]$ and a hash value h, the oracle reveals whether $h = H\big(j(E/\langle R + [\alpha^{(1)}]S\rangle), \ldots, j(E/\langle R + [\alpha^{(k)}]S\rangle)\big)$.*

Note that this oracle provides information related to the k-tuple of static secret keys $(\alpha^{(1)}, \ldots, \alpha^{(k)})$, but it does not immediately reveal information on each individual secret key separately.

To compensate for this limited information, multiple oracle queries will be made using the same malicious points but different hash values. After obtaining the curves $E_{A^{(i)}} := E/\langle A^{(i)}\rangle$ from Alice's public keys, the attacker successively recovers the next bit of all the different secrets simultaneously. This is done by using malicious points in oracle queries as in the GPST attack, guessing all the j-invariants computed by Alice as a result of these malicious points, and verifying each guess with an oracle query.

The attacker recovers the first bit of all secrets with queries of the form $\big(E, R, [1 + 2^{n-1}]S, H(j_1 || \ldots || j_k)\big)$, where the j_i are guesses on the k shared secret curves computed by Alice. Candidate tuples for the guess can be restricted by the following lemma.

Lemma 4. *Let α be any of Alice's secret keys. Consider the isogeny path from E to E_A, and replace the last step in this path by the only other possible step that leaves the path non-backtracking. Let E'_A be the final curve of this path. Let $s \in \{0,1\}$. Let $R' := R - [s][2^{n-1}]S$ and $S' := [1+2^{n-1}]S$. Then the SIDH key computed by Alice is either E_A or E'_A. Moreover, it is E_A if and only if $\alpha_0 = s$.*

The number of candidate tuples is 7^k as for each secret there are 7 possible curves they have to query (the respective $E_{A^{(i)}}$ and six curves which are 4-isogenous to it). In the iterative step the attacker uses queries of the form $(E, R - [K_i^{(a)}][2^{n-i-1}]S, [1 + 2^{n-i-1}]S, H(j_1||\ldots||j_k))$, which correspond to the following elliptic curves: $E/\langle A+[\alpha_i][2^{n-1}]S\rangle$, $E/\langle B + [K_i^{(b)} - K_i^{(a)}][2^{n-i-1}]S + [\beta_i][2^{n-1}]S\rangle$. If to recover the next bits one wanted to perform a similar exhaustive search as for the first bit computation, then one would need an exponential amount of queries even for $k = 2$ as the distance (in the isogeny graph) from the second curves to E_A increases as i grows. To remedy this, the authors observe that $E/\langle B + [K_i^{(b)} - K_i^{(a)}][2^{n-i-1}]S + [\beta_i][2^{n-1}]S\rangle$ is 2-isogenous to $E_i/\langle\psi_{B,i}(B + [K_i^{(b)} - K_i^{(a)}][2^{n-i-1}]S + [\beta_i][2^{n-1}]S)\rangle$ where E_i is the $(n-i)$-th curve in the isogeny path from E to E_B and $\psi_{B,i}$ is the corresponding partial isogeny. In order to be able to compute these curves, one has to compute certain intermediate points on E_i (which the authors refer to as "pullbacks"), namely $\psi_i(B)$ and $[2^{n-i}]\psi_i(S)$. This pullback-computation is required after each key bit has been recovered, and at the i-th step makes use of the known partial keys with the following query:

$$\big(E, R - [K_{i+1}^{(a)}][2^{n-i-1}]S, [1 + 2^{n-i-1}]S, H(j_1, \ldots, j_k)\big).$$

It can be computed that the corresponding curve is $E_{i+1}/\langle\psi_{B,i+1}(B+[K_{i+1}^{(b)}-K_{i+1}^{(a)}][2^{n-i-1}]S)\rangle$ (and not 2-isogenous to it as in the previous stage). Naïvely, the attacker would query the oracle with all the possibilities for $\psi_{B,i+1}(B)$ and $[2^{n-i-1}]\psi_{B,i+1}(S)$. Note however that when the oracle returns 1, there will be two possibilities for the correct pullbacks which, due to the oracle model, cannot be distinguished. One could either have found $\psi_{B,i+1}(B)$

and $[2^{n-i-1}]\psi_{B,i+1}(S)$ or $\psi_{B,i+1}(B)+C$ and $[2^{n-i-1}]\psi_{B,i+1}(S)+C$, where C generates the kernel of the isogeny from E_{i+1} to E_i. Thus the authors choose one pullback $\psi_{B,i}(B)$ for B and then have to keep a 2-element set of candidates for $[2^{n-i-1}]\psi_{B,i+1}(S)$. The computation of bits uses 24^k queries[1] and the pullback computation uses 16^k queries under certain technical conditions which are addressed in the appendix of [DGLTZ20]. At each step, the intermediate isogenies are computed using the following lemma:

Lemma 5. *Let $A^{(i)} = P + [\alpha^{(i)}]Q$ be the generator of the subgroup corresponding to the i-th secret isogeny and let $\psi_j^{(i)} := \phi_n^{(i)} \circ \phi_{n-1}^{(i)} \circ \cdots \circ \phi_{j+1}^{(i)}$. Then, we have*

$$\ker \phi_j^{(i)} = \langle [2^{j-1}]\psi_j^{(i)}(A^{(i)})\rangle, \qquad \ker \hat{\phi}_j^{(i)} = \langle [2^{n-1}]\psi_{j-1}^{(i)}(Q)\rangle.$$

2.4 — The Jao-Urbanik protocol

In this section, we present the Jao-Urbanik protocol [UJ20], the main target of our attack.

To reduce the cost associated to k-SIDH [AJL17], Jao and Urbanik propose to exploit the existence of non-trivial automorphisms on certain elliptic curves for a non-interactive key exchange by using distinct isogenies between isomorphic curves. As in the original SIDH proposal [JD11], the authors suggest choosing parameters as follows: Let ℓ_A and ℓ_B be two small primes, e_A and e_B integers such that $\ell_A^{e_A} \approx \ell_B^{e_B}$; then choose a small cofactor f such that $p = \ell_A^{e_A}\ell_B^{e_B} f \pm 1$ is prime. To simplify our description, we will again set $\ell_A = 2$ and $\ell_B = 3$ (as widely used in discussions of SIDH) when describing the protocol here.

The only elliptic curves with non-trivial automorphisms are curves with j-invariants $j \in \{0, 1728\}$; note these are all supersingular over $\mathbb{F}_p$ for $p = 2^{e_A}3^{e_B} f - 1$ since $p \equiv 2 \pmod 3$ and $p \equiv 3 \pmod 4$. As Jao and Urbanik primarily suggest to use the former, we focus on curves with $j(E) = 0$ in this exposition. For such curves, there exists an automorphism η_6 of order six defined by $\eta_6(x, y) = (\zeta_3 x, -y)$ for ζ_3 a primitive third root of unity. Thus, η_6 further satisfies $\eta_6^2 = \eta_6 - 1$.

[1] Note that this estimation is not given in [DGLTZ20].

The existence of these automorphisms can be exploited in the following way. If $G \subseteq E$ is a subgroup, $\eta_6(G)$ and $\eta_6^2(G)$ are also subgroups of E and we may assume that all three are distinct[2]. Hence, the isogenies from E associated to the kernels $G, \eta_6(G)$ and $\eta_6^2(G)$, respectively, are all distinct while the corresponding quotients are isomorphic. For example, consider $\phi : E \to E/G$; the map $\phi \circ \eta_6^{-1} : E \to E/G$ has kernel $\eta_6(G)$ and hence we have $E/G \cong E/\eta_6(G)$. In an SIDH-setting when Alice sends a public key $(E_A, \phi_A(P_B), \phi_A(Q_B))$, we can thus view this as Alice actually having sent three distinct but related public keys. These keys all have isomorphic target curves $E/\langle A\rangle \cong E/\langle \eta_6(A)\rangle \cong E/\langle \eta_6^2(A)\rangle$, and hence share the same j-invariant, but the corresponding isogenies are not isomorphic. The same applies to any of Bob's public keys.

Lemma 6. *Suppose a base curve E with $j(E) = 0$ together with the parameters as suggested by Jao and Urbanik [UJ20] is used for SIDH. Then a single exchange of Alice's and Bob's SIDH public keys $pk_A = (E_A, \phi_A(P_B), \phi_A(Q_B))$ and $pk_B = (E_B, \phi_B(P_A), \phi_B(Q_A))$, where $\{P_A, Q_A = \eta_6(P_A)\}$ and $\{P_B, Q_B = \eta_6(P_B)\}$ are bases of $E[2^{e_A}]$ and $E[3^{e_B}]$ respectively, yields three shared secret (isomorphism classes of) curves.*

It follows that per public key pair, Alice and Bob obtain three shared secret curves, each identified by its j-invariant, as a secret in the Jao-Urbanik version of SIDH; see Figure 2.1. Hence, in the k'-SIDH setting where each party sends k' public keys, using the Jao-Urbanik technique results in a shared secret

$$h = \mathrm{Hash}(j_{1,1}||j'_{1,1}||j''_{1,1}|| \ldots ||j_{k',k'}||j'_{k',k'}||j''_{k',k'}),$$

obtained by hashing the concatenation of the j-invariants corresponding to the $k = 3(k')^2$ shared secret curves instead of the $(k')^2$ curves as in standard k'-SIDH.

[2]We have $\eta_6(G) = G$ exactly when $G \subset \ker(\eta_6 + k)$ for some odd k. Note that this is impossible since $\eta_6^2 - \eta_6 + 1 = 0$ implies that $\deg(\eta_6) = \mathrm{tr}(\eta_6) = 1$ so that $\deg(\eta_6 + k) = (\eta_6 + k)(\bar{\eta_6} + k) = \deg(\eta_6) + k\,\mathrm{tr}(\eta_6) + k^2 = 1 + k + k^2$ is odd and hence not divisible by 2^{e_A}.

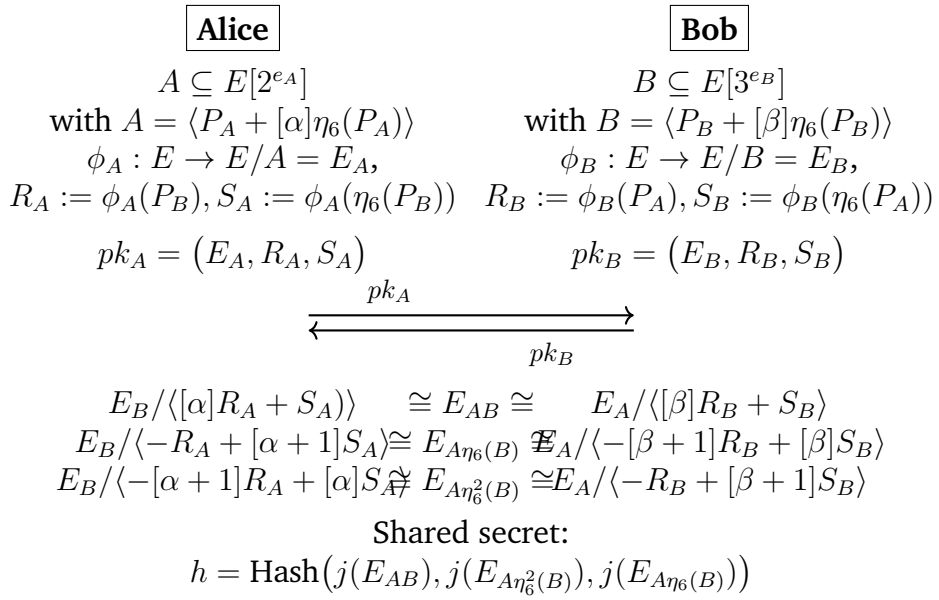

Figure 2.1: Jao-Urbanik's protocol using one key and automorphism η_6; public parameters: $E : y^2 = x^3 + 1$ with $j(E) = 0$ defined over field of characteristic $p = f2^{e_A}3^{e_B} - 1$, bases $\{P_A, \eta_6(P_A)\}$ of $E[2^{e_A}]$ and $\{P_B, \eta_6(P_B)\}$ of $E[3^{e_B}]$.

2.4.1 – Parameter selection

In [UJ20, Section 4] Jao and Urbanik discuss the security of their scheme for general $\ell := \ell_A$. They correctly identify that the relationship between the curves can be exploited for an attack but do not consider this extra structure fully when providing an estimate on the security of the scheme. Based on their brief analysis, they suggest the use of $k' = 18$ keys for $\ell = 11$ when 256-bit security is required. We believe the proposed parameters are safe but that their security analysis could be elaborated on.

In their discussion, the authors do not disclose a precise attack model and consider an oracle which receives a list of curves and returns true if all of them are on the secret isogeny path $E \to E/\langle A\rangle$.[3] However, using such an oracle, the attack proposed by Jao-Urbanik is not optimal. We will show that the extra structure can be exploited further by realizing that all intermediate curves on the three paths associated to one secret are isomorphic. Furthermore, in [DGLTZ20] it is demonstrated that using the straightforward generalization of the GPST oracle to k-SIDH would lead to an exponential-time attack

[3]Note that the GPST attack [GPST16] shows how to implement a similar oracle for SIDH.

even for $k = 2$. In order to go around this issue, Dobson et al. compute extra points which increases the complexity of the attack substantially. In other words, in the k-SIDH setting, the cost of the call to an oracle which returns true if and only if all the guessed curves are on the correct path is not constant but exponential in k. This observation clearly applies to the Jao-Urbanik scheme as well.

2.4.2 – Current impact of DGLTZ on Jao-Urbanik protocol

Applying the DGLTZ attack to the Jao-Urbanik protocol is not straightforward. The DGLTZ attack assumes that all the secret kernels are of the form $\langle [\alpha]P + Q \rangle$ which is not the case in the Jao-Urbanik scheme due to the following. To one secret the following three kernels are associated: $\langle [\alpha]P + Q \rangle$, $\langle -P + [\alpha + 1]Q \rangle$, $\langle -[\alpha + 1]P + [\alpha]Q \rangle$. The parity of the coefficient of Q in the second and the third kernel is different, thus in particular, it is impossible that both of them are odd (hence for every λ-multiple of the kernel the coefficient of Q will be even). This difficulty could potentially be overcome, however a number of $\mathcal{O}(24^k)$ queries, where $k = 3k'$ and k' is the number of secrets, will still be required.

Our aim is that instead of treating the three curves independently we use that the three kernels are related and propose an attack in the next section which uses $\mathcal{O}(32^{\frac{k}{3}})$ queries, thus providing a nearly cube root speedup.

2.5 — Adaptive attack against the Jao-Urbanik scheme

In this section, we describe our adaptive attack on the η_6 case of the Jao-Urbanik protocol [UJ20]. Thus, the starting curve E has j-invariant 0 and admits an automorphism of order 6, η_6. We want to attack Alice's $\ell_A^{e_A}$-torsion, so for simplicity, we again write $\ell := \ell_A$ and $n := e_A$, and set $\ell = 2$ in our exposition. See Subsection 2.5.4 for a discussion on how this attack generalizes to larger ℓ. Let P and $Q = \eta_6(P)$ be such that $\{P, Q\}$ form a basis of $E[2^n]$ and let α be one of Alice's secret keys, to which we associate the following

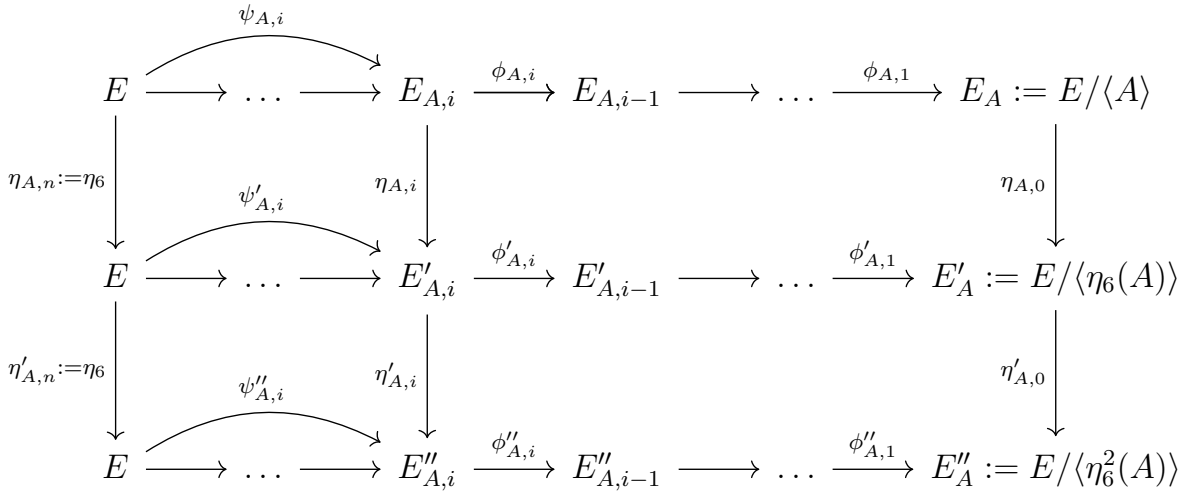

Figure 2.2: Isogeny paths between the relevant curves.

three kernel generators

$$A = [\alpha]P + Q, \qquad A' = \eta_6(A) = -P + [\alpha+1]Q,$$
$$A'' = \eta_6^2(A) = -[\alpha+1]P + [\alpha]Q,$$

and the three isogenies

$$\psi_{A,0} : E \to E_A = E/\langle A\rangle, \qquad \psi'_{A,0} : E \to E'_A = E/\langle A'\rangle,$$
$$\psi''_{A,0} : E \to E''_A = E/\langle A'\rangle.$$

Similarly, we denote with γ any other secret key different from α. The associated kernels are generated by C, C', C'', the curves are E_C, E'_C, E''_C and in general the notation corresponding to γ will have a subscript C. When there is no doubt about the corresponding secret key or when a property holds for all keys, we may drop the subscript.

The isogeny $\psi_{A,0}$ can be decomposed into n individual 2-isogenies. We index intermediate curves by $E_{A,i}$, with $E_{A,0} = E_A$ and $E_{A,n} = E$. The intermediate isogenies are denoted by $\phi_{A,i} : E_{A,i} \to E_{A,i-1}$. We also call $\psi_{A,i}$ the composition $\phi_{A,n} \circ \ldots \circ \phi_{A,i+1}$. We introduce similar notations for E'_A and E''_A, and denote by η_i the isomorphism between E_A and E'_A (see Lemma 8). We summarize all notations in Figure 2.2.

We define

$$A_i = \psi_{A,i}(A), \quad P_i = \psi_{A,i}(P).$$

Our attack is a non-trivial adaption of the GPST and DGLTZ attacks [GPST16; DGLTZ20]. It similarly has two stages. Firstly, we compute the first bit of each key (see Subsection 2.5.2) and we recover the "pullbacks" A_1, A'_1, A''_1 and $[2^{n-1}]P_1$, $[2^{n-1}]P'_1$, $[2^{n-1}]P''_1$ (for every secret A). In the second stage, we show inductively that given the first i bits of every key and A_i, $[2^{n-i}]P_{A,i}$ (for every secret A), we can deduce the $(i+1)$-th bit and the new pullbacks (see Subsection 2.5.3). In other words, if we write

$$\alpha = 2^{i+1}\alpha' + 2^i\alpha_i + K_{A,i},$$

where $K_{A,i}$ indicates the known part of the key, we can recover α_i from knowledge of the i-th pullbacks.

This is not dissimilar to what is done in the DGLTZ attack, but our attack exploits the additional structure between the shared secrets in the Jao-Urbanik protocol to recover the exact pullbacks at each step (instead of keeping two candidates) and reduce the number of queries needed for bit recovery. We thus show that the security of the Jao-Urbanik protocol with k' secret keys is only slightly better than the security of k'-SIDH, thus greatly decreasing the benefits of the Jao-Urbanik protocol. A more detailed study of the complexity of our attack can be found at the end of Subsection 2.5.3.

We present our attack only by querying with points on the starting curve E, as in the DGLTZ attack. Appendix 2.A presents a method to extend our attack to an arbitrary curve, which can also be applied to the DGLTZ attack.

We start by showing essential properties of the partial isogenies $\psi_{A,i}$, $\psi'_{A,i}$, $\psi''_{A,i}$ and of the corresponding curves $E_{A,i}$, $E'_{A,i}$, $E''_{A,i}$ in the following two lemmas.

Lemma 7. *For simplicity, denote subscripts of the form A, i by i. Then,*

$$\ker(\psi_i) = \langle [2^i]A \rangle, \qquad \ker(\psi_i') = \langle [2^i]A' \rangle, \qquad \ker(\psi_i'') = \langle [2^i]A'' \rangle,$$
$$\ker(\phi_i) = \langle [2^{i-1}]A_i \rangle, \qquad \ker(\phi_i') = \langle [2^{i-1}]A_i' \rangle, \qquad \ker(\phi_i'') = \langle [2^{i-1}]A_i'' \rangle,$$
$$\ker(\hat{\phi}_i) = \langle [2^{n-1}]P_{i-1} \rangle, \qquad \ker(\hat{\phi}_i') = \langle [2^{n-1}]P_{i-1}' \rangle, \qquad \ker(\hat{\phi}_i'') = \langle [2^{n-1}]P_{i-1}'' \rangle.$$

Lemma 8. *Let notation be as above. Then $E_{A,i}$, $E'_{A,i}$ and $E''_{A,i}$ are isomorphic.*

Proof. We have that $\ker(\psi_{A,i}) \subseteq \ker(\psi'_{A,i} \circ \eta_{A,n})$. Thus, there exists an isogeny $\eta_{A,i} : E_{A,i} \to E'_{A,i}$ such that $\psi'_{A,i} \circ \eta_{A,n} = \eta_{A,i} \circ \psi_{A,i}$. By examining the degrees, we find that $\deg \eta_{A,i} = 1$ and thus $\eta_{A,i}$ is an isomorphism. The same reasoning holds for $E''_{A,i}$. □

The isomorphisms $\eta_{A,i}$ and $\eta'_{A,i}$ are assumed to be known when $E_{A,i}$, $E'_{A,i}$ and $E''_{A,i}$ are known, since they can be easily computed (a 1-isogeny between two curves can be recovered in $\mathcal{O}(1)$).

2.5.1 – Attack model: a new oracle

In this section, we describe our assumptions and our attack model.

Firstly, let k' denote the number of Alice's secret keys. We assume that Alice has a static set of keys $\alpha^{(1)}, \ldots, \alpha^{(k')}$ and that the attacker impersonates Bob to recover Alice's secret keys. The attacker engages with Alice on sessions of Jao-Urbanik's protocol and sends particularly chosen data, not necessarily conforming to the protocol. By checking whether the two parties have obtained the same shared secret, the attacker may recover information on Alice's keys. We model this information leakage in terms of an oracle and represent each interaction with Alice as an oracle query.

An adaption of the second oracle presented in [DGLTZ20] to the η_6 variant of the Jao-Urbanik protocol gives an oracle $O'(E^{(1)}, \ldots, E^{(k')}, R^{(1)}, S^{(1)}, \ldots, R^{(k')}, S^{(k')}, h)$ that returns *true* if

$$h = \mathrm{Hash}(j_{1,1} || j_{1,2} || \ldots || j_{k',k'-1} || j_{k',k'}),$$

where $j_{r,s}$ denotes the concatenation of

$$j\left(E^{(r)}/\langle[\alpha^{(r)}]R^{(s)}+S^{(s)}\rangle\right),\ j\left(E^{(r)}/\langle-R^{(s)}+[\alpha^{(r)}+1]S^{(s)}\rangle\right),$$
$$j\left(E^{(r)}/\langle-[\alpha^{(r)}+1]R^{(s)}+[\alpha^{(r)}]S^{(s)}\rangle\right).$$

Similarly to what is done for the third oracle in [DGLTZ20], we can simplify the oracle by assuming that the attacker generates one secret key and sends repeated copies of the same curve and points. Note that any information that can be recovered with querying with distinct curves can also be recovered by querying with repeated copies of the same curve.

Hence, we obtain the following oracle

$$O(E,R,S,h)=O'(E,\ldots,E,R,S,\ldots,R,S,h), \tag{2.2}$$

which is the one we use in our attack. As noted in [DGLTZ20], the attacker could change one curve at each iteration, but all but one curves ($k'-1$, in this case) have to remain constant across iterations for the attack to succeed.

2.5.2 – Exploiting the additional structure: first step

Let us focus on one of Alice's secrets α. The attack extends straightforwardly to all the keys. In order to recover the first bits of α, the attacker sends the modified points $P'=[1+2^{n-1}]P$, $Q'=Q$, so that Alice uses the following kernels in her computation of the shared secret:

1. $\hat{A}=\langle[\alpha]P'+Q'\rangle=\langle[\alpha]P+Q+[\alpha_0][2^{n-1}]P\rangle$,

2. $\hat{A}'=\langle-P'+[\alpha+1]Q'\rangle=\langle-P+[\alpha+1]Q+[2^{n-1}]P\rangle$,

3. $\hat{A}''=\langle-[\alpha+1]P'+[\alpha]Q'\rangle=\langle-[\alpha+1]P+[\alpha]Q-[\alpha_0+1][2^{n-1}]P\rangle$.

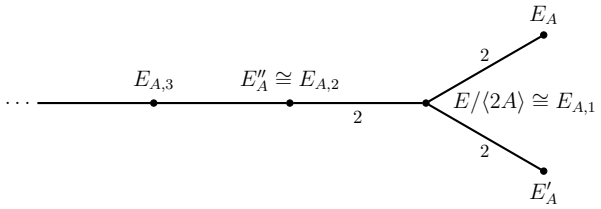

Figure 2.3: The isogeny paths between E_A, E'_A and E''_A.

Note that, depending on the value of the least significant bit α_0, either the first or third curve computed has not been altered by using the modified points. Thus the attacker already knows one of $j(\hat{E}_A)$ or $j(\hat{E}_{A''})$, where $\hat{E}_A = E/\langle \hat{A}\rangle$, although they do not know at this stage which one of the two.

The attacker now computes $\mathcal{E}^*_A$, the sets containing all six proper 4-neighbors of the curves E_A in Alice's public key, and their respective j-invariants. If $\alpha_0 = 0$, $\langle [\alpha]P' + Q'\rangle = \langle A\rangle$, and hence the first curve Alice obtains is isomorphic to her original E_A. The second curve is independent of α_0 and is a 4-neighbor of E'_A, since they share the 2-neighbor $E/\langle 2A'\rangle$. Similarly, the third curve is a 4-neighbor of E''_A since they share 2-neighbor $E/\langle 2A''\rangle$. Note that the intermediate 2-neighbors in this construction are isomorphic since their kernel generators differ only by an application of η_6. Hence, the three curves E_A, $E/\langle -P' + [\alpha+1]Q'\rangle$ and $E/\langle -[\alpha+1]P' + [\alpha]Q'\rangle$ are the three distinct 2-neighbors of $E/\langle 2A\rangle$ (distinctness follows from simple computations on the kernel generators), as depicted in Figure 2.3.

Analogously if $\alpha_0 = 1$, we find that the three computed curves all share a common 2-neighbor. The attacker proceeds analogously for the choices of any other curve. This allows the attacker to match up candidate curves for E_A, E'_A and E''_A among the 4-neighbors of E_A, depending on which combination of first key bits they are querying for at the time: the attacker may choose any curve in $\mathcal{E}^*_A$ as a candidate curve for E'_A, depending on the guessed bit they may select E_A or E''_A to be equal to E_A and then select the unique curve in $\mathcal{E}^*_A$ which is also a 4-neighbor of E'_A as a candidate for the remaining curve. Querying the oracle for all possible combinations ($12^{k/3}$ combinations, six for each neighbor and one for the curve itself) gives the attacker the first bit of each secret.

Now, given the position of E_A, E'_A and E''_A in the isogeny graph, we know that $E/\langle 2A\rangle$ must be the first intermediate curve $E_{A,1}$ and similarly E''_A must be $E_{A,2}$. This means the attacker can easily recover the first two intermediate curves without additional oracle queries, unlike what happens in the DGLTZ attack. Since the isogenies between E_A and $E_{A,1}$ (i.e. $\phi_{A,1}$) and between $E_{A,1}$ and $E_{A,2}$ (i.e. $\phi_{A,2}$) are known, the attacker can compute the first pullbacks of A and $[2^{n-1}]P$ (up to odd scalar multiplication) by setting A_1 to be a generator of $\ker(\phi_{A,1})$ and $[2^{n-1}]P_{A,1}$ a generator of $\ker(\hat{\phi}_{A,2})$ (see Lemma 7). Finally, the attacker obtains the pullbacks $A'_1 = \eta_{A,1}(A_1)$ and $A''_1 = \eta'_{A,1}(A_1)$. This approach can be easily repeated for every following curve.

2.5.3 – Intermediate bit and pullback computation

Suppose we have recovered the first i bits of each key and have the relevant pullbacks. Let α be one of Alice's secrets keys and let γ denote any other secret key. Now, we want to recover the $(i+1)$-th bit and compute the new pullbacks. In the DGLTZ attack, the bit recovery and pulling back are two separate stages, but in order to exploit the additional structure of Jao-Urbanik's scheme, we combine them together.

The attacker does not actively recover the $(i+1)$-th key bits, but instead tries all the $2^{k'}$ possibilities and uses the pullback queries to validate both the bit guesses and the pullback candidates. Using Lemma 7, it is possible to compute $\hat{\phi}_{i+1}$ and thus recover ϕ_{i+1}. With this information, the attacker can obtain candidates for the pullbacks of A and P. The same applies to ϕ'_{i+1} and ϕ''_{i+1}.

The attacker then queries the oracle with the following points

$$P' = [1 + 2^{n-i-1}]P, \quad Q' = Q - [K_{A,i}][2^{n-i-1}]P.$$

These are the oracle's internal kernel computations

$$\begin{aligned}
\langle [\alpha]P' + Q' \rangle &= \langle A + [\alpha_i][2^{n-1}]P \rangle, \\
\langle -P' + [\alpha+1]Q' \rangle &= \langle A' - [K_{A,i}^2 + K_{A,i} + 1][2^{n-i-1}]P \\
&\quad + [K_{A,i}][\alpha_i][2^{n-1}]Q \rangle, \\
\langle -[\alpha+1]P' + \alpha Q' \rangle &= \langle A'' - [K_{A,i}^2 + K_{A,i} + 1][2^{n-i-1}]P \\
&\quad - [K_{A,i} + 1][\alpha_i][2^{n-1}]P \rangle, \\
\langle [\gamma]P' + Q' \rangle &= \langle C + [K_{C,i} - K_{A,i}][2^{n-i-1}]P \\
&\quad + [\gamma_i][2^{n-1}]P \rangle, \\
\langle -P' + [\gamma+1]Q' \rangle &= \langle C' - [K_{C,i}K_{A,i} + K_{A,i} + 1][2^{n-i-1}]P \\
&\quad - [K_{A,i}][\gamma_i][2^{n-1}]P \rangle, \\
\langle -[\gamma+1]P' + [\gamma]Q' \rangle &= \langle C'' - [K_{C,i}K_{A,i} + K_{A,i} + 1][2^{n-i-1}]P \\
&\quad - [K_{A,i} + 1][\gamma_i][2^{n-1}]P \rangle.
\end{aligned}$$

All kernels can be shifted with ψ_{i+1} (e.g. $E/\langle C + [K_{C,i} - K_{A,i}][2^{n-i-1}]P + [\gamma_i][2^{n-1}]P \rangle = E_{C,i+1}/\langle C_{i+1} + [K_{C,i} - K_{A,i}][2^{n-i-1}]P_{C,i+1} + [\gamma_i][2^{n-1}]P \rangle$) similarly to the DGLTZ attack by applying [Sil86, Chapter III, Corollary 4.11.]. Now, since the candidate pullbacks for A_{i+1} (preimages of A_i via $\phi_{A,i}$), C_{i+1} (preimages of C_i via $\phi_{C,i}$), $[2^{n-i-1}]P_{C,i+1}$ (preimages of $\left[\frac{1}{2}\right][2^{n-i}]P_{C,i}$), $[2^{n-i-1}]P_{A,i+1}$ (preimages of $\left[\frac{1}{2}\right][2^{n-i}]P_{A,i}$) and their isomorphic correspondents are known, the attacker can query the oracle with the hash values of all $2^{k'}2^{k'}8^{k'}$ possibilities (2 for each bit, 2 for the kernel generator pullback candidates and $4 \cdot 2$ for the P pullback candidates). Note that the attacker may try a candidate for the first curve and then shift it to the second curve using the isomorphisms η_i or η_i' (therefore reducing an a priori complexity of 32^k to $32^{k'}$). We show that if we find a match, then we have found the correct pullbacks for C_{i+1} and $P_{C,i+1}$ as well as the correct key bits for C. First we prove a simple lemma about parities.

Lemma 9. *Let $K_{A,i}$, $K_{C,i}$ be natural numbers. Then,*

1. $K_{A,i}^2 + K_{A,i} + 1$ *is odd.*

2. *It is not possible that all of* $(K_{A,i} - K_{C,i})$, $(K_{A,i}K_{C,i} + K_{A,i} + 1)$ *and* $(K_{A,i}K_{C,i} + K_{C,i} + 1)$ *have the same parity.*

Proof. The first claim is trivial. For the second claim, observe that the sum of these quantities is even, thus it is not possible that all three of them are odd. If $K_{A,i} - K_{C,i}$ is even, then $K_{A,i}$ and $K_{C,i}$ have the same parity and then $K_{A,i}K_{C,i} + K_{A,i} + 1 = K_{A,i}(K_{C,i} + 1) + 1$ is odd. □

Now, we prove our main lemma.

Lemma 10. *If the oracle query returns* true*, then we have found* γ_i, C_{i+1} *and* $P_{C,i+1}$.

Proof. Suppose the attacker guesses that α_i is 0. It is clear from the above computation that we always get at least one match when we substitute C_{i+1}, γ_i and $P_{C,i+1}$. If $\gamma_i = 0$, then it follows from the computation of [DGLTZ20, Claim 1], that the number of matches for the first curve is exactly two. The other match corresponds to choosing $C_{i+1} + [2^i]C_{i+1}$ as the preimage of C_i and $[2^{n-i-1}]P_{C,i+1} + [2^i]C_{i+1}$ as the preimage of $\left[\frac{1}{2}\right][2^{n-i}]P_{C,i}$. Due to Lemma 9, it is not possible that $(K_{A,i} - K_{C,i})$, $(K_{A,i}K_{C,i} + K_{A,i} + 1)$ and $(K_{A,i}K_{C,i} + K_{C,i} + 1)$ are all odd. Assume for instance that $(K_{A,i} - K_{C,i})$ is odd and $(K_{A,i}K_{C,i} + K_{A,i} + 1)$ is even. Then we show that the second curve will not match as its kernel will be generated by $C'_{i+1} + [K_{C,i}K_{A,i} + K_{A,i} + 1][2^{n-i-1}]P_{C,i+1} + [2^i]C_{i+1}$. Hence it will be 4-isogenous to the queried curve. The other cases follow similarly.

When $\gamma_i = 1$, then there will be another match for the first curve. Namely when we pull back $\left[\frac{1}{2}\right][2^{n-i}]P_i$ as $[2^{n-i-1}]P_{i+1} + [2^{n-1}]P_{i+1}$. However, again a similar calculation to [DGLTZ20, Claim 1] (one has to distinguish cases depending on the parity of $K_{A,i}$ and $K_{C,i}$) shows that either the second or the third curve will not match. The calculations when the attacker guesses α_i to be 1 are analogous. □

Lemma 10 implies that for all secrets except α we know the correct bits and pullbacks (as otherwise we cannot receive 1 from the oracle). However, we have seen that the

coefficient $K_{A,i}^2 + K_{A,i} + 1$ is odd, thus there will be multiple matches. In order to retrieve α_i and the corresponding pullbacks we do another query with different points, switching $K_{A,i}$ with $K_{C,i}$. For this, we can use the previously computed pullbacks and thus only query the oracle 32 times (corresponding to the 32 possibilities for the pullbacks and the bit). Since the correct pullbacks are computed, we are able to recover the isogenies $\phi_{A,i+1}$ and $\phi_{C,i+1}$ using Lemma 7. Finally, since the next intermediate curves are computed we compute the isomorphisms between them. Thus, we have proven the following theorem.

Theorem 1. *1. There exists an algorithm that recovers the first bit of each secret using $\mathcal{O}(12^{k'}) = \mathcal{O}(12^{\frac{k}{3}})$ queries to the oracle defined in (2.2).*

2. There exists an algorithm that recovers the intermediate bits and pullbacks using $\mathcal{O}(32^{k'}) = \mathcal{O}(32^{\frac{k}{3}})$ queries to the oracle defined in (2.2).

2.5.4 – Attack costs for general ℓ

So far, we have demonstrated our attack on the Jao-Urbanik protocol with parameter choice $\ell = 2$ for simplicity. However, in their proposal, the authors suggest the use of $\ell = 11$ or $\ell = 13$ and further compute that $k' = 18$ keys are necessary to obtain security against Grover's algorithm for $\ell = 11$; see [UJ20, Section 4]. Thus we briefly assess the cost of our attack and the DGLTZ attack for arbitrary ℓ. We divide the discussion into two parts. First, we estimate the number of queries needed for computing the first key bits and later the number of queries needed in the iterative step.

The complexity estimate of our attack is a straightforward generalization of Theorem 1. During the recovery of the first bit of every key, we query - as before - for any of the $\ell^{k'}$ possible first ℓ-adic digit combinations by first fixing the curve (either E_A or E_A'' using notation as in Subsection 2.5.2) corresponding to the guessed key digit to be the curve given in Alice's public key. Then we select any of the $\ell(\ell+1)$ ℓ^2-neighbors of the correct curve to be E_A' and choose one of the remaining $\ell - 1$ curves which are ℓ^2-isogenous to both previously selected curves as the third curve associated to a given key. Hence,

for each possible combination of first key digits we have $\big(\ell(\ell+1)(\ell-1)\big)^{k'}$ choices of curves. Thus, there exists an algorithm which recovers the first digit of each secret using $\mathcal{O}(\ell^{k'}\ell^{3k'}) = \mathcal{O}(\ell^{4k'}) = \mathcal{O}(\ell^{\frac{4k}{3}})$ oracle queries.

For the iterative step, we again first guess the i-th ℓ-adic digits and then compute candidate preimages for the first curve and shift them to the other two curves using the respective isomorphisms. There are $\ell^{k'}$ possibilities for the digits and $\ell^{2k'}$ possibilities for each preimage. This implies that we need $\mathcal{O}(\ell^{5k'})$ queries in total.

Hence, for general ℓ, we can summarize our findings in the following theorem.

Theorem 2. 1. *There exists an algorithm that recovers the first digit of each secret using $\mathcal{O}(\ell^{4k'}) = \mathcal{O}(\ell^{\frac{4k}{3}})$ queries to the oracle defined in (2.2).*

2. *There exists an algorithm that recovers the intermediate digits and pullbacks using $\mathcal{O}(\ell^{5k'}) = \mathcal{O}(\ell^{\frac{5k}{3}})$ queries to the oracle defined in (2.2).*

2.5.5 – Comparison of k'-SIDH and Jao-Urbanik's protocol

Theorem 2 does not break the security parameters suggested by Jao and Urbanik. However, in order to assess the security gain of Jao-Urbanik's protocol, we compare it with the security of k'-SIDH for arbitrary ℓ. Since the DGLTZ method requires an extra step which computes the i-th digits and then uses that information to compute candidate pullbacks, the overall complexity of the attack is $\ell^{4k'}$ for k'-SIDH. The following table gives an overview of the number of SIDH-instances and public keys occurring when executing the different protocols, as well as the respective cost of attacking the ℓ-torsion.

Therefore, we can observe that the Jao-Urbanik protocol with k' secrets is as secure as $\frac{5k'}{4}$-SIDH when comparing necessary oracle queries. Consequently, it is more efficient to use $\frac{5k'}{4}$-SIDH than the Jao-Urbanik scheme with k' keys and the same ℓ when measuring security with respect to the currently known attacks, as the former has a computational cost equivalent to $3(k')^2$ SIDH exchanges, whereas the latter has a computational cost equivalent to $1.56(k')^2$ SIDH exchanges. Note that the Jao-Urbanik scheme maintains a

Table 2.1: Comparisons between Jao-Urbanik's scheme and k-SIDH

	# SIDH instances	# public key exchanges	Attack cost
Jao-Urbanik with k' keys	$3(k')^2$	$(k')^2$	$\mathcal{O}(\ell^{5k'})$
k-SIDH with $k = k'$	$(k')^2$	$(k')^2$	$\mathcal{O}(\ell^{4k'})$
k-SIDH with $k = \frac{5}{4}k'$	$(\frac{5}{4}k')^2 \approx 1.56(k')^2$	$\approx 1.56(k')^2$	$\mathcal{O}(\ell^{4\frac{5}{4}k'}) = \mathcal{O}(\ell^{5k'})$

moderate advantage in public key size, since it requires sharing k' keys, compared to the $\frac{5}{4}k'$ keys shared in k-SIDH.

2.6 — Conclusion

We have introduced an adaptive attack against Jao-Urbanik's protocol with parameter $\ell = 2$. While Jao and Urbanik suggest using $\ell = 11$ or $\ell = 13$, our attack can be extended to that case as briefly described in the previous section. The complexity of such an attack increases significantly, possibly reaching levels where the protocol is secure for the specified parameter sets. However, even in that case, our attack provides a nearly cubic speedup compared to a generic application of Dobson et al.'s attack against the Jao-Urbanik scheme. Assessing security of k-SIDH and Jao-Urbanik's variant of it with respect to currently known attacks, we conclude that Jao-Urbanik's protocol does not seem to offer a sufficient security improvement over k-SIDH with the same number of secret keys to justify the roughly two times more computations needed.

We leave a more thorough examination of whether a combination of stages in an attack on k-SIDH can evoke further optimizations to future work. Any potential improvements in the attack cost would then make it necessary to reevaluate the efficiency-security trade-off when comparing k-SIDH and the Jao-Urbanik protocol.

Acknowledgments We would like to thank David Jao and David Urbanik for their valuable comments and feedback on this work. Furthermore, we are grateful to Samuel Dobson, Steven D. Galbraith, Jason LeGrow, Yan Bo Ti, and Lukas Zobernig for their helpful clarifications regarding the DGLTZ attack.

Work by the second and fourth authors was supported by an EPSRC New Investigator grant (EP/S01361X/1).

2.A — Querying with E_B

The following lemma shows how to lift from the path $E_B \to E_{AB}$ to the path $E \to E_A$.

Lemma 11. *Let $\psi_{A,i}$ be the partial isogeny from E to E_i and let $\psi^B_{A,i}$ be the corresponding partial isogeny from E_B to E_{AB}. Let A be the kernel of the isogeny from E to E_A and let $A_B = \phi_B(A)$. Let E_i be the i-th curve in the isogeny path from E to E_A and E^B_i be the i-th curve in the isogeny path from E_B to E_{AB}. Let $\delta_i : E^B_i \to E_i$ be the isogeny which is the SIDH lift of ϕ_B. Assume we know $\psi'_i(A_B)$ and $\psi'_i(\phi_B(Q))$. Then we can compute $[3^n]\psi_i(A)$ and $[3^n]\psi_i(Q)$.*

Proof. The proof follows from the observation that $\delta_i \circ \psi'_i = \psi_i \circ \hat{\phi_B}$. □

The Lemma can be applied to compute the relevant pullbacks on the isogeny paths from E to E_A, E' to E'_A and E'' to E''_A in the following manner. First one computes a pullback candidate on the path starting from E_B. Then it is lifted with the above lemma to the path starting from E (using the fact that 3^n is odd). Then it can further be shifted to the other two isomorphic curves. Finally these points can be shifted back with ϕ_B.

Chapter 3

Another Look at Adaptive Attacks on SIDH: Breaking HealSIDH

All poets write bad poetry. Bad poets publish them, good poets burn them.
— U. Eco

This chapter is based on the following paper:

Andrea Basso, Tako Boris Fouotsa, Christophe Petit, and Charlotte Weitkämper. *Another Look at Adaptive Attacks on SIDH: Breaking HealSIDH*. Unpublished. 2022

I was the main author of the two attacks on HealSIDH and its improved variant, as well as the generalized GPST attack.

Abstract: The SIDH protocol is an efficient and practical isogeny-based key exchange. Its active security is threatened by the GPST attack, which allows a malicious attacker to extract the long-term secret key of an SIDH participant. Recently, Fouotsa and Petit suggested an efficient validation method that prevents adaptive attacks, leading to the key-exchange protocol HealSIDH.

In this work, we analyze the adaptive security of SIDH and related protocols. Firstly, we revisit the GPST attacks on SIDH and introduce a matrix notation that enables a better

understanding of the attack. In particular, it allows us to determine all possible malicious torsion points that could be used in a generalized GPST attack.

Furthermore, we use this formalism to rewrite the HealSIDH protocol and SIDH key validation method, which allows us to develop a GPST-like attack on HealSIDH. We show that an attacker can craft torsion points which do not affect the correctness of the SIDH exchange, while the validation method fails or succeeds depending on a single bit of the targeted secret key. We further discuss possible countermeasures, and we show they are similarly vulnerable to an extended version of our adaptive attack.

Lastly, we propose a translation of the HealSIDH countermeasure to the genus two setting, and we discuss possible approaches to translate the attack.

3.1 — Introduction

The Supersingular Isogeny Diffie-Hellman (SIDH) key exchange [JD11] is an efficient post-quantum protocol based on the hardness of finding an isogeny of fixed degree between a given pair of supersingular isogenous elliptic curves with the additional knowledge of some torsion point information. SIDH has been shown to be vulnerable against a series of active attacks: the simplest such attacks are discussed in [CLN16] and countered using a simple pairing check. Further adaptive attacks on SIDH were presented by Galbraith, Petit, Shani and Ti [GPST16] in 2016, namely the GPST attack, and more recently (2022) by Fouotsa and Petit [FP22]. While the Fujisaki-Okamoto transform [FO13] can detect and prevent active attacks, it requires that one party reveals their secret key; as such, it can only be used in the ephemeral-static setting. There is currently no efficient way to prevent such attacks in the static-static settings, where both parties need to maintain their secret information private.

The GPST attack consists in recovering an honest participant's static secret key over

multiple key exchange sessions by using public keys with malformed torsion point information. There have been several attempts to introduce effective countermeasures that would thwart the adaptive attacks and enable the usage of a static key over multiple SIDH instances. Most prominently, combining the basic SIDH protocol with a version of the Fujisaki-Okamoto transform due to Dent and Hofheinz, Hövelmanns, and Kiltz [FO13; Den03; HHK17] results in the NIST third round alternate candidate SIKE (Supersingular Isogeny Key Encapsulation) [JACCD+20]. This transform forces a party to reveal their secret key to the other and hence cannot be used with static encryption keys. There have been other attempts to prevent the GPST attacks. The k-SIDH protocol [AJL17] and an SIDH-variant by Urbanik and Jao [UJ20] propose methods to obtain multiple SIDH instances yielding several shared secret curves whose j-invariants are then hashed together. Unfortunately, both of these countermeasures have not succeeded at avoiding the vulnerabilities exhibited by static-static SIDH and have been shown to be ineffective at preventing GPST-style adaptive attacks [DGLTZ20; BKMPW20].

Recently, Fouotsa and Petit [FP21] presented a new key validation technique for SIDH at Asiacrypt 2021 which, instead of requiring multiple SIDH instances or the disclosure of a secret key, proves the honest generation of the torsion point information to the other party by providing larger torsion points and revealing additional information on ths shared curve. In this work, we show the claim is inaccurate by presenting an efficient strategy to recover HealSIDH secret keys. A similar goal was achieved in independent work by Galbraith and Lai [GL22].

Contributions. In this work, we revisit the GPST attack on SIDH and describe a generalised framework for these types of active torsion point attacks. This enables a better understanding and an improved abstraction of the original torsion point attack. We then present GPST-like adaptive attacks on HealSIDH. Furthermore, we discuss some other versions of the validation method that could have been used in HealSIDH and show that they are also vulnerable to adaptive attacks. In particular, we make the following contributions:

1. We introduce a new formalism that treats isogenies as linear maps on torsion groups. This new formalism leads to a better understanding of the GPST adaptive attacks. We use this formalism to give a general description of the GPST adaptive attack on SIDH [GPST16].

2. We present GPST-like adaptive attacks on HealSIDH. A first one extracts the largest power of two dividing the secret key, but this could be countered by simply fixing secret keys to be odd (this does not reduce the security of the scheme since the length of the secret is longer than λ bits). A second attack extracts the entire secret key. The attack relies on an honest execution of the underlying SIDH protocol but uses maliciously-generated validation points.

3. We exhibit a modified countermeasure technique that is resistant against our attack and the one in [GL22]. We also show, however, that the new countermeasure can be defeated by a more elaborate attack.

4. We also discuss the extension of the HealSIDH countermeasure to the genus two setting and the obstacles that prevent an application of the active attack shown before.

Outline. This paper is structured as follows: we present the technical background and existing active attacks in Section 3.2. We then introduce a generalisation of the GPST attack and present the Fouotsa–Petit (FP) countermeasure using a more formalised notation for the SIDH protocol in Section 3.3. In Section 3.4, we first describe the HealSIDH key exchange oracle and present a straightforward adaptive attack on HealSIDH that recovers the largest power of two dividing Alice's static secret key α. We proceed by describing a GPST-style attack on HealSIDH which fully recovers the secret key in $O(\log \alpha)$ queries to the key exchange oracle. We conclude in Section 3.5 by briefly summarising and discussing some avenues for further inquiry: we explore the applicability of the attacks on a variant of the FP countermeasure (which appears to be equally vulnerable) and investigate the extension of the FP countermeasure to Genus-Two SIDH [FT19].

3.2 — Preliminaries

In this section, we recall relevant information on elliptic curves and isogenies. We refer to Silverman [Sil86], De Feo's notes [Feo17] or the following theses [Fou22; Pan21] for a more thorough treatment. Furthermore, we briefly describe SIDH, the adaptive attacks on SIDH, and the existing countermeasures.

3.2.1 – Elliptic curves and isogenies

An *elliptic curve* is a smooth rational curve of genus one with a distinguished point $\mathcal{O}$, which in this work we model to be at infinity. Isomorphism classes can be represented by the j-invariant associated with the curves in the class. The set of rational points on an elliptic curve can be seen as commutative groups with respect to a group addition having the point at infinity as the neutral element. When an elliptic curve E is defined over a finite field $\mathbb{F}_q$, the set of $\mathbb{F}_q$-rational points $E(\mathbb{F}_q)$ of E is a subgroup of $E(\bar{\mathbb{F}}_q)$. For every integer N coprime with q, the N-torsion subgroup $E[N]$ of $E(\mathbb{F}_q)$ is isomorphic to $\mathbb{Z}/N\mathbb{Z} \oplus \mathbb{Z}/N\mathbb{Z}$. An elliptic curve E defined over $\mathbb{F}_{p^k}$ is supersingular if $E[p] = \{\mathcal{O}\}$, i.e. there is no point of order not coprime with the characteristic except for the neutral point. The j-invariant of supersingular elliptic curve is always defined over $\mathbb{F}_{p^2}$, and thus for any supersingular curve there exists an isomorphic curve which is defined over $\mathbb{F}_{p^2}$.

An *isogeny* from E_1 to E_2 is a non-constant rational map from E_1 to E_2 which is also a group morphism. The kernel of an isogeny is always finite and entirely defines the isogeny up to isomorphism and powers of the Frobenius map. Given a finite subgroup G of a supersingular curve $E_1(\mathbb{F}_q)$, there exists a Frobenius-free isogeny of domain E_1 with kernel equal to G. We call such an isogeny *separable*. Its degree is equal to the size of its kernel, that is $\#G$. The codomain of this isogeny is denoted by E_1/G. The isogeny and the codomain E_1/G can be computed from the knowledge of the kernel using Vélu's formulae [Vél71] whose efficiency depends on the smoothness of the isogeny degree. If

$\phi : E_1 \to E_2$ is an isogeny of degree d and $P, Q \in E_1[N]$ where N is an integer, then

$$e_N(\phi(P), \phi(Q)) = e_N(P, Q)^d$$

where e_N is the N-Weil pairing.

Let $\phi : E_1 \to E_2$ be a separable isogeny of degree d. Then there exists a unique isogeny $\hat{\phi}$ with the property that $\phi \circ \hat{\phi} = [d]$ and $\hat{\phi} \circ \phi = [d]$, where $[d]$ denotes multiplication by d on E_2 and E_1. This isogeny $\hat{\phi}$ is called the *dual* isogeny of ϕ and it is also of degree d. The kernel of $\hat{\phi}$ is $\phi(E_1[d])$. In the rest of the paper, we will restrict ourselves to supersingular elliptic curves and separable isogenies.

3.2.2 – An overview of SIDH

We give a very brief description of the SIDH [JD11] key exchange. The protocol proceeds as follows.

Setup. The public parameters of SIDH are a prime p of the form $p = ABf - 1$, where $A = 2^a$, $B = 3^b$ for some choice of a and b and f is a small cofactor, and a supersingular elliptic curve E_0 defined over $\mathbb{F}_{p^2}$ together with points P_A, Q_A, P_B, Q_B such that $E_0[A] = \langle P_A, Q_A \rangle$ and $E_0[B] = \langle P_B, Q_B \rangle$.

KeyGeneration. Alice chooses a random integer $\alpha \in \mathbb{Z}/A\mathbb{Z}$ and computes the A-isogeny $\phi_A : E_0 \to E_A = E_0/G_A$ where $G_A = \langle P_A + [\alpha]Q_A \rangle$. Bob chooses a random integer $\beta \in \mathbb{Z}/B\mathbb{Z}$ and computes the B-isogeny $\phi_B : E_0 \to E_B = E_0/G_B$ where $G_B = \langle P_B + [\beta]Q_B \rangle$.

KeyExchange. Alice sends the curve E_A and the two points $\phi_A(P_B), \phi_A(Q_B)$ to Bob. Similarly, Bob sends $\big(E_B, \phi_B(P_A), \phi_B(Q_A)\big)$ to Alice. Alice and Bob use the given torsion points to obtain the shared secret $j(E_0/\langle G_A, G_B \rangle)$. To do so, Alice computes $\phi_B(G_A) = \langle \phi_B(P_A) + [\alpha]\phi_B(Q_A) \rangle$ and uses the fact that $E_0/\langle G_A, G_B \rangle \cong E_B/\phi_B(G_A)$. Bob proceeds analogously.

Note that revealing the torsion point images in SIDH is a crucial feature of the protocol: without them, it is not known how to make the diagram in Fig. 3.1 commute without

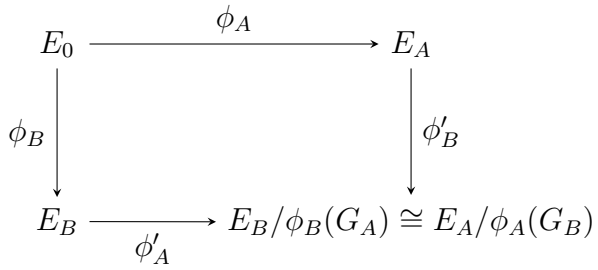

Figure 3.1: Overview of the SIDH protocol.

knowledge of the other party's secret. Despite being essential for the success of the exchange in SIDH, providing these torsion points also makes the protocol vulnerable. For example, torsion point information has been used to design both adaptive [CLN16; GPST16; FP22] and passive attacks [Pet17; QKLMP+21] on SIDH.

3.2.3 – Adaptive attacks and countermeasures

We recall the most prominent adaptive attacks on SIDH and some existing attempts to counter them.

Several validation methods have been proposed to protect a party's static secret key. The simplest checks ensure that the additional information provided by Bob to make the diagram in Fig. 3.1 satisfies two criteria: the torsion points belong to the provided elliptic curve, and they generate a basis of the correct torsion subgroup. These checks make it possible for Alice to detect the most unrefined attacks and can easily be performed using Weil pairing computations.

The more sophisticated Galbraith–Petit–Shani–Ti (GPST) attack [GPST16] exploits the fact that Alice does not have any means other than the ones previously mentioned to check whether Bob generates his public key honestly, i.e. according the SIDH protocol specifications. Even with the restrictions on Bob's points guaranteed by the checks presented above, The GPST attack shows that the points can be manipulated so that Bob can learn one bit of Alice's static secret key with each malicious interaction. In particular, the points are modified to force the shared secret of the interaction to be the same as the one resulting from an honest interaction if and only if the targeted bit of Alice's key is

zero. By checking whether the key exchange was successful, a malicious Bob can learn the value of the targeted bit. We give a more detailed description of the strategy of GPST in Section 3.3.3.

Further variations of the SIDH protocol have been suggested to avoid its vulnerabilities to the GPST attack: Azarderakhsh, Jao and Leonardi [AJL17] proposed k-SIDH, a protocol where each party chooses k secret keys (instead of a single one). This allows each party to compute k different kernel subgroups and corresponding isogenies. The two parties can then compute an SIDH exchange for each of the possible key pairs, which results in k^2 shared elliptic curves. Taking a hash of the concatenation of all j-invariants finally provides the shared secret. The presence of the hash in the shared secret makes it hard to apply the GPST attack, since modified torsion points that target one secret key cannot target all the others at once. However, Dobson, Galbraith, LeGrow, Ti, and Zobernig [DGLTZ20] showed that the protocol is insecure for small values of k, as the protocol is vulnerable to a GPST-like adaptive attack, whose complexity grows exponentially in k. To ensure sufficient security, the protocol need to choose high value of k (≈ 90), which incurs in a significant efficiency loss.

A further variant aiming to reduce the communication cost of k-SIDH was proposed by Urbanik and Jao [UJ20]. The authors note that for specific primes, the elliptic curves with j-invariants $j = 0$ and $j = 1728$ have non-trivial automorphisms of order six and four, respectively. Thus, it is possible to attach multiple SIDH instances to a single pair of secret SIDH keys. For example, suppose μ is the non-trivial order-four automorphism on E_{1728} and Alice uses a secret key corresponding to a kernel subgroup A while Bob uses the subgroup B. Then, a regular execution of the SIDH protocol yields the shared curve $E_{1728}/\langle A, B\rangle$. However, if each party also additionally uses the subgroup obtained by applying μ, from this single key pair, Alice and Bob can find the curves $E_{1728}/\langle A, B\rangle, E_{1728}/\langle \mu(A), B\rangle, E_{1728}/\langle A, \mu(B)\rangle$ and $E_{1728}/\langle \mu(A), \mu(B)\rangle$. It turns out that the first and last of these curves as well as the second and third are isomorphic (since their kernels only differ by the application of an automorphism), so this variant yields two

distinct curves (up to isomorphism) per secret key pair. In the order-six automorphism case, three distinct shared curves (up to isomorphism) can be obtained from each key pair. Again, this variant can be attacked via a GPST-like attack: Basso, Kutas, Merz, Petit, and Weitkämper [BKMPW20] showed that the interrelation of the kernels can be exploited during a key recovery attack; hence, the Jao–Urbanik variant does not provide the desired efficiency improvements over k-SIDH.

3.3 — A new notational approach

In this section, while providing some additional technical background, we present elliptic curve isogenies, the SIDH protocol and relevant related techniques using a more formalised notation in the hope that this will make the adaptive attack on HealSIDH presented in the following more intuitive. We introduce this alternative notation first before reframing SIDH, the GPST attack and the Fouotsa–Petit countermeasure using this formalism. We especially highlight currently available countermeasures to avoid attacks on a static-static SIDH setting and present the GPST attack in such a way that exposes a general technique to launching similar attacks on SIDH and any of its variants.

3.3.1 – Notation

Let $A = 2^a$ and $B = 3^b$ such that the integer $p = ABf - 1$ is a prime for some small cofactor f. For some prime power N (we will mainly consider $N \in \{A, A^2, B, B^2\}$), let μ_N be a canonically chosen primitive N-th root of unity. For every supersingular elliptic curve E defined over $\mathbb{F}_{p^2}$, let the row vector $\mathcal{B}_{E,N} = \begin{bmatrix} P & Q \end{bmatrix}$ denote a canonically generated basis of $E[N]$ such that the Weil pairing on the basis points satisfies

$$e_N(\mathcal{B}_{E,N}) := e_N(P, Q) = \mu_N.$$

Such a basis can be efficiently computed when the order N is smooth: one generates a basis $\begin{bmatrix} U & V \end{bmatrix}$, computes $\gamma = e_N(U, V)$, solves for t such that $\mu_N = \gamma^t$, and sets $P = U$, $Q = [t]V$.

Given $\mathcal{B}_{E,N}$, a point $R \in E[N]$ can alternatively be represented by the column vector of its coordinates $\begin{bmatrix} \gamma & \eta \end{bmatrix}_N^{\mathsf{tr}}$ where γ and η are the coordinates of R in the basis $\mathcal{B}_{E,N}$, i.e. $R = [\gamma]P + [\eta]Q$, and tr denotes the transpose. The cyclic group generated by the point $R = \mathcal{B}_{E,N} \cdot \begin{bmatrix} \gamma & \eta \end{bmatrix}_N^{\mathsf{tr}}$ is denoted (projectively) by $[\gamma : \eta]_N^{\mathsf{tr}}$. In the following, we might drop the subscript and use $\begin{bmatrix} \gamma & \eta \end{bmatrix}^{\mathsf{tr}}$ or $[\gamma : \eta]^{\mathsf{tr}}$ if it is clear from the context which basis the coordinates of the point or the group generator refer to. For instance, given a non-canonical basis $\mathcal{B}$, we may write $R = \mathcal{B} \cdot \begin{bmatrix} \gamma & \eta \end{bmatrix}^{\mathsf{tr}}$ or $G = \mathcal{B} \cdot [\gamma : \eta]^{\mathsf{tr}}$ to refer to a point or a cyclic group with respect to the basis $\mathcal{B}$.

For every supersingular isogeny $\phi : E_1 \to E_2$ of degree A, let $\mathcal{M}_{\phi,B} \in M_2(\mathbb{Z}/B/\mathbb{Z})$ be the matrix that satisfies

$$\phi(\mathcal{B}_{E_1,B}) = \begin{bmatrix} \phi(P) & \phi(Q) \end{bmatrix} = \mathcal{B}_{E_2,B} \cdot \mathcal{M}_{\phi,B},$$

where A, B are not necessarily coprime prime powers, and $\mathcal{B}_{E_1,B} = \begin{bmatrix} P & Q \end{bmatrix}$. Computing the image of some point $R = \mathcal{B}_{E_1,B} \cdot \begin{bmatrix} x & y \end{bmatrix}^{\mathsf{tr}} \in E_1[B]$ through ϕ is as follows:

$$\phi(R) = \mathcal{B}_{E_2,B} \cdot \mathcal{M}_{\phi,B} \cdot \begin{bmatrix} x \\ y \end{bmatrix}.$$

Note that by definition, the bases $\mathcal{B}_{E_1,B}$ and $\mathcal{B}_{E_2,B}$ have the same pairing value $e_B(\mathcal{B}_{E_1,B}) = \mu_B = e_B(\mathcal{B}_{E_1,B})$. Hence, the determinant of the matrix $\mathcal{M}_{\phi,B}$ is equal to the degree of ϕ modulo B, i.e. $A \pmod B$, as shown in the following lemma. This property will be helpful in our analyses later, and it motivates our choice of canonical basis as above.

Lemma 12. *Let $A = 2^a$ and $B = 3^b$ and consider an isogeny $\phi : E \to E_B$ of degree B.*

Further let $\mathcal{B}_{E,A}$ and $\mathcal{B}_{E_B,A}$ be the canonical A-torsion bases of E and E_B respectively, and denote by $\mathcal{M}$ the matrix corresponding to the action of ϕ on $E[A]$. Then

$$e_A(\phi(\mathcal{B}_{E,A})) = e_A(\mathcal{B}_{E_B,A})^{\deg\phi} = e_A(\mathcal{B}_{E_B,A})^{\det\mathcal{M}}.$$

Proof. The first equality follows from the fact that A, the order of the pairing and the torsion considered, and B, the degree of ϕ, are coprime. Further, if we let $\mathcal{B}_{E,A} = \begin{bmatrix} P & Q \end{bmatrix}$, $\mathcal{B}_{E_B,A} = \begin{bmatrix} P' & Q' \end{bmatrix}$ and we write $\mathcal{M} = \begin{bmatrix} a & c \\ b & d \end{bmatrix}$, then

$$e_A(\phi(\mathcal{B}_{E,A})) = e_A(\phi(P),\phi(Q)) = e_A(aP'+bQ', cP'+dQ') = e_A(\mathcal{B}_{E_B,A})^{ad-bc}$$

by the bilinearity and alternating properties of the Weil pairing. □

It is further possible to express the composition of isogenies by multiplication of the corresponding matrices. Recall that $\phi : E_1 \to E_2$ is an isogeny of degree A with matrix $\mathcal{M}_{\phi,B} := \begin{bmatrix} a & c \\ b & d \end{bmatrix}$ expressing its action on $E[B]$. Let further $\psi : E_2 \to E_3$ be an isogeny of degree C and $\mathcal{M}_{\psi,B} := \begin{bmatrix} e & g \\ f & h \end{bmatrix}$ such that $\psi(\mathcal{B}_{E_2,B}) = \mathcal{B}_{E_3,B} \cdot \mathcal{M}_{\psi,B}$. Then, we have

$$\mathcal{M}_{(\psi\circ\phi),B} = \mathcal{M}_{\psi,B} \cdot \mathcal{M}_{\phi,B}$$

since, for fixed bases $\mathcal{B}_{E_i,B} = \begin{bmatrix} P^{(i)} & Q^{(i)} \end{bmatrix}$ for $i \in \{1,2,3\}$, we have

$$\begin{aligned}
\psi\circ\phi(\mathcal{B}_{E_1,B}) &= \psi\circ\phi\left(\begin{bmatrix} P^{(1)} & Q^{(1)} \end{bmatrix}\right) = \psi\left(\begin{bmatrix} P^{(2)} & Q^{(2)} \end{bmatrix}\begin{bmatrix} a & c \\ b & d \end{bmatrix}\right) \\
&= \psi\left(\begin{bmatrix} aP^{(2)}+bQ^{(2)} & cP^{(2)}+dQ^{(2)} \end{bmatrix}\right) \\
&= \begin{bmatrix} a\psi(P^{(2)})+b\psi(Q^{(2)}) & c\psi(P^{(2)})+d\psi(Q^{(2)}) \end{bmatrix}
\end{aligned}$$

$$= \psi\left(\begin{bmatrix} P^{(2)} & Q^{(2)} \end{bmatrix}\right) \cdot \begin{bmatrix} a & c \\ b & d \end{bmatrix} = \begin{bmatrix} P^{(3)} & Q^{(3)} \end{bmatrix} \cdot \begin{bmatrix} e & g \\ f & h \end{bmatrix} \cdot \begin{bmatrix} a & c \\ b & d \end{bmatrix}$$

$$= \mathcal{B}_{E_3,B} \cdot \mathcal{M}_{\psi,B} \cdot \mathcal{M}_{\phi,B}.$$

The following table summarizes the new notation and its translation to the more common notation.

New formalism	**Standard notation**
$\mathcal{B}_{E,N} = \begin{bmatrix} P & Q \end{bmatrix}$	$\langle P, Q \rangle = E[N], \quad e_N(P,Q) = \mu_N$
$R = \mathcal{B}_{E,N} \cdot \begin{bmatrix} \gamma & \eta \end{bmatrix}_N^{\mathsf{tr}}$	$R = [\gamma]P + [\eta]Q$
$\langle R \rangle = [\gamma : \eta]_N^{\mathsf{tr}}$	$\langle R = [\gamma]P + [\eta]Q \rangle$
$\phi_B(\mathcal{B}_{E,A}) = \mathcal{B}_{E_B,A} \cdot \mathcal{M}_{\phi_B,A}$	$\phi_B(P_A) = [a]P'_A + [b]Q'_A$, $\phi_B(Q_A) = [c]P'_A + [d]Q'_A$, for some $a, b, c, d \in \mathbb{Z}/A\mathbb{Z}$ where $E[A] = \langle P_A, Q_A \rangle, E_B[A] = \langle P'_A, Q'_A \rangle$
$\ker \phi_B = [1 : \beta]^{\mathsf{tr}} = \langle \mathcal{B}_{E,A} \cdot \begin{bmatrix} 1 & \beta \end{bmatrix} \rangle$	$\ker \phi_B = \langle P_A + [\beta]Q_A \rangle$
$\ker \phi'_A = \langle \mathcal{B}_{E_B,A} \cdot \mathcal{M}_{\phi_B,A} \cdot \begin{bmatrix} 1 & \alpha \end{bmatrix}^{\mathsf{tr}} \rangle$	$\ker \phi'_A = \langle \phi_B(P_A) + [\alpha]\phi_B(Q_A) \rangle$ $(= [a + \alpha c : b + \alpha d]^{\mathsf{tr}})$
$\mathcal{M}_{\phi'_A \circ \phi_B, A} = \mathcal{M}_{\phi'_A, A} \cdot \mathcal{M}_{\phi_B, A}$	$\phi_{A'B} = \phi'_A \circ \phi_B$

Table 3.1: Translation table between the new formalism and older standard notation.

3.3.2 – SIDH

With respect to the above notation, revealing the images of N-torsion points through an isogeny $\phi : E_1 \to E_2$ is equivalent to revealing $\mathcal{M}_{\phi,N}$. In fact, this is exactly what is done in the compressed version of SIKE [JACCD+20].

Setup. Let $A = 2^a$, $B = 3^b$ such that $p = ABf - 1$ is a prime. Let E_0 be a supersingular elliptic curve defined over $\mathbb{F}_{p^2}$. The public parameters are p, E_0, $\mathcal{B}_{E_0,A}$, and $\mathcal{B}_{E_0,B}$.

KeyGeneration. As proposed in [KTW22], Alice picks a random integer $\alpha \in \mathbb{Z}/A\mathbb{Z}$ and computes the A-isogeny $\phi_A : E_0 \to E_A$ of kernel $[1 : \alpha]_A^{\mathsf{tr}}$ together with $\mathcal{M}_{\phi_A,B}$. Her secret key is α and her public key is $(E_A, \mathcal{M}_{\phi_A,B})$. Bob proceeds analogously by picking a

random integer $\beta \in \mathbb{Z}/B\mathbb{Z}$ and computing the B-isogeny $\phi_B : E_0 \to E_B$ of kernel $[1 : \beta]_B^{\mathsf{tr}}$ together with $\mathcal{M}_{\phi_B,A}$. His secret key is β and his public key is $(E_B, \mathcal{M}_{\phi_B,A})$.

KeyExchange. Given $(E_B, \mathcal{M}_{\phi_B,A})$, Alice computes the A-isogeny $\phi'_A : E_B \to E_{BA}$ of kernel $\mathcal{M}_{\phi_B,A} \cdot [1 : \alpha]^{\mathsf{tr}}$. Given $(E_A, \mathcal{M}_{\phi_A,B})$, Bob computes the B-isogeny $\phi'_B : E_A \to E_{AB}$ of kernel $\mathcal{M}_{\phi_A,B} \cdot [1 : \beta]^{\mathsf{tr}}$. The j-invariants j_{BA} and j_{AB} of the curves E_{BA} and E_{AB} respectively are equal and are then used as the shared secret.

In order to counter some trivial adaptive attacks, Costello, Longa, and Naehrig [CLN16] suggest the use of the Weil pairing equality

$$e_N(\phi(P), \phi(Q)) = e_N(P, Q)^{\deg \phi},$$

where $\{N, \deg \phi\} = \{A, B\}$, to validate the public keys in SIDH. Since the canonical N-bases of $\deg \phi$-isogenous curves have the same pairing value, the pairing check reduces to verifying that the determinant of the matrix in a given public key is equal to the degree of the related secret isogeny. That is, Alice and Bob respectively check that

$$\det \mathcal{M}_{\phi_B,A} = B \pmod{A} \qquad \text{and} \qquad \det \mathcal{M}_{\phi_A,B} = A \pmod{B}.$$

Note that such countermeasure that rely on pairing checks certify the correctness of one entry, but it potentially leaves two degrees of freedom to play with for the attacker; this is exploited in the GPST attack.

3.3.3 – GPST adaptive attack on SIDH

In the GPST attack [GPST16], the authors design the following adaptive attack which is not countered by the above pairing validation method.

Recall that we have fixed $A = 2^a$ and $B = 3^b$. We suppose here that Bob is the malicious attacking party and aims to recover Alice's static secret key α. The key exchange oracle

is modeled by a map

$$O : SS_p \times GL_2(\mathbb{Z}/B\mathbb{Z}) \times SS_p \to \{0, 1\},$$

where SS_p denotes the set of all supersingular elliptic curves defined over $\mathbb{F}_{p^2}$, defined by

$$O(E, \mathcal{M}, E') = \begin{cases} 1 & \text{if } j(E') = j(E/\mathcal{M} \cdot [1 : \alpha]^{\mathsf{tr}}) \\ 0 & \text{otherwise} \end{cases}.$$

The GPST attack is an iterative technique where at each step, modified torsion points are provided to Alice, or equivalently the oracle O, whose construction utilises the previously recovered key bits. The oracle's response then lets Bob deduce the next bit of Alice's secret key α.

After having recovered the i least significant bits of α, say $K_i = \alpha \pmod{2^i}$ where α is written as $\alpha = K_i + 2^i\alpha_i + 2^{i+1}\alpha'$ with $\alpha_i \in \{0, 1\}$, Bob generates a malicious public key $(E_i, \mathcal{M}_i)$ such that

$$\det \mathcal{M}_i = B \pmod{A}$$

and

$$j(E_i') = j(E_i/\mathcal{M}_i \cdot [1 : \alpha]^{\mathsf{tr}}) \quad \text{if and only if} \quad \alpha_i = 0$$

so that

$$O(E_i, \mathcal{M}_i, E_i') = \begin{cases} 1 & \text{if } \alpha_i = 0 \\ 0 & \text{otherwise} \end{cases}$$

for some curve E_i'.

To generate a valid and expedient tuple $(E_i, \mathcal{M}_i, E_i')$ which can be used as input to the key exchange oracle, Bob first computes a well-formed public key $(E_B, \mathcal{M}_{\phi_B,A})$ together with the corresponding shared curve E_{AB} and uses this honest shared secret as a reference for the oracle queries. For each iterative step of the attack, he then right-multiplies $\mathcal{M}_{\phi_B,A}$

by the matrix

$$\mathcal{X}_i = \theta_i \begin{bmatrix} 1 & 0 \\ -2^{a-i-1}K_i & 1+2^{a-i-1} \end{bmatrix} \quad \text{where} \quad \theta_i = \sqrt{(1+2^{a-i-1})^{-1}}.$$

Note here that θ_i is well-defined for all but the last two key bits; these have to be recovered by brute force.

The input to the key exchange oracle at the i-th iteration is thus

$$E_i = E_B, \quad \mathcal{M}_i = \mathcal{M}_{\phi_B,A} \cdot \mathcal{X}_i, \quad E_i' = E_{AB}.$$

In particular, since the determinant of $\mathcal{X}_i$ is $1 \pmod A$, 'scaling' $\mathcal{M}_{\phi_B,A}$ with $\mathcal{X}_i$ does not affect the determinant check performed by Alice for key validation.

Generalizing the GPST attack. The new notational approach simplifies many computations and reasonings. As an example of this, we present a generalization of the GPST attack that gives the attacker more flexibility in computing the maliciously-generated torsion points. This is of independent interest because a thorough analysis of the possible attacks is an important step in understanding the capabilities of an attacker, and possibly in developing some countermeasures. While this could be explicitly computed in terms of the kernel generators, the new notation greatly simplifies the result and gives a concise argument.

In the original attack [GPST16], the attacker recovers the i-th key bit by applying the matrix $\mathcal{X}$ to the honestly-generated torsion points where $\mathcal{X}$ is given by

$$\mathcal{X}_i = \begin{bmatrix} 1 & 0 \\ -2^{a-i-1}K_i & 1+2^{a-i-1} \end{bmatrix} = \mathcal{I} + 2^{a-i-1} \underbrace{\begin{bmatrix} 0 & 0 \\ -K_i & 1 \end{bmatrix}}_{\Delta_i},$$

up to multiplication by some odd scalar to satisfy the pairing condition. This attack

strategy works because the vector $[1\ \alpha]^{\mathsf{tr}}$ is in the kernel space of the matrix

$$\begin{bmatrix} 1 & 0 \\ -2^{a-i-1}K_{i+1} & 1+2^{a-i-1} \end{bmatrix},$$

which relies on $i+1$ bits of the secret key. Thus, the vector $[1\ \alpha]^{\mathsf{tr}}$ is in the kernel space of Δ_i if and only if the i-th bit of α is zero, which can be used to bitwise recover the secret key. Moreover, the attack uses the identity matrix to obtain the same kernel generator as in the honest case when the vector $[1\ \alpha]^{\mathsf{tr}}$ is in the kernel space of Δ_i.

We can now extend the attack to other matrices with a more generalized form. Firstly, we want the matrix Δ' to have the vector $[1\ \alpha]^{\mathsf{tr}}$ in its kernel spaces. Hence, this property is satisfied by any matrix of the form

$$\Delta'_i = \begin{bmatrix} -xK_i & x \\ -yK_i & y \end{bmatrix},$$

where the values x and y can range over any value in $\mathbb{Z}/2^a\mathbb{Z}$. However, we also want that the vector $[1\ \alpha]^{\mathsf{tr}}$ is not in the kernel space of Δ'_{i+1} because this allows the attacker to distinguish zero bits from one bits. If the key α is even, then the value y needs to be odd to prevent both queries from passing. This means that for the first query, the attacker needs to pick an odd y to retrieve the parity of α, whereas for the subsequent queries y can be even if the key is odd. It may also be possible for an attacker to select the values of x and y in terms of K_i, i.e. x or y are polynomials evaluated at K_i. Secondly, in the case where the vector $[1\ \alpha]^{\mathsf{tr}}$ lies in the kernel space of Δ', the attack does not need to recompute the same kernel generator as in the honest case. Any odd scalar multiple of it suffices since the generated kernel is the same. Thus, the identity matrix can be replaced by $\lambda\mathcal{I}$ for any odd value λ.

In summary, a generalized GPST attack can be constructed with any matrix $\mathcal{X}'$ of the

form

$$\mathcal{X}' = \lambda\mathcal{I} + 2^{a-i-1}\begin{bmatrix} -xK_i & x \\ -yK_i & y \end{bmatrix} = \begin{bmatrix} \lambda - xK_i 2^{a-i-1} & x2^{a-i-1} \\ -yK_i 2^{a-i-1} & \lambda + y2^{a-i-1} \end{bmatrix},$$

where λ is an odd integer while x and y can be any integer. Such a matrix has determinant $\lambda(1 + (y - xK_i)2^{a-i-1})$, which is generally different from one. It would thus not pass the pairing check. However, the matrix $\mathcal{X}'$ can be scaled by a value $\theta_i = \sqrt{1/\det \mathcal{X}'}$ so that $\det(\theta\mathcal{X}') = 1$. Such a value θ_i always exists (except for the three highest values of i[1]) because the value λ is odd, which ensures an odd determinant of Δ'. Note that the original GPST attack corresponds to the case where $\lambda = 1$, $x = 0$, and $y = 1$.

3.3.4 – The FP countermeasure and HealSIDH

We now re-introduce the Fouotsa–Petit (FP) countermeasure from [FP21] using the formalised notation specified in Section 3.3.1.

Let E_0, A, B and p be the public parameters of an SIDH instance, and let $\mathcal{B}_{E_0,A}$ and $\mathcal{B}_{E_0,B}$ be the canonically generated torsion bases needed for the protocol execution. E_A and $\mathcal{M}_{\phi_A,B}$ represent Alice's public key. Suppose that Alice uses isogenies of degree $A = 2^a$ and Bob uses isogenies of degree $B = 3^b$ for some integers a and b. In SIDH, Bob computes a cyclic isogeny $\phi_B : E_0 \to E_B$ of degree B as well as the matrix $\mathcal{M}_{\phi_B,A}$ describing its action on the A-torsion of E_0. The torsion point information $\mathcal{M}$ provided by Bob is said to be *correct* if there exists $\lambda \in (\mathbb{Z}/A^2\mathbb{Z})^\times$ such that $A\mathcal{M} = \lambda \cdot A \cdot \mathcal{M}_{\phi_B,A^2}$. Note that in this definition, the correctness mostly concerns the A-torsion points.

As a countermeasure to the GPST adaptive attack on SIDH, Fouotsa and Petit [FP21, §3] suggest that Bob reveals the action of his secret isogenies ϕ_B and ϕ'_B on the A^2-torsion groups of the respective starting curve. Thus, Alice can verify that

$$[A] \circ \phi_B = \widehat{\phi'_A} \circ \phi'_B \circ \phi_A \tag{3.1}$$

[1] In this case, the remaining bits of the secret key have to be brute-forced. The same issue arises in the original GPST attack.

on $E_0[A^2]$. The authors rely on the equation

$$\phi'_A \circ \phi_B = \phi'_B \circ \phi_A \quad \text{on} \quad E_0[A^2] \tag{3.2}$$

to design their countermeasure.

Concretely, the FP countermeasure is based on the following theorem which can be obtained from rewriting [FP21, Theorem 2] with the notation we introduced in Section 3.3.1.

Theorem 3 ([FP21, Theorem 2]). *Let $p = A^2B^2f - 1$ and let E_0 be a supersingular elliptic curve defined over $\mathbb{F}_{p^2}$. Let $(E_B, \mathcal{M}_{\phi_B,A^2})$ be Bob's public key. Then Bob's torsion points are correct if $e_{A^2}(\mathcal{B}_{E_B,A^2} \cdot \mathcal{M}_{\phi_B,A^2}) = e_{A^2}(\mathcal{B}_{E_0,A^2})^B$ and $\mathcal{M}_{\phi'_B,A^2} \cdot \mathcal{M}_{\phi_A,A^2} = \mathcal{M}_{\phi'_A,A^2} \cdot \mathcal{M}_{\phi_B,A^2}$.*

The authors further design HealSIDH, a variant of SIDH which integrates their countermeasure. We will show in Section 3.4 that Theorem 3 does not hold, and that HealSIDH is vulnerable to adaptive attacks. The HealSIDH key exchange is as follows.

Setup. Let $p = A^2B^2f - 1$ be a prime. Let E_0 be a random supersingular elliptic curve with unknown endomorphism ring defined over $\mathbb{F}_{p^2}$. The public parameters are p, E_0, A^2 and B^2 with $\mathcal{B}_{E_0,A^2}$ and $\mathcal{B}_{E_0,B^2}$ implicitly defined.

KeyGeneration. Alice picks a random integer $\alpha \in \mathbb{Z}/A\mathbb{Z}$ and computes the A-isogeny $\phi_A : E_0 \to E_A$ of kernel $[A : \alpha \cdot A]^{\mathsf{tr}}_{A^2}$ together with $\mathcal{M}_{\phi_A,B^2}$ and $\mathcal{M}_{\phi_A,A^2}$. Her secret key is α (the matrix $\mathcal{M}_{\phi_A,A^2}$ may be precomputed in KeyGeneration for efficiency reasons) and her public key is $(E_A, \mathcal{M}_{\phi_A,B^2})$. Bob proceeds similarly, he picks a random integer $\beta \in \mathbb{Z}/B\mathbb{Z}$ and computes the B-isogeny $\phi_B : E_0 \to E_B$ of kernel $[B : \beta \cdot B]^{\mathsf{tr}}_{B^2}$ together with $\mathcal{M}_{\phi_B,A^2}$ and $\mathcal{M}_{\phi_B,B^2}$. His secret key is $(\beta, \mathcal{M}_{\phi_B,B^2})$ and his public key is $(E_B, \mathcal{M}_{\phi_B,A^2})$.

Interaction. Given $(E_B, \mathcal{M}_{\phi_B,A^2})$, Alice computes the A-isogeny $\phi'_A : E_B \to E_{BA}$ of kernel $\mathcal{M}_{\phi_B,A^2} \cdot [A : \alpha \cdot A]^{\mathsf{tr}}$ together with $\mathcal{M}_{\phi'_A,B^2}$ and $\mathcal{M}_{\phi'_A,A^2}$. She sends $\mathcal{M}_{\phi'_A,B^2}$ to Bob. Given $(E_A, \mathcal{M}_{\phi_A,B^2})$, Bob computes the B-isogeny $\phi'_B : E_A \to E_{AB}$ of kernel $\mathcal{M}_{\phi_A,B^2} \cdot [B : \beta \cdot B]^{\mathsf{tr}}$ together with $\mathcal{M}_{\phi'_B,A^2}$ and $\mathcal{M}_{\phi'_B,B^2}$. He sends $\mathcal{M}_{\phi'_B,A^2}$ to Alice.

KeyExchangecompletion. Given $\mathcal{M}_{\phi_B,A^2}$ and $\mathcal{M}_{\phi'_B,A^2}$, Alice checks that the determinant of the received values is correct, i.e. $\det \mathcal{M}_{\phi_B,A^2} \equiv B \pmod{A^2}$, and she checks that

$$\mathcal{M}_{\phi'_B,A^2} \cdot \mathcal{M}_{\phi_A,A^2} = \mathcal{M}_{\phi'_A,A^2} \cdot \mathcal{M}_{\phi_B,A^2}$$

to verify Eq. (3.2). If the check is successful, the j-invariant j_{BA} of the curve E_{BA} is her shared secret. If not, she aborts. Given $\mathcal{M}_{\phi_A,B^2}$ and $\mathcal{M}_{\phi'_A,B^2}$, Bob checks that $\det \mathcal{M}_{\phi_A,B^2} \equiv A \pmod{B^2}$ and

$$\mathcal{M}_{\phi'_A,B^2} \cdot \mathcal{M}_{\phi_B,B^2} = \mathcal{M}_{\phi'_B,B^2} \cdot \mathcal{M}_{\phi_A,B^2}.$$

If the check is successful, the j-invariant j_{AB} of the curve E_{BA} is his shared secret. If not, he aborts.

3.4 — Breaking HealSIDH

In this section, we first present a simple attack that allows a malicious adversary to recover the largest power-of-two 2^e dividing Alice's secret α. The attack uses a GPST-style approach, and it can easily be prevented by some new countermeasures. Then, we extend the GPST approach to provide a polynomial-time adaptive attack against the Fouotsa–Petit validation method. This translates to an active key-recovery attack on HealSIDH, SHealS and HealS.

3.4.1 – Defining the HealSIDH key exchange oracle

Let $p = A^2B^2f - 1$ where $A = 2^a$ and $B = 3^b$, and let E_0 be a random supersingular elliptic curve with unknown endomorphism ring defined over $\mathbb{F}_{p^2}$. Let $(E_B, \mathcal{M})$ be Bob's public key and let $\mathcal{M}'$ be the action of ϕ'_B on $E_A[A^2]$, thus $\mathcal{M}'$ is a matrix in $\mathrm{GL}_2(\mathbb{Z}/A^2\mathbb{Z})$.

The validation in HealSIDH consists of verifying that

$$\det \mathcal{M} = B = \det \mathcal{M}' \pmod{A^2} \tag{3.3}$$

and

$$\mathcal{M}' \cdot \mathcal{M}_{\phi_A, A^2} = \mathcal{M}_{\phi'_A, A^2} \cdot \mathcal{M}. \tag{3.4}$$

Hence, the HealSIDH key exchange oracle is defined as follows.

$$O(E_B, \mathcal{M}, \mathcal{M}', E') = \begin{cases} 1 & \text{if } E' = E_{BA}, \text{Eq. (3.3)} \text{ and } \text{Eq. (3.4)} \text{ hold} \\ 0 & \text{otherwise} \end{cases}$$

In SIDH, the oracle is constructed by examining whether the two parties obtain the same shared secret. This requires further interactions between Alice and Bob, but in HealSIDH Alice explicitly aborts if the validation procedure fails. The proposed oracle can then be immediately realized.

An honest Bob would produce $\mathcal{M} = \mathcal{M}_{\phi_B, A}$ and $\mathcal{M}' = \mathcal{M}_{\phi'_B, A}$ while a malicious attacker could return $\mathcal{M} = \mathcal{M}_{\phi_B, A} \cdot \mathcal{X}$ and $\mathcal{M}' = \mathcal{M}_{\phi'_B, A} \cdot \mathcal{X}'$, i.e. return modified torsion points information obtained by applying the matrices $\mathcal{X}$ and $\mathcal{X}'$ in $\mathrm{GL}_2(\mathbb{Z})$ to the honestly-generated ones. In this case, Eq. (3.3) is equivalent to

$$\det \mathcal{X} = 1 = \det \mathcal{X}' \pmod{A}. \tag{3.5}$$

Eq. (3.4) can then be developed as follows.

$$\begin{aligned} & \mathcal{M}' \cdot \mathcal{M}_{\phi_A, A} && = && \mathcal{M}_{\phi'_A, A} \cdot \mathcal{M} \\ \Leftrightarrow \quad & \mathcal{M}_{\phi'_B, A} \cdot \mathcal{X}' \cdot \mathcal{M}_{\phi_A, A} && = && \mathcal{M}_{\phi'_A, A} \cdot \mathcal{M}_{\phi_B, A} \cdot \mathcal{X} \\ \Leftrightarrow \quad & \mathcal{M}_{\phi'_B, A} \cdot \mathcal{X}' \cdot \mathcal{M}_{\phi_A, A} && = && \mathcal{M}_{\phi'_B, A} \cdot \mathcal{M}_{\phi_A, A} \cdot \mathcal{X} \\ \Leftrightarrow \quad & \mathcal{X}' \cdot \mathcal{M}_{\phi_A, A} && = && \mathcal{M}_{\phi_A, A} \cdot \mathcal{X}, \end{aligned} \tag{3.6}$$

where we can replace $\mathcal{M}_{\phi'_B,A} \cdot \mathcal{M}_{\phi_A,A}$ with $\mathcal{M}_{\phi'_A,A} \cdot \mathcal{M}_{\phi_B,A}$ because of the commutativity of the SIDH diagram. The matrix $\mathcal{M}_{\phi'_B,A}$ also represents the action of an isogeny on torsion points whose order is coprime with the isogeny degree, hence the matrix is invertible.

In the remainder of this paper, our attack will be using matrices $\mathcal{X}$ and $\mathcal{X}'$ such that $A \cdot \mathcal{X} = A \cdot \mathcal{X}' = \mathcal{I} \pmod{A^2}$. As a consequence, the kernel of the isogeny ϕ'_A is always the correct one because the torsion points used to compute the kernel are scaled by A. Thus, the curve E' used in the oracle query also satisfies $E' = E_{BA}$. We can therefore simplify the oracle as follows.

$$O_s(\mathcal{X}, \mathcal{X}') = \begin{cases} 1 & \text{if Eq. (3.5) and Eq. (3.6) hold} \\ 0 & \text{otherwise} \end{cases}$$

3.4.2 – Recovering the power of two diving α

A kernel generator of Alice's isogeny is of the form $[A]P + [\alpha \cdot A]Q$, where $\mathcal{B}_{E_0,A^2} = \begin{bmatrix} P & Q \end{bmatrix}$. Hence $[A]\phi_A(P) = [-\alpha \cdot A]\phi_A(Q)$. That is, the matrix $\mathcal{M}_{\phi_A,A^2}$ of ϕ_A on $E_0[A^2]$ is of the form

$$\mathcal{M}_{\phi_A,A^2} = \begin{bmatrix} e_1 & e_2 \\ f_1 & f_2 \end{bmatrix} = \begin{bmatrix} -\alpha e_2 + Ax & e_2 \\ -\alpha f_2 + Ay & f_2 \end{bmatrix}. \tag{3.7}$$

Let $1 \leq i \leq a$ be an integer. To determine whether α is divisible by 2^i or not, Bob sets his public key to be $(E_B, \mathcal{M}_i)$ where

$$\mathcal{M}_i = \mathcal{M}_{\phi_B,A^2} \cdot \mathcal{X}_i \quad \text{with} \quad \mathcal{X}_i = \begin{bmatrix} 1 & 2^{2a-i} \\ 0 & 1 \end{bmatrix}$$

and he honestly reveals the action of ϕ'_B on $E_A[A^2]$, that is $\mathcal{M}'_i = \mathcal{M}_{\phi'_B,A^2}$ (or equivalently, $\mathcal{X}'_i = \mathcal{I}$).

Clearly, $\det \mathcal{X}_i = 1 = \det \mathcal{X}'_i$, hence Eq. (3.5) holds. Moreover,

$$\mathcal{M}_{\phi_A,A^2} \cdot \mathcal{X}_i = \begin{bmatrix} -\alpha e_2 + Ax & e_2 \\ -\alpha f_2 + Ay & f_2 \end{bmatrix} \cdot \begin{bmatrix} 1 & 2^{2a-i} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} -\alpha e_2 + Ax & e_2 - 2^{2a-i}\alpha e_2 \\ -\alpha f_2 + Ay & f_2 - 2^{2a-i}\alpha f_2 \end{bmatrix}$$

Since either e_2 or f_2 is invertible, then we have that

$$\begin{bmatrix} -\alpha e_2 + Ax & e_2 \\ -\alpha f_2 + Ay & f_2 \end{bmatrix} = \begin{bmatrix} -\alpha e_2 + Ax & e_2 - 2^{2a-i}\alpha e_2 \\ -\alpha f_2 + Ay & f_2 - 2^{2a-i}\alpha f_2 \end{bmatrix} \quad \text{iff} \quad \alpha \pmod{2^i} \equiv 0.$$

That is, $\mathcal{M}_{\phi_A,A^2} = \mathcal{M}_{\phi_A,A^2} \cdot \mathcal{X}_i$ if and only if α is divisible by 2^i. We hence get $O_s(\mathcal{X}_i, \mathcal{X}'_i) = 1$ if and only if α is divisible by 2^i.

An easy repair. A simple countermeasure to the above attack would consist of setting Alice's secret α to be an odd integer. Since the attack recovers the largest power of 2 dividing α, it is then impossible to apply it. Note that a similar attack applies when using isogenies with degrees equal to a power of three. Hence, this countermeasure would extend to setting the corresponding secret β to be an integer coprime to 3.

We now show that HealSIDH and the FP countermeasures are vulnerable to a full adaptive attack which cannot be prevented by these countermeasures.

3.4.3 – A complete key-recovery attack

Attack strategy. First, we note that recovering the matrix $\mathcal{M}_{\phi_A,A}$ is sufficient to recover the secret key α, because $\mathcal{M}_{\phi_A,A} \cdot \mathcal{B}_{E_A,4^a}$ (where $\mathcal{B}_{E_A,4^a}$ is a canonical basis on E_A) gives the image of P_A and Q_A, which form the canonical basis $\mathcal{B}_{E_0,4^a}$ under the isogeny ϕ_A. The point Q_A is always linearly independent (modulo 2^a) of the generator of $\ker \phi_A$ because it is computed as $[A](P_A + [\alpha]Q_A)$, and thus its scaled image $[A]\phi_A(Q_A)$ generates the kernel of the dual isogeny $\hat{\phi}_A$. More precisely, if we use the same notation as [FP21] and

write the matrix $\mathcal{M}_{\phi_A,A}$ as

$$\mathcal{M}_{\phi_A,A} = \begin{bmatrix} e_1 & e_2 \\ f_1 & f_2 \end{bmatrix},$$

the kernel of the dual isogeny $\hat{\phi}_A$ is generated by $[A]\mathcal{B}_{E_A,4^a}[e_2\ f_2]^{\mathsf{tr}}$. Either e_2 or f_2 is invertible, otherwise they would both be even resulting in the dual isogeny having an incorrect degree. Hence, we have that the kernel of $\hat{\phi}_A$ is generated by either

$$[A]\mathcal{B}_{E_A,A^2}\begin{bmatrix} e_2/f_2 \\ 1 \end{bmatrix} \qquad \text{or} \qquad [A]\mathcal{B}_{E_A,4^a}\begin{bmatrix} 1 \\ f_2/e_2 \end{bmatrix}.$$

Write $\hat{\alpha}$ for f_2/e_2 (or e_2/f_2 in the other case). The goal of the attack is to devise oracle queries such that the output of the oracle reveals one bit of $\hat{\alpha}$. After recovering the entire value $\hat{\alpha}$, the attacker can compute the isogeny $\hat{\phi}_A$ and derive its dual ϕ_A. Knowing the isogeny ϕ_A, it is straightforward to extract the secret key α.

Building the attack. We present an attack where Bob's public key is honestly generated but the action of ϕ'_B on the torsion group $E_A[A^2]$ is malicious, thus the matrix $\mathcal{X}$ is the identity while $\mathcal{X}'$ is not. We show it is possible to cheat only on the validation data and still obtain meaningful information about the secret key α. This also simplifies the attack because we are guaranteed that Alice's isogeny ϕ'_A is always the 'correct' one, i.e. the same isogeny she would have computed if Bob had been honest.

Let us start with Eq. (3.6), and assume $\mathcal{X} = \mathcal{I}$. We have that

$$\mathcal{X}'\begin{bmatrix} e_1 & e_2 \\ f_1 & f_2 \end{bmatrix} = \begin{bmatrix} e_1 & e_2 \\ f_1 & f_2 \end{bmatrix},$$

and thus the vector $[e_2\ f_2]^{\mathsf{tr}}$ is an eigenvector of $\mathcal{X}'$ with eigenvalue 1. Moreover, the matrix $\mathcal{X}'$ must have unitary determinant, thus both eigenvalues of $\mathcal{X}'$ are $\lambda_1 = \lambda_2 = 1$.

We also do not want the matrix $\mathcal{X}'$ to be the identity because otherwise $\mathcal{X} = \mathcal{X}' = \mathcal{I}$

which corresponds to the case where Bob is honest. Hence, any suitable candidate for $\mathcal{X}'$ is of the form

$$\mathcal{X}' = \begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} \begin{bmatrix} 1 & \beta \\ 0 & 1 \end{bmatrix} \begin{bmatrix} a & b \\ c & d \end{bmatrix},$$

where the matrix $\left[\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right]$ is an invertible matrix. Moreover, since we want to select specific bits of the secret $\hat{\alpha}$, let us fix β to be a power of two 2^ℓ. The previous equation implies that the matrix $\mathcal{X}'$ is of the form

$$\mathcal{X}' = \begin{bmatrix} 1 + 2^\ell cd & 2^\ell d^2 \\ -2^\ell c^2 & 1 - 2^\ell cd \end{bmatrix}.$$

Since the vector $[e_2 \ f_2]^{\mathsf{tr}}$ is an eigenvector of $\mathcal{X}'$, we have that

$$e_2 + 2^\ell cde_2 + 2^\ell d^2 f_2 = e_2 \quad \text{and} \quad -2^\ell c^2 e_2 + f_2 - 2^\ell cdf_2 = f_2.$$

Remark 1. *Note that since $\mathcal{X}'$ is of the form $\mathcal{X}' = \mathcal{I} + 2^\ell \mathcal{X}_0$ and $\begin{bmatrix} e_1 & f_1 \end{bmatrix}^{\mathsf{tr}} = -\alpha \begin{bmatrix} e_2 & f_2 \end{bmatrix}^{\mathsf{tr}} + 2^\ell \begin{bmatrix} x & y \end{bmatrix}^{\mathsf{tr}}$ (see Eq. (3.7)), then every time $\mathcal{X}' \cdot \begin{bmatrix} e_2 & f_2 \end{bmatrix}^{\mathsf{tr}} = \begin{bmatrix} e_2 & f_2 \end{bmatrix}^{\mathsf{tr}}$, we also have $\mathcal{X}' \cdot \begin{bmatrix} e_1 & f_1 \end{bmatrix}^{\mathsf{tr}} = \begin{bmatrix} e_1 & f_1 \end{bmatrix}^{\mathsf{tr}}$.*

Thus, if we assume that f_2 is invertible, the previous equations are always satisfied if the following equation holds:

$$\left(2^\ell \frac{e_2}{f_2}\right) c = -2^\ell d \mod A^2. \tag{3.8}$$

Hence, an attacker can craft values i, c, d to recover the value e_2/f_2 bit by bit. In particular, write K_i for the recovered lower i bits of e_2/f_2 (with possibly $i = 0$). Then, an attacker can recover the $i+1$-th bit by setting

$$\ell = a - 1 - i, \quad c = 1, \quad d = -K_i,$$

i.e. the matrix $\mathcal{X}'$ is

$$\mathcal{X}' = \begin{bmatrix} 1 + 2^{a-1-i}K_i & 2^{a-1-i}K_i^2 \\ -2^{a-1-i} & 1 + 2^{a-1-i}K_i \end{bmatrix}.$$

If the points created with such values i, c, d pass the validation, then we have that $2^{n-1-\ell}e_2/f_2 = 2^{n-1-\ell}K_\ell$. Hence, we derive that $K_{\ell+1} = K_\ell$ and the $\ell+1$-th bit of e_2/f_2 is *zero*. If the validation does not pass, the $\ell+1$-th bit is *one* and $K_{\ell+1} = K_\ell + 2^\ell$.

If f_2 is not invertible, then e_2 must be, and the attacker can progressively recover the value e_2/f_2. Eq. (3.8) thus becomes

$$\left(2^i \frac{e_2}{f_2}\right) d = -2^i c,$$

which means the attack can proceed as before with simply swapping the values for c and d, i.e. the matrix $\mathcal{X}'$ is transposed. Lastly, the attacker can recognize whether e_2 or f_2 is invertible by assuming that f_2 is invertible and trying to extract the first bit. This means that the attacker can make two oracle queries with $\mathcal{X} = \mathcal{I}$ and matrices $\mathcal{X}'$ given by

$$\mathcal{X}' = \begin{bmatrix} 1 & 0 \\ -2^{n-1} & 1 \end{bmatrix}, \quad \text{and} \quad \mathcal{X}' = \begin{bmatrix} 1 + 2^{n-1} & 2^{n-1} \\ -2^{n-1} & 1 + 2^{n-1} \end{bmatrix}.$$

These correspond to setting $i = n-1$, $c = 1$ and $K_1 = 0$ (for the left matrix) and $K_1 = 1$ (for the right matrix). If either query returns true, then f_2 is invertible, otherwise e_2 is invertible. The complete attack is summarized in Algorithm 1.

3.5 — Failed countermeasures and further discussion

In this section, we relate our attack to that of Galbraith and Lai which independently developed an attack on HealSIDH. We also describe some countermeasure ideas to prevent both attacks. We show that these countermeasures are also vulnerable to a more generic adaptive attack. This suggests that the approach of HealSIDH-like countermeasures to

Algorithm 1 Our attack on HealSIDH.

Compute E_B and E_{BA} as in honest HealSIDH
$\mathcal{X}'_0 \leftarrow \begin{pmatrix} 1 & 0 \\ -2^{a-1} & 1 \end{pmatrix}$ ▷ Extract whether e_2 or f_2 is invertible
$\mathcal{X}'_1 \leftarrow \begin{pmatrix} 1+2^{a-1} & 2^{a-1} \\ -2^{a-1} & 1-2^{a-1} \end{pmatrix}$
$sel \leftarrow O(E_B, I, \mathcal{X}'_0, E_{BA})$ **or** $O(E_B, I, \mathcal{X}'_1, E_{BA})$ ▷ And store it in sel
$K \leftarrow 0$
for i in $[0, \ldots, n-1]$ **do** ▷ The iterative step of the attack
    if sel **then**
        $c \leftarrow 1$; $d \leftarrow K$;
    else
        $c \leftarrow K$; $d \leftarrow 1$;
    $\mathcal{X}' \leftarrow \begin{pmatrix} 1+2^{a-i-1}cd & 2^{a-i-1}d^2 \\ -2^{a-i-1}c^2 & 1-2^{a-i-1}cd \end{pmatrix}$
    if $O(E_B, I, \mathcal{X}', E_{BA})$ **then** ▷ The oracle queries
        $K \leftarrow K + 2^i$
if sel **then** ▷ Extract α from the kernel of the dual isogeny
    $S \leftarrow 2^a \mathcal{B}_{E_A, 4^a}[K\ 1]^{\mathsf{tr}}$
else
    $S \leftarrow 2^a \mathcal{B}_{E_A, 4^a}[1\ K]^{\mathsf{tr}}$
Compute $\hat{\phi}_A$ with kernel S
Compute the dual $\hat{\hat{\phi}}_A = \phi_A$
Express a generator of $\ker \phi_A$ as $2^a \mathcal{B}_{E_0, 4^a}[1\ \alpha]^{\mathsf{tr}}$
return α

GPST attacks is likely to be vulnerable in general. We end this section with a discussion on the translation of theses countermeasures and attacks to the genus-two setting.

3.5.1 – Comparison with the Galbraith–Lai attack

We discuss the similarities and differences between our attack and the Galbraith-Lai [GL22] attack. Using the HealSIDH key exchange oracle introduced in section 3.4.1, the Galbraith–Lai attack can be summarized as follows:

1. Recover the largest power 2^j of two dividing the secret integer α. You may use the the attack described in Section 3.4.2 or that described in [GL22].

2. Recursively recover the remaining bits of α by querying the key exchange oracle

with

$$\mathcal{X} = \begin{bmatrix} 1 + 2^{2a-i+j-1} & -\hat{\alpha}_l 2^{2a-i-1} \\ -\alpha_l 2^{2a-i+j-1} & 1 + 2^{2a-i+j-1} \end{bmatrix} \quad \text{and} \quad \mathcal{X}' = \mathcal{I}$$

where $\alpha_l = \alpha \mod 2^{i-j}$ is the previously recovered part of α.

Both attacks are adaptive and recover the secret isogeny after $\mathcal{O}(\log p)$ queries to the key exchange oracle. They are, in some sense, orthogonal to each other, as the Galbraith-Lai attack directly recovers the secret α used in the generation of the kernel of the secret isogeny ϕ_A, while ours recovers the corresponding secret integer for a kernel generator of the dual isogeny $\hat{\phi}_A$ instead, that is e_2/f_2 or f_2/e_2 depending on whether f_2 or e_2 is invertible. Moreover, our attack only modifies $\mathcal{X}'$, while the Galbraith–Lai attack only modifies $\mathcal{X}$. Despite these similarities, they turn out to be quite distinct. In fact, we have noticed that one could counter one without automatically countering the other. This suggests the countermeasures discussed in the next section.

3.5.2 – An attempt at better countermeasures

One could try to counter the attack presented in Section 3.4 by basing the validation method in HealSIDH on Eq. (3.1) instead of Eq. (3.2). Suppose that the validation consists in verifying that the following equation holds:

$$[A]\phi_B = \widehat{\phi'_A} \circ \phi'_B \circ \phi_A.$$

In terms of matrices, this translates to

$$A\mathcal{M} = \mathcal{M}^*_{\psi'_A,A} \cdot \mathcal{M}' \cdot \mathcal{M}_{\phi_A,A}, \tag{3.9}$$

where $\mathcal{M}^*_{\psi'_A,A} = \mathcal{M}_{\widehat{\psi'_A},A}$ is the matrix associated with ψ'_A, which is also the pseudo-inverse of $\mathcal{M}_{\psi'_A,A}$, i.e. their product is the identity matrix scaled by A.[2] Alternatively, it is possible

[2]We deal with pseudo-inverses because the matrix $\mathcal{M}_{\psi'_A,A}$ is not invertible, since it represents the action of an isogeny whose degree is not coprime with the order of the torsion basis.

to consider the following equation:

$$\phi'_A \circ \phi_B \circ \widehat{\phi_A} = [A]\phi'_B$$

which leads to the following matrix check:

$$A\mathcal{M}' = \mathcal{M}_{\psi_A,A} \cdot \mathcal{M} \cdot \mathcal{M}^*_{\phi_A,A}. \tag{3.10}$$

Translating the analysis made in Section 3.4.1 in the context of Eq. (3.9), the validation checks that both the determinants of $\mathcal{X}$ and $\mathcal{X}'$ are 1 and that

$$\mathcal{M}^*_{\phi_A,A} \cdot \mathcal{X}' \cdot \mathcal{M}_{\phi_A,A} = A\mathcal{X}. \tag{3.11}$$

In the context of Eq. (3.10), the validation checks that both the determinants of $\mathcal{X}$ and $\mathcal{X}'$ are 1 and that

$$\mathcal{M}_{\phi_A,A} \cdot \mathcal{X} \cdot \mathcal{M}^*_{\phi_A,A} = A\mathcal{X}', \tag{3.12}$$

Clearly, using Eq. (3.11) in the key validation would counter the Galbraith-Lai [GL22] attack (since it implies $A\mathcal{I} = A\mathcal{X}$ because $\mathcal{X}' = \mathcal{I}$), while using Eq. (3.12) would counter the attack presented in Section 3.4.2 (since it implies $A\mathcal{X}' = A\mathcal{I}$ because $\mathcal{X} = \mathcal{I}$).

What is less clear is whether using Eq. (3.11) in the key validation counters the attack presented in Section 3.4, and whether using Eq. (3.12) counters that of Galbraith-Lai [GL22]. Experimental evidence suggests so. Nevertheless, in the following section, we sketch a general adaptive attack on this countermeasure idea where Eq. (3.11) is used in the validation. A similar attack applies when Eq. (3.12) is used instead.

3.5.3 – Extending the attack

We now demonstrate that the potential repairs suggested in the previous section, even though they defeat the attack in Section 3.4 and in [GL22], are still vulnerable to adaptive

attacks.

Firstly, since $\hat{\phi}_A \circ \phi_A = [A]$, then

$$\mathcal{M}_{\hat{\phi}_A, A^2} \cdot \mathcal{M}_{\phi_A, A^2} = A \cdot \mathcal{I}.$$

That is, $\mathcal{M}_{\hat{\phi}_A, A^2}$ is a pseudo-inverse $\mathcal{M}^*_{\phi_A, A^2}$ of $\mathcal{M}_{\phi_A, A^2}$. Moreover, we can write $\mathcal{M}_{\phi_A, A^2}$ as

$$\mathcal{M}_{\phi_A, A^2} = \mathcal{P} \cdot \begin{bmatrix} 1 & 0 \\ 0 & A \end{bmatrix} \cdot \mathcal{P}^{-1},$$

where $\mathcal{P}$ is an invertible matrix. In this case, we have that

$$\mathcal{M}_{\hat{\phi}_A, A^2} = \mathcal{M}^*_{\phi_A, A^2} = \mathcal{P} \cdot \begin{bmatrix} A & 0 \\ 0 & 1 \end{bmatrix} \cdot \mathcal{P}^{-1}.$$

Suppose that the validation method uses Eq. (3.11). We set $\mathcal{X} = \mathcal{I}$, as in the previous attack, and write

$$\mathcal{X}' = \mathcal{I} + 2^{2a-i}\Delta.$$

Then Eq. (3.11) translates to

$$\mathcal{P} \cdot \begin{bmatrix} A & 0 \\ 0 & 1 \end{bmatrix} \cdot \mathcal{P}^{-1} \cdot \mathcal{X}' \cdot \mathcal{P} \cdot \begin{bmatrix} 1 & 0 \\ 0 & A \end{bmatrix} \cdot \mathcal{P}^{-1} \equiv A\mathcal{I} \pmod{A^2},$$

that is

$$\begin{bmatrix} A & 0 \\ 0 & 1 \end{bmatrix} \cdot \mathcal{P}^{-1} \cdot \mathcal{X}' \cdot \mathcal{P} \cdot \begin{bmatrix} 1 & 0 \\ 0 & A \end{bmatrix} \equiv A\mathcal{I} \pmod{A^2}.$$

Plugging in $\mathcal{X}' = \mathcal{I} + 2^i\Delta$, we get

$$2^i \begin{bmatrix} A & 0 \\ 0 & 1 \end{bmatrix} \cdot \mathcal{P}^{-1} \cdot \Delta \cdot \mathcal{P} \cdot \begin{bmatrix} 1 & 0 \\ 0 & A \end{bmatrix} \equiv 0 \pmod{A^2}.$$

At this stage, it is not clear yet whether an attack that recovers a bit of the secret per interaction is possible. Instead, we consider a generalized attack where the attacker mantains a set $\mathcal{K}_\ell$ of possible partial keys. Each $K \in \mathcal{K}_\ell$ passes validation for the first ℓ bits, i.e. when the matrix Δ is scaled by $2^{n-\ell-1}$. At each step, the attacker queries the oracle with K and $K + 2^\ell$ with $i = n - \ell - 2$, i.e. testing the $\ell + 1$ bits. Any new key that passes validation is then added to the set $\mathcal{K}_{\ell+1}$. Eventually, when the attacker recovers $\mathcal{K}_n$, i.e. the set of possible full keys, they can simply test each key to find the correct one.

The attacker can then build an attack using matrices Δ of the form

$$\Delta = \begin{bmatrix} -1 & K^{*2} - 2K^* \\ 1 & 1 \end{bmatrix},$$

where $K^* = K$ or $K^* = K + 2^\ell$. While the number of possible solutions, i.e. the size of $\mathcal{K}$ doubles at some stages, experimentally the number of doubling remains limited, and the attack appears to be linear in the security parameter. In particular, we ran the attack against setups with 13 bits, 20 bits and 40 bits of security for 1000 repetitions each. We obtained that the number of average oracle queries scales linearly in the security parameter.

Note that the attack is redundant and makes more oracle queries than required, since

$$2^{n-\ell-1}(K^2 - 2K) = 2^{n-\ell-1}((K + 2^\ell)^2 - 2(K + 2^\ell)) \qquad (\mathrm{mod}\ 2)^n$$

Thus, one may hope that it is possible to reduce the number of oracle queries required to recover the secret key, and possibly reduce (or avoid altogether) the number of partial key candidates. Further work is required to assess and develop these possibilities, but the attack outlined in this paragraph shows that the countermeasures of Section 3.5.2 are unlikely to offer a solution.

3.5.4 – Perspectives

Finding key validation methods for SIDH keys seems to remain a difficult problem. In [UJ18], it is shown that the problem of validating an SIDH public key can be reduced to more standard problems in isogeny-based cryptography that are considered to be hard. This suggests that it is probably impossible to devise new countermeasures without revealing any additional information. Hence, new countermeasures will necessarily require the two parties to exchange additional information.

The main difficulty in developing new countermeasures of this type seems to be the fact that the validation procedure relies on some secret (and static) value, possibly the secret key α. If not, the attacker can most likely cheat by knowing how the validation is computed. However, the attacker is free to manipulate the torsion information. There is a close connection between torsion points that have been manipulated by adding points of order 2^i and the lower i bits of the secret. It seems that most possible countermeasures would still be vulnerable to attacks where the attacker recovers the secret information used to validate the public key bit by bit. Once that secret is recovered, even if different from the secret key, the attacker can probably cheat since they know the validation secret. Thus, any successful countermeasure would necessarily need to break this connection in such a way that manipulating the torsion points with points of order 2^i would affect more bits of the secret value than just the lower i bits.

3.6 — Generalizing to genus two

There have recently been efforts to translate the SIDH protocol which uses isogenies of elliptic curves, i.e. principally polarised supersingular abelian varieties of dimension one, into a higher-dimensional setting. In this section, we propose a translation of HealSIDH to the genus-two setting, and we discuss the difficulties in adapting the attack on HealSIDH to the newly proposed protocol. This may give hope that a genus-two version of HealSIDH is more resistant against adaptive attacks.

The Genus-Two SIDH (G2SIDH) key exchange was introduced by Flynn and Ti [FT19] and is a natural generalisation using $(2,2)$- and $(3,3)$-isogenies between principally polarised supersingular abelian surfaces. The most current version of G2SIDH, however, uses only a subset of these varieties; in particular, Castryck, Decru, and Smith [CDS20] restrict the varieties considered to the principally polarised superspecial abelian surfaces (PPSSASs), and Kunzweiler, Ti, and Weitkämper [KTW22] suggest the use of symplectic torsion bases as well as a restriction of the keyspace to improve efficiency.

3.6.1 – G2SIDH

We give a brief description of the key exchange scheme here using the formalised notation introduced above. The protocol is very similar to the one-dimensional, 'standard' SIDH scheme.

Consider a prime of the form $p = 2^a \cdot 3^b \cdot f - 1$ where $2^{e_A} \approx 3^{e_B}$ and f is a small cofactor not divisible by 2 or 3. A starting PPSSAS J_0 is then chosen by taking a random walk in the $(2,2)$-isogeny graph originating from the Jacobian of the hyperelliptic curve defined by $y^2 = x^6 + 1$ over $\mathbb{F}_{p^2}$, and torsion bases $\mathcal{B}_{J_0,2^a}$ of $J_0[2^a]$ and $\mathcal{B}_{J_0,3^b}$ of $J_0[3^b]$ are fixed with all individual points defined over $\mathbb{F}_{p^2}$. Note here that $J_0[N] \cong (\mathbb{Z}/N\mathbb{Z})^4$ for some prime power N. These torsion bases are required to be symplectic with respect to the Weil pairing for the relevant torsion subgroup. This means for the N-torsion if $\mathcal{B}_{J_0,N} = \begin{bmatrix} P_1 & P_2 & Q_1 & Q_2 \end{bmatrix}$, we require $e_N(P_1,P_2) = e_N(Q_1,Q_2) = \mu_N^0 = 1$ and $e_N(P_i,Q_j) = \mu_N^{\delta_{ij}}$ for μ_N a primitive N-th root of unity, $\delta_{ij} = 1$ if $i = j$, and $\delta_{ij} = 0$ if $i \neq j$.

To generate a secret key, each of the parties chooses a secret maximal isotropic subgroup G of their respective J_0-torsion which then corresponds to an isogeny ϕ_N with kernel G and codomain J_N. Focusing on the restricted keyspace suggested in [KTW22], this kernel can be determined by two generators of full order in the torsion subgroup, given as linear combinations of the symplectic basis points.

Setup. Let $A = 2^a$, $B = 3^a$ such that $p = ABf - 1$ is a prime. Let J_0 be a superspecial abelian surface defined over $\mathbb{F}_{p^2}$. The public parameters are p, J_0, A and B, so that each

party can compute canonically generated symplectic torsion bases $\mathcal{B}_{J_0,A}$, and $\mathcal{B}_{J_0,B}$. While efficiently generating a symplectic base remains an open problem, [KTW22, Algorithm 1] provides a possibly inefficient method to do so. Since such bases are generated once as part of the protocol parameters, the inefficiency of the method does not affect the runtime of the protocol.

KeyGeneration. Alice picks three random integers $0 \leq \alpha_1, \alpha_2, \alpha_3 \leq 2^a - 1$ and computes the (A, A)-isogeny $\phi_A : J_0 \to J_A$ of kernel given by

$$\mathcal{A} := \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ \alpha_1 & \alpha_2 \\ \alpha_2 & \alpha_3 \end{bmatrix}$$

together with $\mathcal{M}_{\phi_A,B}$, defined analogously to the matrices defined in Section 3.3. Her secret key is $\mathcal{A}$ and her public key is $(J_A, \mathcal{M}_{\phi_A,B})$. Bob proceeds analogously by picking random integers β_1, β_2 and β_3 from $\{0, \ldots, 3^b - 1\}$ and computing the B-isogeny $\phi_B : J_0 \to J_B$ of kernel generated by

$$\mathcal{B} := \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ \beta_1 & \beta_2 \\ \beta_2 & \beta_3 \end{bmatrix}$$

together with $\mathcal{M}_{\phi_B,A}$. His secret key is $\mathcal{B}$ and his public key is $(J_B, \mathcal{M}_{\phi_B,A})$.

KeyExchange. Given $(J_B, \mathcal{M}_{\phi_B,A})$, Alice computes the (A, A)-isogeny $\phi'_A : J_B \to J_{BA}$ of kernel generated by $\mathcal{M}_{\phi_B,A} \cdot \mathcal{A}$. Given $(J_A, \mathcal{M}_{\phi_A,B})$, Bob computes the (B, B)-isogeny $\phi'_B : J_A \to J_{AB}$ of kernel generated by $\mathcal{M}_{\phi_A,B} \cdot \mathcal{B}$. The abelian surfaces J_{AB} and J_{BA} are isomorphic as principally polarised abelian surfaces and hence their Igusa isomorphism invariants can be used as the shared secret.

Like its genus-one counterpart, G2SIDH is also vulnerable to GPST-style adaptive attacks, as shown by Kunzweiler, Ti, and Weitkämper [KTW22].

3.6.2 – HealSIDH with PPSSAS

As in the elliptic curve SIDH setting, it is possible to introduce a second round to the key exchange protocol which allows both parties to verify the correctness of the other's previously provided public key, i.e. torsion point information, via checking Eq. (3.2).

The genus-two variant of HealSIDH thus works as follows:

Setup. Let $A = 2^a$, $B = 3^a$ such that $p = A^2B^2f - 1$ is a prime. Let J_0 be a principally polarised superspecial abelian surface defined over $\mathbb{F}_{p^2}$. The public parameters are p, J_0, as well as A and B such that canonically generated symplectic torsion bases $\mathcal{B}_{J_0,A^2}$, and $\mathcal{B}_{J_0,B^2}$ can be computed. As in the G2SIDH cases, such bases can be computed, as part of the parameter generation for the protocol, via [KTW22, Algorithm 1].

KeyGeneration. Alice chooses her secret scalars as in Section 3.6.1 but uses an altered matrix $\mathcal{A} = \begin{bmatrix} 2^a & 0 & 2^a\alpha_1 & 2^a\alpha_2 \\ 0 & 2^a & 2^a\alpha_2 & 2^a\alpha_3 \end{bmatrix}^T$ to give her kernel generators. She also computes the matrices $\mathcal{M}_{\phi_A,B^2}$ and $\mathcal{M}_{\phi_A,A^2}$. Bob proceeds analogously to find his secret matrix $\mathcal{B}$ and compute $\phi_B : J_0 \to J_B$, $\mathcal{M}_{\phi_B,A^2}$ and $\mathcal{M}_{\phi_B,B^2}$.

Interaction. After receiving Bob's public key $(J_B, \mathcal{M}_{\phi_B,A^2})$, Alice can compute the 2^a-isogeny $\phi'_A : J_B \to J_{BA}$ with kernel given by $\mathcal{M}_{\phi_B,A^2} \cdot \mathcal{A}$ together with the related matrices $\mathcal{M}_{\phi'_A,B^2}$ and $\mathcal{M}_{\phi'_A,A^2}$, of which she sends $\mathcal{M}_{\phi'_A,B^2}$ to Bob. Bob proceeds analogously to obtain J_{AB}, $\mathcal{M}_{\phi'_B,A^2}$ and $\mathcal{M}_{\phi'_B,B^2}$. He sends $\mathcal{M}_{\phi'_B,A^2}$ to Alice.

KeyExchangecompletion. Upon receiving both matrices $\mathcal{M}_{\phi_B,A^2}$ and $\mathcal{M}_{\phi'_B,A^2}$ from Bob, Alice first checks whether both produce symplectic bases on $J_B[A^2]$ and $J_{AB}[A^2]$ with respect to the correct primitive A^2-th root, say $\mu_{A^2}^{deg\phi_B} = \mu_{A^2}^{deg\phi'_B} = \mu_{A^2}^B$, respectively. Alice further confirms that the equality $\mathcal{M}_{\phi'_B,A^2} \cdot \mathcal{M}_{\phi_A,A^2} = \mathcal{M}_{\phi'_A,A^2} \cdot \mathcal{M}_{\phi_B,A^2}$ holds. If both checks are successful, Alice uses the isomorphism invariants of J_{BA} as the shared secret. Bob proceeds analogously.

Note here that the new kernel generators given by $\mathcal{A}$ and $\mathcal{B}$ do generate valid maximal A- and B-isotropic subgroups of the A- and B-torsion of J_0, respectively, if we consider

the bases obtained via $A \cdot \mathcal{B}_{J_0,A^2}$ and $B \cdot \mathcal{B}_{J_0,B^2}$ to be the canonical symplectic bases of $J_0[A]$ and $J_0[B]$, respectively.

It is possible to generalise some of the attack strategy from Section 3.4. For example, it is possible to define a key exchange oracle in much the same way as for the elliptic curve HealSIDH case. To comply with the technical detail the additional dimension adds, this could be formulated as

$$\mathcal{O}(J_B, \mathcal{M}, \mathcal{M}', J') = 1 \iff \begin{cases} J' \cong J_{BA}, \\ \mathcal{M}, \mathcal{M}' \text{ define symplectic bases of } J_B[A^2], J'[A^2], \\ \mathcal{M}' \cdot \mathcal{M}_{\phi_A,A^2} = \mathcal{M}_{\phi'_A,A^2} \cdot \mathcal{M} \end{cases}$$

and $\mathcal{O}(J_B, \mathcal{M}, \mathcal{M}', J') = 0$ otherwise.

Note here that both the second and third condition for the oracle to output 1 complicate finding matrices $\mathcal{X}, \mathcal{X}' \in GL_4(\mathbb{Z}/A^2\mathbb{Z})$ to manipulate Bob's torsion point information with. Passing the second condition, i.e. that $\mathcal{M}$ and $\mathcal{M}'$ define symplectic bases of $J_B[A^2]$ and $J'[A^2]$ respectively, when the honestly generated matrices are transformed using X and X', requires the matrices $\mathcal{X}$ and $\mathcal{X}'$ to correspond to symplectic transformations. Unfortunately, making this explicit requires the matrices $\mathcal{X}$ and $\mathcal{X}'$ to satisfy further conditions modulo A^2 since having determinant equal to 1 is not a sufficient condition for 4×4 matrices to be symplectic. In particular, we can fix a non-singular, skew-symmetric matrix Ω, and we then require $\mathcal{X}$ and $\mathcal{X}'$ to fulfill

$$\mathcal{X}^T \Omega \mathcal{X} = \Omega \quad \text{and} \quad (\mathcal{X}')^T \Omega \mathcal{X}' = \Omega \quad \text{for } \Omega = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \end{bmatrix}.$$

Satisfying the third oracle condition requires analogous reasoning to the genus-one case where we equivalently require the equality $\mathcal{X}' \cdot \mathcal{M}_{\phi_A,A^2} = \mathcal{M}_{\phi_A,A^2} \cdot \mathcal{X}$ to hold.

However, it is not possible to define and recover a ratio $\hat{\alpha}$ due to the larger matrix size and since the choice of admissible symplectic matrices makes it harder to target individual entries in $\mathcal{M}_{\phi_A,A^2}$ for recovery. Furthermore, learning the image of a single (possibly specific) A^2-torsion point under the isogeny ϕ_A does not provide enough information for an attacker to reconstruct the original isogenies since the involved kernels are of rank at least two when PPSSAS are used.

We leave overcoming these technical issues and finding explicit matrices $\mathcal{X}$ and $\mathcal{X}'$ launching an attack on genus-two HealSIDH for future work.

Chapter 4

Cryptanalysis of an oblivious PRF from supersingular isogenies

If you try and take a cat apart to see how it works, the first thing you have on your hands is a non-working cat.

— D. Adams, impromptu speech at Digital Biota Conference

This chapter is a verbatim reproduction of the following paper:

Andrea Basso, Péter Kutas, Simon-Philipp Merz, Christophe Petit, and Antonio Sanso. "Cryptanalysis of an Oblivious PRF from Supersingular Isogenies". In: *ASIACRYPT 2021, Part I*. ed. by Mehdi Tibouchi and Huaxiong Wang. Vol. 13090. LNCS. Springer, Heidelberg, Dec. 2021, pp. 160–184. DOI: 10.1007/978-3-030-92062-3_6

I contributed to the development of the proposed attacks and their implementation.

Abstract: We cryptanalyse the SIDH-based oblivious pseudorandom function from supersingular isogenies proposed at Asiacrypt'20 by Boneh, Kogan and Woo. To this end, we give an attack on an assumption, the auxiliary one-more assumption, that was introduced by Boneh et al. and we show that this leads to an attack on the oblivious PRF itself. The attack breaks the pseudorandomness as it allows adversaries to evaluate the OPRF without further interactions with the server after some initial OPRF evaluations

and some offline computations. More specifically, we first propose a polynomial-time attack. Then, we argue it is easy to change the OPRF protocol to include some countermeasures, and present a second subexponential attack that succeeds in the presence of said countermeasures. Both attacks break the security parameters suggested by Boneh et al. Furthermore, we provide a proof of concept implementation as well as some timings of our attack. Finally, we examine the generation of one of the OPRF parameters and argue that a trusted third party is needed to guarantee provable security.

4.1 — Introduction

An oblivious pseudorandom function (OPRF) is a two-party protocol between a client and a server that computes a pseudorandom function (PRF) on a client's input with the server's key. At the end, the server does not learn anything about the client's input or the output of the function and the client learns the evaluation of the OPRF but nothing about the server's key. In particular, a client should not be able to compute the OPRF on any input without the server's participation.

Moreover, a *verifiable* oblivious pseudo random function (VOPRF) is an OPRF, where a server commits to some key and the client is ensured that the server used this key to evaluate the OPRF. In particular, the client is guaranteed that a server does not change their secret key across different executions of the protocol.

Oblivious pseudorandom functions are an important building block in many cryptographic applications. They can be used for private set intersection [JL09], which in turn has many applications such as private contact discovery for messaging services [DRRT18] or checking for compromised credentials [LPASC+19]. Further applications of (V)OPRFs include password-authenticated key exchange [JKX18], password-management systems [ECSJR15], adaptive oblivious transfer [JL09], password-protected

secret sharing [JKK14] and privacy-preserving CAPTCHA systems [DGSTV18].

Apart from their theoretical relevance in cryptography, OPRFs have had significant real-world impact recently. The password-authenticated key exchange OPAQUE [JKX18] which is built on an OPRF is intended for use in TLS 1.3 [SKFB21].

The privacy-preserving authorisation mechanism known as Privacy Pass by Davidson et al. [DGSTV18] is also based entirely on the security of a VOPRF. Privacy Pass is currently used at scale by Cloudflare. There is an ongoing effort to standardise OPRFs at the Crypto Forum Research Group (CFRG) [DSW19].

Generic techniques from two-party computation and zero-knowledge proofs can be used to construct verifiable OPRFs. However, the resulting protocols might be inefficient. Therefore, all of the real-world use-cases of (V)OPRFs are currently instantiated with performant (V)OPRFs which are based on classical security assumptions. Practical constructions are currently based either on the hardness of the decisional Diffie-Hellman problem, called DH-OPRF [JKK14], or they are derived from RSA blind signatures [Cha82; DSW19].

For quantum-secure OPRFs, there are only few proposals. Indeed, only three such solutions appear in the literature to date. In 2019, Albrecht et al. proposed a lattice-based VOPRF [ADDS21] based on the ring learning with errors problem and the short integer solution problem in one dimension. Seres et al. constructed an OPRF based on the shifted Legendre symbol problem [SHB21] and Boneh et al. presented two isogeny-based (V)OPRFs at ASIACRYPT 2020 [BKW20].

Isogeny-based cryptography is one of the branches of post-quantum cryptography that are currently being explored. The particularly small key sizes required by isogeny-based cryptosystems make them very attractive in some areas of information security. Isogeny-based cryptography was first proposed by Couveignes in 1997 [Cou06]. However, his ideas were not published at the time and they were independently rediscovered by Rostovtsev and Stolbunov [RS06]. The idea of Couveignes and Rostovtsev-Stolbunov (CRS) was to build a Diffie-Hellman type key exchange using the class group of the endomorphism

ring of ordinary elliptic curves. However, neither of the suggested schemes was efficient enough to be considered practical. Meanwhile, supersingular elliptic curves were first used in cryptography by Charles, Lauter and Goren [CLG09] to build a hash function.

Jao and De Feo took a different approach to isogeny-based cryptography when they introduced the *supersingular isogeny Diffie-Hellman* (SIDH) key exchange [JD11]. Instead of computing class group actions as in the case of CRS, Jao and De Feo use the following observation. Two subgroups of an elliptic curve of coprime cardinality are only intersecting at the point at infinity. Independent of the order in which two such subgroups are divided out of an elliptic curve, the resulting curve will be equal up to isomorphism. The only isogeny-based cryptosystem submitted to NIST's ongoing post-quantum standardization process is the SIDH-based candidate SIKE [JACCD+20] which has been selected as one of the alternate finalists.

Later, the idea of CRS-type schemes was resurrected, when Castryck et al. adapted it to supersingular elliptic curves and managed to eliminate most of its performance issues [CLMPR18]. The resulting scheme is called CSIDH.

In their ASIACRYPT 2020 paper [BKW20], Boneh et al. propose an augmentable commitment framework that can be used to build an OPRF and is instantiated with both an SIDH-based scheme that can be made verifiable, and with a CSIDH-based one. The SIDH-based variant relies on the hardness of the decisional supersingular isogeny problem, a standard assumption in the area, and a novel 'one-more' isogeny assumption.

Our contributions In this paper, we cryptanalyse the SIDH-based 'one-more' assumption introduced by Boneh, Kogan and Woo. We first give multiple variants of an attack on the assumption itself. A first variant leads to a polynomial-time attack against the proposed SIDH-based OPRF protocol. We then argue that a simple modification of the (V)OPRF protocol prevents such an attack. Then, we show that a second variant of the attack leads to an attack on the protocol even in the presence of those countermeasures. This attack has a subexponential complexity, but there appear to be no simple countermeasures.

Developing countermeasures is left as an open problem. As a result of our attack, the parameters suggested by Boneh et al. fall short of their estimated security level.

The attacks on the OPRF allow malicious clients to evaluate the OPRF on arbitrary inputs after some initial queries to the server, without even interacting with the server any further. This breaks the pseudorandomness property of the OPRF and could lead to significant attacks on OPRF-based protocols. In the context of private set intersection based on oblivious PRFs, the proposed attack allows the attacker to brute-force the other party's set elements and break the privacy requirement. In the Privacy Pass protocol used to guarantee privacy-preserving CAPTCHAs, our attack allows the attacker to generate unlimited tokens, thus avoiding solving CAPTCHAs and fully breaking the security of the system.

Furthermore, we discuss how one of the parameters of the SIDH-based OPRF by Boneh et al. is generated and which party should compute it. We argue there are security implications if the server, the client or any third party maliciously generates this parameter. The client or a third party can introduce a backdoor through this parameter to recover the secret key of the server, whereas if the server is malicious, they can break the supersingular-isogeny collision assumption on which Boneh et al.'s security proofs are built. We suggest that a trusted setup may be needed to guarantee provable security.

Finally, we want to emphasise that the CSIDH-based OPRF proposal by Boneh et al. is not affected by our attacks.

Outline. In Section 4.2, we introduce some background on isogeny-based cryptography, the security properties of (verifiable) oblivious PRFs and Boneh et al.'s construction. The attacks against the 'one-more' assumption are presented in Section 4.3, while their application to the OPRF protocol by Boneh et al. is discussed in Section 4.4. We present our implementation of the attack and discuss its results in Section 4.5. In Section 4.6, we argue that a trusted setup should be used for the OPRF and briefly sketch two attacks that follow a lack of trusted setup before concluding the paper in Section 4.7.

4.2 — Preliminaries

In this section we introduce the necessary mathematical background on isogenies and the SIDH key exchange, we summarize the security properties of OPRFs and we briefly recall Boneh et al.'s OPRF construction [BKW20].

4.2.1 – Mathematical background on isogenies

Let $\mathbb{F}_q$ be a finite field of characteristic p. In the following, we assume $p \geq 3$ and therefore an elliptic curve E over $\mathbb{F}_q$ can be defined by its short Weierstrass form

$$E(\mathbb{F}_q) = \{(x, y) \in \mathbb{F}_q^2 \mid y^2 = x^3 + Ax + B\} \cup \{\mathcal{O}_E\}$$

with $A, B \in \mathbb{F}_q$ such that the discriminant is non-zero and $\mathcal{O}_E$ denotes the point $(X : Y : Z) = (0 : 1 : 0)$ on the associated projective curve $Y^2Z = X^3 + AXZ^2 + BZ^3$. The *j-invariant* of an elliptic curve is $j(E) = 1728\frac{4A^3}{4A^3+27B^2}$.

A non-constant rational map $\phi : E_1 \to E_2$ between two elliptic curves is an *isogeny* if it sends the point at infinity of E_1 to the point at infinity of E_2. Equivalently, an isogeny is a rational map which is also a group homomorphism. Thus an isogeny is the natural morphism of the category of elliptic curves. An isogeny ϕ induces a field extension between the function fields of E_1 and E_2. The degree of this extension is the degree of the isogeny. We call an isogeny *separable* if this field extension is separable. The kernel of a separable isogeny as a group homomorphism is finite and is equal to the degree of the isogeny. If $\phi : E_1 \to E_2$ is an isogeny of degree d, then there exists a unique isogeny $\hat{\phi}$ of degree d such that $\phi \circ \hat{\phi} = [d]$, where $[d]$ denotes multiplication by d. The isogeny $\hat{\phi}$ is called the *dual isogeny* of ϕ. An isomorphism of elliptic curves is an isogeny of degree 1 and there is an isomorphism of curves $f : E_0 \to E_1$ if and only if $j(E_0) = j(E_1)$.

An *endomorphism* of E is an isogeny from E to itself. Endomorphisms of E form a ring under composition and addition denoted by $\mathrm{End}(E)$. The endomorphism ring of

an elliptic curve over a finite field is either an order in an imaginary quadratic field (in which case the curve is called *ordinary*) or a maximal order in the quaternion algebra ramified at infinity and p (in which case the curve is called *supersingular*).

The j-invariant of any supersingular elliptic curve defined over $\mathbb{F}_q$ lies in $\mathbb{F}_{p^2}$.

For a thorough introduction to elliptic curves and isogeny-based cryptography, we refer to Silverman [Sil86] and De Feo [Feo17], respectively.

4.2.2 – SIDH

We briefly recall the *supersingular isogeny Diffie-Hellman* key exchange introduced by Jao and De Feo [JD11].

Let E_0 be a supersingular elliptic curve defined over $\mathbb{F}_{p^2}$, where p is a prime of the form $f \cdot N_1 N_2 \pm 1$. Here $f \in \mathbb{Z}$ is a small cofactor and N_1, N_2 are two coprime smooth integers (e.g. a power of 2 and 3 respectively). Furthermore, fix two bases P_A, Q_A and P_B, Q_B such that $\langle P_A, Q_A \rangle = E_0[N_1]$ and $\langle P_B, Q_B \rangle = E_0[N_2]$. To agree on a secret key over an insecure channel, Alice and Bob proceed as follows:

1. Alice chooses a random cyclic subgroup of $E_0[N_1]$ generated by a point of the form $A = P_A + [x_A]Q_A$ as her secret. Similarly, Bob chooses his secret as $\langle B \rangle := \langle P_B + [x_B]Q_B \rangle \subset E_0[N_2]$.

2. Then, Alice and Bob compute their secret isogeny $\varphi_A : E_0 \to E_0/\langle A \rangle$ and $\varphi_B : E_0 \to E_0/\langle B \rangle$, respectively.

3. Alice sends the curve $E_A := E_0/\langle A \rangle$ and the points $\varphi_A(P_B), \varphi_A(Q_B)$ to Bob. Mutatis mutandis, Bob sends $E_B := E_0/\langle B \rangle, \varphi_B(P_A)$ and $\varphi_B(Q_A)$ to Alice.

4. Both Alice and Bob can compute the shared secret curve $E_{AB} := E_0/\langle A, B \rangle$ up to isomorphism as

$$E_{AB} \cong E_B/\langle \varphi_B(P_A) + [x_A]\varphi_B(Q_A) \rangle \cong E_A/\langle \varphi_A(P_B) + [x_B]\varphi_A(Q_B) \rangle.$$

Since isomorphic curves have the same j-invariant, Alice and Bob use $j(E_{AB})$ as their shared secret.

4.2.3 – Security properties of (V)OPRFs

In the following, we will call a function $\mu : \mathbb{N} \to \mathbb{R}$ *negligible* if for every positive polynomial $\mathsf{poly}(\cdot)$ there exists an integer $N_{\mathsf{poly}} > 0$ such that for all $x > N_{\mathsf{poly}}$, we have $|\mu(x)| < 1/\mathsf{poly}(x)$.

The security properties of an oblivious pseudorandom function (OPRF) include those of a standard pseudorandom function (PRF).

Definition 3. *Let $F : K \times X \to Y$ be an efficiently computable function. F is a* pseudorandom function *(PRF) if for all probabilistic polynomial-time distinguishers D, there is a negligible function negl such that*

$$\mathbb{P}[D^{F(k,\cdot)}(1^n) = 1] - \mathbb{P}[D^{f(\cdot)}(1^n) = 1] \leq \mathsf{negl}(n),$$

where the first probability is taken over uniform choices of $k \in \{0,1\}^n$ and the randomness of D, and the second probability is taken over uniform choices of functions $f : X \to Y$ and the randomness of D.

A consequence of pseudorandomness is that one cannot compute a new evaluation of $F(k,\cdot)$ from existing evaluations. However, our attack on Boneh et al.'s OPRF will allow adversaries to evaluate $F(k,\cdot)$ on arbitrary inputs after some initial evaluations.

Furthermore, an OPRF is oblivious in the following sense.

Definition 4 ([FIPR05])**.** *Let $F : K \times X \to Y$ be a PRF. A protocol between a client with input $x \in X$ and a server with key $k \in K$ is called* oblivious *PRF, if the client learns $F(k,x)$ and nothing else and the server learns nothing about x or $F(k,x)$ at the end of the protocol.*

In particular, the server will learn nothing about the input x of the client and the client will learn nothing about the server's key k.

Additionally, an OPRF can be verifiable.

Definition 5. *An OPRF is said to be* verifiable *if the evaluation y that the client obtains at the end of the protocol is correct, i.e. if it satisfies $y = F(k, x)$, where $x \in X$ is the client's input and $k \in K$ is the server's private key.*

In practice, verifiability is ensured by the server committing to a key k prior to the execution of the verifiable OPRF (VOPRF) and providing a zero-knowledge proof that the VOPRF execution uses the same key as the committed value.

4.2.4 – An isogeny-based OPRF by Boneh, Kogan and Woo

We provide a simplified description of Boneh et al.'s OPRF based on the SIDH key exchange protocol.

Let λ be the security parameter and let $p = fN_{\mathsf{K}}N_{\mathsf{M}}N_{\mathsf{V}}N_{\mathsf{R}}N_{\mathsf{S}} - 1$ be a prime where $f \in \mathbb{Z}$ is a small cofactor and N_i are powers of distinct small primes such that $N_{\mathsf{K}}, N_{\mathsf{M}}, N_{\mathsf{V}}, N_{\mathsf{R}}$ are roughly of size $2^{5\lambda/2}$ and $N_{\mathsf{S}} \approx 2^{2\lambda}$. To prevent an attack by Merz et al. [MMP20], the factors $N_{\mathsf{K}}, N_{\mathsf{M}}, N_{\mathsf{V}}, N_{\mathsf{R}}$ are of size $2^{5\lambda/2}$ instead of the more common size $2^{2\lambda}$ in the SIDH setting. Moreover, let $H_1 : \{0,1\}^* \to \mathbb{Z}_{N_{\mathsf{M}}}$ be a cryptographic hash function. In their proofs, Boneh et al. treat H_1 as a random oracle. Finally, let E_0 be a randomly chosen supersingular elliptic curve over $\mathbb{F}_{p^2}$ and let $\{P_i, Q_i\}$ denote a basis of $E_0[N_i]$ for $i = K, M, V, R, S$. While Boneh et al. only require E_0 to be a randomly chosen elliptic curve, we will discuss how it is generated in Section 4.6 and argue that this choice should be done by a trusted third party.

First, the server chooses their private key k which is the PRF key and publishes a commitment to this key. To evaluate the OPRF at an input x in the input space, a client computes the hash $H_1(x) = m \in \mathbb{Z}_{N_{\mathsf{M}}}$. Furthermore, the client randomly chooses an element $r \in \mathbb{Z}_{N_{\mathsf{R}}}$.

The client computes the isogenies $\phi_m : E_0 \to E_m := E_0/\langle P_M + [m]Q_M\rangle$ and $\phi_r : E_m \to E_{mr} := E_m/\langle \phi_m(P_R) + [r]\phi_m(Q_R)\rangle$. Then, the client sends E_{mr} together with the

torsion point images of P_i, Q_i for $i = V, K, S$ to the server as well as a basis of $E_{mr}[N_\mathsf{R}]$. To avoid active attacks like the GPST attack [GPST16], where a malicious client tries to recover information about the server's private key by sending manipulated torsion point information, the client proves to the server in a non-interactive zero-knowledge proof that they know the kernel of the isogeny from E_0 to E_{mr} and that the provided torsion point images are indeed the images under this isogeny. For full details about the zero-knowledge proof we refer to Section 5 of [BKW20].

Subsequently, the server computes their secret isogeny $\phi_k : E_{mr} \to E_{mrk}$, where $E_{mrk} := E_{mr}/\langle \phi_r \circ \phi_m(P_K) + [k]\phi_r \circ \phi_m(Q_K)\rangle$. Moreover, the server computes the images of the order N_V torsion points and the basis of $E_{mr}[N_\mathsf{R}]$ provided by the client. The server sends E_{mrk} together with the torsion point information to the client. Using an interactive zero-knowledge proof with a cut-and-choose approach between server and client, the server can prove to the client that it computed the isogeny and the torsion point images correctly. This proof uses the torsion point images of order N_V and the server's initial commitment to the key k. Details about this zero-knowledge proof can be found in Section 6 of [BKW20].

After executing the zero-knowledge proof with the server to convince itself of the correctness of the server's reply, the client uses the images of the $E_{mr}[N_\mathsf{R}]$ torsion to "unblind" E_{mrk}. The unblinding isogeny $\hat{\phi}'_r$ is a translation of the dual of ϕ_r starting from E_{mrk}. This allows the client to compute a curve isomorphic to $E_{mk} := E_m/\langle \phi_m(P_K) + k\phi_m(Q_K)\rangle$ without knowing k at any point in time. Hashing the input together with the j-invariant of E_{mk} and the server's initial commitment to his key k yields the output of the VOPRF. The entire evaluation is sketched in Figure 4.1.

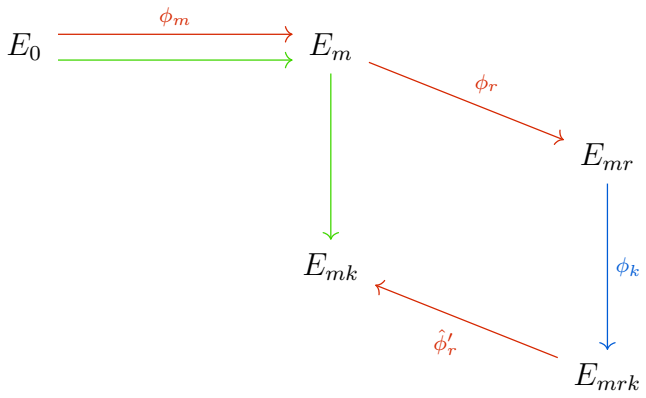

Figure 4.1: Sketch of Boneh et al.'s isogeny-based VOPRF. The isogenies computed by the client are marked in red (ϕ_m, ϕ_r, and $\hat{\phi}'_r$) while the server's isogeny is noted in blue (ϕ_k). The green isogenies represent the PRF which is jointly evaluated by the client and the server. The output of the OPRF is computed as $F(k, x) = H(x, j(E_{mk}), \mathsf{pk})$, where H is a cryptographic hash function and pk is the server's (public) commitment to his key k.

4.3 — Attacks on the auxiliary one-more SIDH assumption

In [BKW20], Boneh et al. introduce the auxiliary one-more SIDH assumption. This is a new security assumption to prove the unpredictability of their isogeny-based VOPRF. In this section we challenge the validity of this assumption and we present multiple attacks on the corresponding computational problem.

All of the attacks follow a similar strategy. First, an attacker recovers certain torsion point images up to a scalar under the secret isogeny using queries in the security game. Having recovered these torsion point images, an attacker is capable of answering any challenge set by the challenger correctly. This breaks the security assumption and also leads to an attack on Boneh et al.'s (V)OPRF.

We start by recalling the security assumption introduced by Boneh et al. [BKW20]. Then, we show that recovering said torsion point images up to a scalar is sufficient to compute the correct answer to arbitrary challenges in the corresponding security game. Subsequently, we give multiple approaches to recover these torsion point images. In

Section 4.4, we will show how the attack on the security assumption translates to an attack on the (V)OPRF itself.

4.3.1 – The auxiliary one-more SIDH assumption

First, we recall the game underlying the auxiliary one-more SIDH assumption as defined by Boneh et al. [BKW20]. While Boneh et al. use the "decision queries" defined in the following game in their security proofs, our attacks will not make use of decision queries and a reader may ignore this additional ability of an adversary.

Game 1 (Auxiliary One-More SIDH)**.** *Let $p = f \cdot N_1 \cdots N_n - 1$ be a prime depending on the security level λ and n, where N_i are smooth coprime integers and f is a small cofactor, and let $M, K \in \{1, \ldots, n\}$ be two distinct indices. Consider the following game between a challenger and an adversary:*

- *The challenger chooses a random supersingular curve $E_0/\mathbb{F}_{p^2}$ and a basis $\{P, Q\}$ of $E_0[(p+1)/(N_M \cdot N_K)]$. Moreover, it chooses $K \in E_0$ of order N_K, computes $\phi_K : E_0 \to E_K := E_0/\langle K \rangle$, and sends E_0, P, Q, and E_K to the adversary.*
- *The adversary can make a sequence of queries of the following types to the challenger:*
  - *Challenge query: The challenger chooses $M \in E_0[N_M]$ randomly and sends it to the adversary*
  - *Solve query: The adversary submits $V \in E_0[(p+1)/N_K]$ to the challenger[1], who computes $\phi_{KV} : E_0 \to E_0/\langle K, V \rangle$ and sends $j(E_0/\langle K, V \rangle)$, $\phi_{KV}(P)$, and $\phi_{KV}(Q)$ to the adversary.*
  - *Decision query: The adversary submits a pair (i, j) to the challenger, where i is a positive integer bounded by the number of challenge queries made so far,*

and $j \in \mathbb{F}_{p^2}$. The challenger responds true if $j = j(E_0/\langle K, M\rangle)$, where M is the challenger's response to the ith challenge query, and false otherwise.

- *The adversary outputs a list of distinct pairs of the form (i, j), where i is a positive integer bounded by the number of challenge queries made and $j \in \mathbb{F}_{p^2}$.*

We call an output-pair (i, j) correct, if j is the j-invariant of $E_0/\langle K, M\rangle$, where M is the challenger's response to the ith challenge query. An adversary wins the game, if the number of correct pairs exceeds the number of Solve queries.

Assumption 1 (Auxiliary One-More SIDH [BKW20])**.** *For every constant n and every distinct $M, K \in \{1, \ldots, n\}$, every efficient adversary wins the above game with probability negligible in λ.*

In the following, we will show that the auxiliary one-more SIDH assumption does not hold. We will give different attacks on the security problem underlying Assumption 1 that follow a similar strategy. Let K be the server's secret, determining the isogeny $\phi_K : E_0 \to E_0/\langle K\rangle$. The idea is to use a number of solve queries to subsequently predict $E_0/\langle K, M\rangle$ for any $M \in E_0[N_M]$. To this end, we will derive a method to extract the subgroup generated by $\phi_K(P)$ for any $P \in E_0[N_M]$ with a number of solve queries. Using this procedure, an adversary can extract the subgroups generated by $\phi_K(P_M)$, $\phi_K(Q_M)$ and $\phi_K(P_M + Q_M)$, where $\{P_M, Q_M\}$ is a basis of $E_0[N_M]$.

Knowing these subgroups allows the adversary to compute the subgroups generated by $\phi_K(M)$ for arbitrary $M \in E_0[N_M]$ without any further solve queries. Given a generator of $\langle\phi_K(M)\rangle$, the adversary can compute the j-invariant of $E_0/\langle K, M\rangle$ as $E_0/\langle K, M\rangle \cong E_K/\langle\phi_K(M)\rangle$. In particular, the adversary can produce arbitrarily many correct output-pairs and win the security game underlying the auxiliary one-more SIDH assumption (Assumption 1).

[1] In Algorithm 2, we will describe how an adversary can win the game in polynomial time, if the point V is not required to be of full order.

4.3.2 – Winning the security game given torsion point images

In this section, we show how mapping three different N_{M}-order subgroups to $E_K := E_0/\langle K \rangle$ is enough to recover sufficient information to compute a generator of $\langle \phi_K(M) \rangle \in E_K$ for any point $M \in E_0[N_{\mathsf{M}}]$.

Lemma 13. *Let P_V, Q_V, $R_V := P_V + Q_V \in E_0$ be pairwise linearly independent points of smooth order N_M and let $\phi_K : E_0 \to E_K$ be an unknown isogeny of degree coprime to N_M. Given the points P_V, Q_V, R_V and the subgroups $\langle \phi_K(P_V) \rangle$, $\langle \phi_K(Q_V) \rangle$ and $\langle \phi_K(R_V) \rangle$, an adversary can compute $\langle \phi_K(M) \rangle$ for arbitrary $M \in E_0[N_M]$.*

Proof. Fix P', Q', and R' to be generators of $\langle \phi_K(P_V) \rangle$, $\langle \phi_K(Q_V) \rangle$ and $\langle \phi_K(R_V) \rangle$ respectively. Note that the given information $\langle \phi_K(P_V) \rangle$, $\langle \phi_K(Q_V) \rangle$ and $\langle \phi_K(R_V) \rangle$ is the same as knowing $\phi_K(P_V)$, $\phi_K(Q_V)$, $\phi_K(R_V)$ up to a scalar multiple. There are many different generators for the groups $\langle \phi_K(P_V) \rangle$, $\langle \phi_K(Q_V) \rangle$ and $\langle \phi_K(R_V) \rangle$ but for any fixed choice we have

$$\begin{aligned} P' &= \alpha \phi_K(P_V), \\ Q' &= \beta \phi_K(Q_V), \\ R' &= \gamma \phi_K(R_V) \end{aligned}$$

for some (unknown) integers α, β, γ coprime to N_{M}. As isogenies are homomorphisms, we have $\phi_K(R_V) = \phi_K(P_V) + \phi_K(Q_V)$. One finds a, b such that $R' = aP' + bQ'$, which can be done efficiently as computing discrete logarithms is easy in a group of smooth order N_{M}. We have $\gamma = a\alpha = b\beta$. Thus, it is possible for the attacker to recover the ratio $\alpha/\beta = b/a$.

Given any $M \in E_0[N_{\mathsf{M}}]$, an adversary can compute integers k_1, k_2 such that $M = k_1 P_V + k_2 Q_V$ (which again is possible because N_{M} is smooth) and obtain $\langle \phi_K(M) \rangle$ by computing $\langle k_1 \phi_K(P) + k_2 \phi_K(Q) \rangle = \langle k_1 P' + k_2 \frac{\alpha}{\beta} Q' \rangle$. □

In particular, an adversary who knows $\phi_K(P_V)$, $\phi_K(Q_V)$ and $\phi_K(R_V)$ up to a scalar

and $E_K := E_0/\langle K \rangle$ can compute $E_0/\langle K, M \rangle \cong E_K/\langle \phi_K(M) \rangle$ for any $M \in E_0[N_\mathsf{M}]$.

4.3.3 – Recovering points in $\phi_K(E_0[N_\mathsf{M}])$ up to a scalar

The previous subsection shows that $E_0/\langle K, M \rangle$ can be computed by an adversary for arbitrary $M \in E_0[N_\mathsf{M}]$ as long as they can recover images of points in $E_0[N_\mathsf{M}]$ under the secret isogeny ϕ_K up to scalar. In this section, we will present multiple ways an adversary can recover this information. For didactic purposes, we include not only a polynomial and a subexponential attack (in case countermeasures to prevent the former one are put in place) but also an exponential attack in our exposition.

Query points of arbitrary order Let $M \in E_0[N_\mathsf{M}]$. We are interested in recovering $\phi_K(M)$ up to a scalar, given access to the oracle provided by the "solve queries" in Game 1. Note that our attack will not use "decision queries" as defined in the same game.

There is a simple procedure to compute an isogeny between E_K and $E_M := E_K/\langle \phi_K(M) \rangle$ and therefore $\phi_K(M)$ up to scalar, if "solve queries" are allowed for points of arbitrary order. Recall that during a solve query in Game 1, an adversary gets to submit points $V \in E_0[(p+1)/N_\mathsf{K}]$ to the challenger, who replies with the j-invariant of $E_0/\langle K, V \rangle$ and some additional torsion point images. Algorithm 2 describes how an adversary can recover $\phi_K(M)$ up to a scalar for arbitrary $M \in E_0[N_\mathsf{M}]$. The Algorithm recovers the isogeny from E_K to $E_K/\langle \phi_K(M) \rangle$ by using solve queries to obtain all intermediate curves. This allows to recover the isogeny $E_K \to E_K/\langle \phi_K(M) \rangle$ one step at a time and therefore its kernel $\langle \phi_K(M) \rangle$.

Algorithm 2 Computation of $\langle \phi_K(M) \rangle$ using solve queries on points of arbitrary order

Let $\{l_i\}_{i=0}^{n}$ be an integer sequence of all divisors of N_M such that l_{i+1}/l_i is a prime, $l_i < l_{i+1}$, with $l_0 := 1$, $l_n := N_\mathsf{M}$.
Input: E_K, $M \in E_0[N_\mathsf{M}]$ and access to an oracle answering solve queries as defined in Game 1.
Output: A generator of $\langle \phi_K(M) \rangle$.

1: $E^{(n)} \leftarrow E_0/\langle K \rangle$;
2: **for** $i = n-1, \ldots, 0$ **do**
3: Query the oracle with the point $V_i := [l_i]M$ and obtain the curve $E^{(i)} := E_0/\langle K, V_i \rangle = E_0/\langle K, [l_i]M \rangle = E_K/\langle [l_i]\phi_K(M) \rangle$;
4: Find l_{i+1}/l_i-isogeny ϕ_i from $E^{(i+1)}$ to $E^{(i)}$;
5: **return** *A generator of* $\ker(\phi_0 \circ \cdots \circ \phi_{n-1})$;

Lemma 14. *Algorithm 2 returns $\lambda\phi_K(M)$, where $\lambda \in \mathbb{Z}$ is coprime to N_M.*

Proof. Let ψ_M be the isogeny from E_K to $E_K/\phi_K(M)$. Then the claim follows from the observation that $E_0/\langle K, [l_i]M \rangle \cong E_0/\langle [l_i]K, [l_i]M \rangle$, since l_i is coprime to the order of K. □

Remark 2. *Note that an attacker can easily change the attack to require fewer queries. Instead of using one query for each intermediate curve, an attacker can choose any factorisation $f_1 \cdots f_t$ of N_M such that f_i are roughly of equal size and query the oracle with $\left[\prod_{j=1}^{b} f_i\right] M$ for $b = 1, \ldots, t$. Then, the attacker is left to recover the isogeny between any two consecutive queries, i.e. the isogenies of degree f_i for $i = 1, \ldots, t$, using a meet-in-the-middle attack.*

In Game 1, Boneh et al. did not specify any restrictions on the points of $E_0[(p+1)/N_\mathsf{K}]$ that can be submitted to the solve queries. However, in the context of the game, this attack can be easily thwarted by answering to a solve query only if the submitted point is of order $(p+1)/N_\mathsf{K}$. This property can be checked efficiently by the challenger. In Section 4.4, we discuss how this polynomial-time attack and its countermeasures translate to the VOPRF protocol.

Query points of order $(p+1)/N_\mathsf{K}$ Next, we present how an attacker can retrieve the necessary information even if they are only allowed to send solve queries on points of

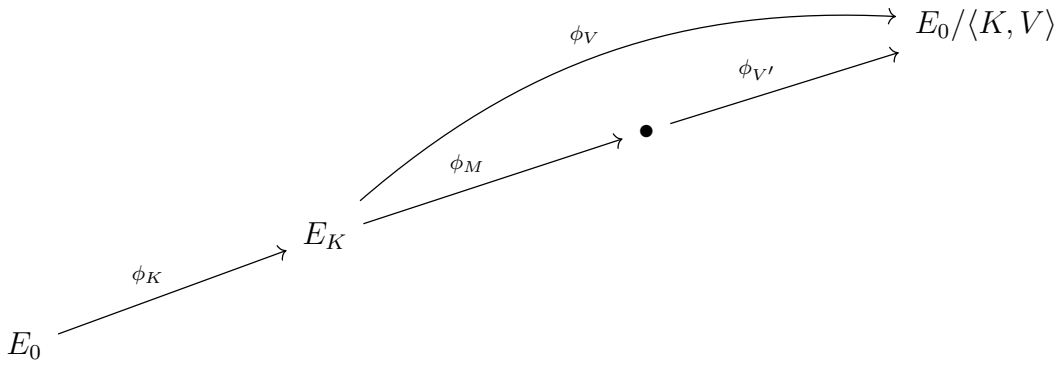

Figure 4.2: Depiction of the isogenies of a solve query

order $(p+1)/N_{\mathsf{K}}$, i.e. if the challenger checks the order of a submitted point and only replies to a query if the point is of order $(p+1)/N_{\mathsf{K}}$.

Let ϕ_V denote the isogeny $E_K \to E_0/\langle V, K\rangle$ of degree $(p+1)/N_{\mathsf{K}}$ and let $\phi_V = \phi_{V'} \circ \phi_M$ be its decomposition into a degree $(p+1)/(N_{\mathsf{K}}N_{\mathsf{M}})$ and a degree N_{M} isogeny. Our attack aims to recover the image of multiple subgroups of $E_0[N_{\mathsf{M}}]$ under the isogeny ϕ_K, i.e. we are interested in the kernel of the isogeny ϕ_M for different points V. The isogenies are depicted in Figure 4.2.

Recovering $\phi_{V'}$ from torsion point information Let $P, Q \in E_0[(p+1)/N_{\mathsf{M}}N_{\mathsf{K}}]$ be the torsion point basis provided by the challenger and let $V \in E_0[(p+1)/N_{\mathsf{K}}]$ be linearly independent of P or Q. Then, we can use the torsion point images provided during a solve queries to compute $\hat{\phi_V'}$ as follows.

Let $P' := \phi_V \circ \phi_K(P)$, $Q' := \phi_V \circ \phi_K(Q)$ be the images of the torsion points provided by the challenger. The adversary can compute $\hat{\phi}_{V'}$ as the isogeny from $E_0/\langle K, V\rangle$ with kernel $\langle P', Q'\rangle$. Note that $\langle P', Q'\rangle \subset \ker(\hat{\phi}_{V'})$, because $\hat{\phi}_{V'} \circ \phi_{V'} = [(p+1)/N_{\mathsf{M}}N_{\mathsf{K}}]$ is the order of the points P, Q. As V is linearly independent to at least one of P and Q, the other inclusion follows from $\langle P', Q'\rangle$ spanning a subgroup of size $(p+1)/N_{\mathsf{M}}N_{\mathsf{K}}$.

Choosing P_V, Q_V as a basis of $E_0[(p+1)/N_{\mathsf{K}}]$ such that $[N_{\mathsf{M}}]P_V = P + [(p+1)/N_{\mathsf{M}}N_{\mathsf{K}}]Q$ and $[N_{\mathsf{M}}]Q_V = [(p+1)/N_{\mathsf{M}}N_{\mathsf{K}}]P + Q$, every point of the form $P_V + [i]Q_V$ or $[i]P_V + Q_V$

will be linearly independent of P or Q.

As a consequence of $\phi_{V'}$ being easy to recover, we may assume that during a solve query an attacker can send a point M of order N_M to the challenger who returns $E_0/\langle K, M\rangle$. We are left to recover the kernel of ϕ_M.

Naïve attack to recover ϕ_M Next we describe an exponential attack that recovers $\hat{\phi}_M$ using meet-in-the-middle (MITM) computations of increasing size. In the subsequent part, we will introduce a trade-off between queries and computation costs that reduces the complexity of the attack to subexponential.

Let P_M, Q_M denote a basis of $E_0[N_\mathsf{M}]$. For simplicity of exposition we treat N_M as a prime power and we write $N_\mathsf{M} = \ell_M^{e_M}$. The attack recovers $\phi_M : E_K \to E_K/\langle P_M\rangle$ by recovering each of the e_M intermediate curves one at a time.

The attacker starts by querying the solve oracle with two points $V_0 := P_M$ and $V_1 := P_M + [\ell_M^{e_M-1}]Q_M$. Note that the curves $E_K/\langle\phi_K(V_0)\rangle$ and $E_K/\langle\phi_K(V_1)\rangle$ are ℓ_M^2-isogenous, since they are both ℓ_M-isogenous to $E_K/\langle[\ell_M]\phi_K(V_0)\rangle = E_K/\langle[\ell_M]\phi_K(V_1)\rangle$. The attacker recovers the curve $E_K/\langle[\ell_M]\phi_K(V_0)\rangle$, which is the first intermediate curve on the ϕ_M isogeny path by computing the common neighbour of $E_K/\langle\phi_K(V_0)\rangle$ and $E_K/\langle\phi_K(V_1)\rangle$.

The rest of the attacks proceeds similarly. The attacker queries with the points $V_i := P_M + [\ell_M^{e_M-i}]Q_M$, $i = 1, \ldots, e_M/2$ and runs a MITM attack to recover $E_K/\langle[\ell_M^i]\phi_K(V_0)\rangle$ given $E_K/\langle\phi_K(V_i)\rangle$ and $E_K/\langle[\ell_M^{i-1}]\phi_K(V_0)]\rangle$. This could be repeated e_M times to recover the entire isogeny ϕ_M. However, the attacker does not need to recover the last part of the isogeny through this strategy, since it is faster to directly compute the MITM between $E_K/\langle[\ell_M^{e_M/2}]V_0\rangle$ and the starting curve E_K. The attack with the required meet-in-the-middle computations is shown in Figure 4.3.

Note that the isogenies that need to be recovered using MITM grow at each step. To recover the i-th intermediate curve, the attacker needs to compute an isogeny between two curves that are $\ell_M^{(i+1)}$-isogenous, which takes roughly $O(\ell_M^{(i+1)/2})$.

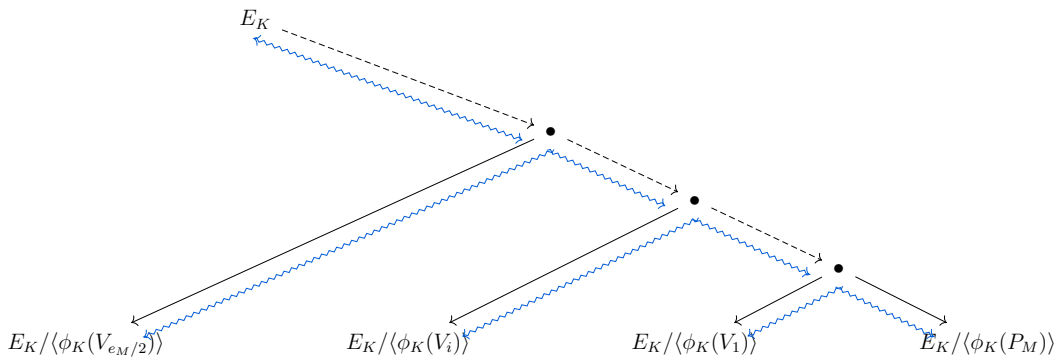

Figure 4.3: Naïve attack where isogenies of increasing length need to be recovered. The blue lines represent the meet-in-the-middle computations.

Clearly, this attack can be optimised by recovering multiple steps of ϕ_M at a time, and by making sure that the different MITM attacks that need to be executed have similar complexity. We will discuss these improvements in the following.

Full attack with query-time trade-off We can reduce the complexity of the naïve attack by introducing a trade-off between queries and the cost of MITM computations. This is because the attacker recovers the whole path between two isogenies during a MITM computation. Thus, it is possible to recover more than one intermediate curve with a single (longer) MITM computation. Moreover, the queries can be spaced out more in order to reduce the length of the isogenies that have to be recovered using MITM strategies.

More formally, let 2^q denote the number of queries that an attacker can (or wants to) send to the challenger. For simplicity of this exposition, assume that $2e_m$ is divisible by $q+2$. The attacker chooses the V_i such that $E_0/\langle K, V_i\rangle$ correspond to curves that are the leaves of a binary isogeny tree. The V_i should be chosen such that there is an $\ell_M^{2e_M/(q+2)}$ isogeny between any two siblings in the binary tree and the curve that is $\ell_M^{e_M/(q+2)}$-isogenous to both leaves is their parent in the tree. Again, the parent and its sibling should be $\ell_M^{2e_M/(q+2)}$-isogenous, etc.

Remark 3. *Note that it is easy to choose such a set of points V_i. Let P_M, Q_M be a basis of*

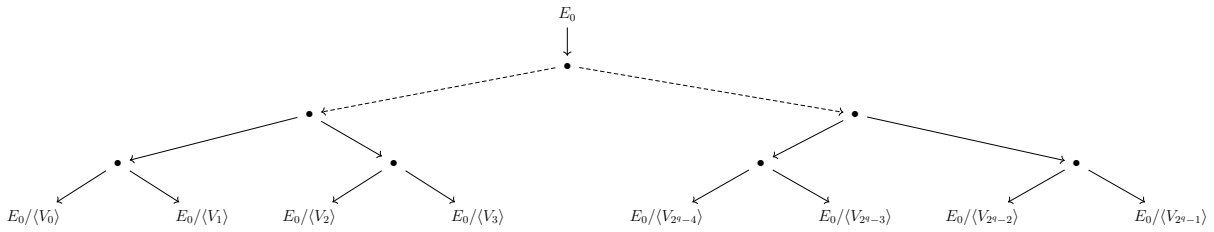

Figure 4.4: The attacker queries the challenger on points corresponding to isogeny kernels leading to the leaves of this binary tree

$E_0[\ell_M^{e_M}]$. *The attacker can choose*

$$V_0 := P_M$$
$$V_i := V_{i-2^{\lfloor \log i \rfloor}} + [\ell_M^{e_M-(\lfloor \log i \rfloor+1)2e_M/(q+2)}]Q_M$$

Lemma 15. *Let $E_0/\langle V_i\rangle$ and $E_0/\langle V_j\rangle$ be ℓ_M^k isogenous curves. Then $E_K/\langle \phi_K(V_i)\rangle$ and $E_K/\langle \phi_K(V_j)\rangle$ are ℓ_M^k-isogenous curves too.*

Proof. This follows from $N_\mathsf{K} = \deg(\phi_K)$ being coprime to ℓ_M^k. □

In particular, $\{\phi_K(P_M), \phi_K(Q_M)\}$ is a basis of $E_K[N_\mathsf{M}]$ and $E_K/\langle \phi_K(V_i)\rangle$ are the leaves in a binary tree where all siblings are $\ell_M^{2e_M/(q+2)}$ isogenous.

After querying the oracle to obtain $E_K/\langle \phi_K(V_i)\rangle = E_0/\langle K, V_i\rangle$, an attacker recovers iteratively parent nodes in the binary tree using a meet-in-the-middle approach. Any siblings in the tree correspond to curves that are $\ell_M^{2e_M/(q+2)}$-isogenous, thus this can be done in $O(\ell_M^{e_M/(q+2)})$. Note that the root of the binary tree is recovered after $2^q - 1$ such meet-in-the-middle instances, i.e. the number of internal nodes in the binary tree. This root of the binary tree is then by construction $\ell_M^{2e_M/(q+2)}$-isogenous to E_0. This final isogeny can be recovered using meet-in-the-middle again. An attacker recovers and saves the intermediate nodes and isogenies from E_K to the leaf $E_K/\phi_K(V_0)$. Clearly, the kernel of this isogeny is $\phi_K(V_0)$.

In summary, we can recover the isogeny from $E_K \to E_K/\langle \phi_K(P_M)\rangle$ for any P_M with 2^q queries to the challenger and 2^q instances of meet-in-the-middle isogeny computations with cost of $O(\ell_M^{e_M/(q+2)})$ each.

Remark 4. *If $\ell_M = 2$, we get q bits for free, i.e. one additional bit per layer of the binary tree. This is because every parent node in the binary tree has three outgoing edges: two edges leading to its children and one edge leading towards the root. Thus, having recovered both paths to the children an attacker gets one step towards the root for free.*

4.3.4 – Attack analysis

The proposed attack is composed of two stages: firstly the generators of $\langle \phi_K(P_V) \rangle$, $\langle \phi_K(Q_V) \rangle$, and $\langle \phi_K(R_V) \rangle$ are recovered, and then these points are used to recover $\phi_K(M)$ for any possibly challenge $M \in E_0[N_\mathsf{M}]$.

The second part consists mostly of pairing evaluations and discrete log computations in groups of smooth order. Thus, it runs in polynomial time. The complexity of the attack is dominated by the complexity of recovering the subgroups in the first step.

The algorithm proposed in Section 4.3.3 offers different trade-offs between computation costs and solve queries. As little as two solve queries can be enough to recover ϕ_M with two meet-in-the-middle computations. If we write $N_\mathsf{M} \approx 2^m$, each meet-in-the-middle requires $O(2^{m/3})$ operations. This is already an improvement over the standard meet-in-the-middle attack that requires $O(2^{m/2})$ time. The OPRF protocol targets 128 bits of security, which corresponds to $m \approx 5\lambda/2 = 320$. Thus six queries (two per generator) are enough to reduce the security to $m/3 = 106$ bits. The number of solve queries can be significantly increased to obtain a faster attack. Note that OPRF protocols are usually used for applications such as private set intersection, that support a large number of queries. Thus, common scenarios where the OPRF may be used would easily lend themselves to an attack with many queries.

Since OPRFs are used in protocols where the clients interact several times with the server, we can expect the attacker to be able to run several OPRF instances. Thus, we model a solve query as an oracle query, where it has a unitary complexity. Then, the overall complexity of recovering a generator of $\langle \phi_K(P_V) \rangle$ with 2^q solve queries is $O(2^{m/(q+2)+q})$ operations, since the attacker needs to compute 2^q meet-in-the-middle instances between

curves which are $2^{2m/(q+2)}$-isogenous. In terms of the security parameter, that complexity is equivalent to $O(2^{5\lambda/2(q+2)} + q)$, since the OPRF protocol suggests using $m \approx 5\lambda/2$. If the number of solve queries is unrestricted, the complexity of the attack is minimized for $q = \sqrt{5\lambda/2} - 2$, which gives an overall complexity of $O(2^{\sqrt{10\lambda}-2})$, or using the L-notation $L[1/2, c]$, for some constant c. This shows the attack is subexponential, assuming that the solve query complexity is $O(1)$.

At 128-bit of security, our attack becomes feasible with around 64 solve queries, when it requires 64 meet-in-the-middle computations between curves which are 2^{80}-isogenous, i.e. each MITM has a complexity of 2^{40} operations. If the number of solve queries is unrestricted, an attacker can use 2^{18} solve queries to reduce the overall complexity of the attack to 2^{18} MITM computations, where each MITM operations has a complexity of 2^{16} operations.

The high-level attack does not generally require much memory. Storing the isogeny tree in memory is not particularly demanding, especially if the tree is traversed depth-first. In particular, memory is used only to store the part of the recovered isogeny, together with the two curves between which the meet-in-the-middle needs to be computed. However, a more significant amount of memory is used by the meet-in-the-middle computations, and indeed we see that the memory used by a single meet-in-the-middle generally outweighs the memory used by the rest of the attack. Meet-in-the-middle computations between curves which are 2^n-isogenous require to store $2^{n/2}$ curves. Thus, their memory requirements are given by $2 \cdot 2^{n/2} \log p$, since each curve can be represented by its j-invariant in $\mathbb{F}_{p^2}$. For common security levels, such as those proposed by Boneh et al. [BKW20], the memory requirements remain moderate. In Section 4.5, we show that indeed our attack requires about 3 GB of memory to break 128 bits of security. However, for a more complete asymptotical analysis, we note that the memory requirements may become a bottleneck for the attack against higher security levels. In those instances, it may be preferable to substitute the meet-in-the-middle approach with the van Oorschot-Wiener algorithm [vW99]. This reduces the memory consumption at the cost of higher asymptotic

complexity. In particular, the vOW algorithm requires $O(2^{3n/4})$ computations (compared to $O(2^{n/2})$ of MITM) to recover the halfway curve between curves which are 2^n-isogenous. Thus, while the concrete performance of the attack may differ, its asymptotic complexity remains subexponential.

Future improvements A natural question to ask is whether the proposed attack that queries points of the correct order may be improved to achieve a polynomial running time. Consider that an attacker chooses an isogeny $\phi_V : E_0 \to E_0/\langle V \rangle$ and he is given the curve $E_0/\langle K, V \rangle$. Since the attacker knows the entire isogeny ϕ_V, backtracking from $E_0/\langle K, V \rangle$ to E_K to recover $\phi_K(V)$ in polynomial time does not seem too far fetched, since the attacker knows the entire isogeny ϕ_V. A possible strategy may start by retrieving $E_0/\langle K, V \rangle$ and $E_0/\langle K, V + \ell_M^{e_M} V' \rangle$, for a point V' linearly independent of V. Their common ℓ_M-neighbour is the first curve on the isogeny path. Then, the attacker may use the knowledge of ϕ_V starting from E_0 to distinguish between the ℓ_M possible candidates for the next curve on the isogeny path. Unfortunately, our efforts to develop such an attack did not succeed. It remains an open problem whether such an attack is possible.

4.4 — Attack on the OPRF

Having presented an attack on one of the security assumptions underlying the isogeny-based OPRF by Boneh et al., we investigate how an adversary can use the same method to attack the OPRF itself.

We will show that a malicious client can send carefully crafted queries to the server for which it can produce all necessary NIZK proofs required by the protocol that was summarized in Section 4.2.4. However, after some offline computation analogously to the attack on the auxiliary one-more SIDH assumption outlined in the previous section, the malicious client can evaluate the OPRF on any input without the help of the server. Even though the malicious client does not recover the server's secret key k, this breaks

the "pseudorandomness", Definition 3, of the OPRF. We will use the same notation as in Section 4.2.4 to refer to different isogenies of the OPRF.

A malicious client will not use a hashed input to obtain the kernel for the first isogeny $\phi_m : E_0 \to E_m$ but rather choose the kernel of this first isogeny maliciously. The choice is analogous to the points from $E_0[N_\mathsf{M}]$ that the attacker submitted to the solve queries in the attack of the previous section. In other words, instead of computing E_m as $E_0/\langle P+H(x)Q\rangle$ for some input x, the malicious client chooses a point V_i and computes E_m as $E_0/\langle V_i\rangle$ in the i-th evaluation of the OPRF.

The rest of the protocol is executed honestly. The malicious client can pick some $r \in N_\mathsf{R}$ to blind his maliciously chosen E_m. And it can compute the torsion point information for the server honestly since it knows the kernel of the isogeny $E_0 \to E_{mr} = E_m/\langle \phi_m(P_R) + [r]\phi_m(Q_R)\rangle$. In particular, the malicious client will always be able to produce the valid non-interactive zero-knowledge proof of knowledge for the kernel of $E_0 \to E_{mr}$ and the correct computation of the torsion point information.

Following through with the rest of the OPRF protocol, the malicious client obtains the j-invariant of the curve $E_0/\langle V_i, K\rangle$ after unblinding. Here K denotes the server's secret $P_K + [k]Q_K$. This is exactly what corresponds to a "solve query" in the auxiliary one-more SIDH game, Game 1.

Now the malicious client can proceed as in the attacks on the auxiliary one-more SIDH assumption.

In the attack using points of arbitrary order dividing N_M, the malicious client recovers the isogeny $E_K \to E_K/\langle\phi_K(P)\rangle = E_0/\langle K, P\rangle$ and therefore $\langle\phi_K(P)\rangle$ for any $P \in E_0[N_\mathsf{M}]$ in polynomial time. This is done by submitting points of lower order, i.e. choosing the isogeny $E_0 \to E_m$ shorter, to recover the isogeny stepwise. After recovering three such isogenies corresponding to pairwise linearly independent points $P, Q, P+Q$, the malicious client can compute $E_0/\langle M, K\rangle$ for any $M \in E_0[N_\mathsf{M}]$ as was shown in Section 4.3.2.

Then, the malicious client can evaluate the OPRF on arbitrary inputs x as follows: They compute the point $M := P_M + H_1(x)Q_M$ as in the honest evaluation and then they

compute $j(E_0/\langle M, K\rangle)$ directly. Hashing this j-invariant together with the input x and public information of the server yields the output of the OPRF. Note that the malicious client does not even need to interact with the server to evaluate the OPRF on arbitrary inputs.

Clearly, this breaks the pseudorandomness property of an OPRF, see Definition 3, as a malicious client will be able to predict the output of the OPRF for any input after the initial queries.

Remark 5. *The SIDH-based OPRF protocol by Boneh et al. does not prohibit malicious clients from using points of smaller order dividing N_M, i.e. from using a shorter isogeny $E_0 \to E_m$. However, this attack could be thwarted if the server checked that the submitted curve is of correct distance from the starting curve. A simple test using pairing computations on the provided torsion point information may be tricked, but the NIZK of the client could be extended to include a proof that the client's witness, i.e. the kernel of the isogeny $E_0 \to E_{mr}$, is of full order $N_M N_R$.*

Even if countermeasures for this polynomial-time attack are put in place, we are left with the following subexponential attack when points of full order are used.

The client evaluates the OPRF on a certain number of inputs that correspond to solve queries in the auxiliary one-more game. More precisely, the client chooses the kernel of his first isogeny as in the subexponential attack of the previous section. After blinding, evaluation of the server and unblinding, the client obtains what would have been the result of a "solve query" in the previous section. After the offline computation which, using meet-in-the-middle routines, recovers the binary tree described in Section 4.3.3, the client obtains torsion point images of $E_0[N_M]$ up to scalar under the isogeny $E_0 \to E_K := E_0/\langle P_K + [k]Q_K\rangle$. Again this is enough to compute $E_0/\langle M, K\rangle$ for any $M \in E_0[N_M]$ by Section 4.3.2, allowing the client to compute the OPRF on arbitrary inputs and therefore breaking the pseudorandomness property.

Possible countermeasures In the case where the degree of the client's isogeny is forced to be $N_{\mathsf{M}}N_{\mathsf{R}}$, the proposed attack has subexponential complexity, and thus possible countermeasures may include increasing the parameter sizes. However, the solve queries to time trade-off may reduce the feasibility of such an approach. If the number of possible solve queries is unrestricted, to get 128-bit security one would need the isogeny degree N_{M} to be $\approx 2^{(128^2)}$. This can be partially mitigated by guaranteeing security only up to a certain number of queries. Given a limit of 2^Q queries, the exponent m needs to guarantee that $\min\{2^{\sqrt{m}-2}, 2^{m/(Q+2)+Q}\}$ is at least 2^λ. Thus, for 128-bit security, with $Q = 64$ the isogeny degree N_{M} would have to be increased to $\approx 2^{4224}$, whereas $Q = 32$ would require a degree $N_{\mathsf{M}} \approx 2^{3264}$. Note that handling 2^{32} queries may well be within the scope of several OPRF applications, and isogenies of such a size may become impractically large. Their feasibility, however, depends on the specifics of the OPRF application and its time and bandwidth requirements. Thus, while the attack is subexponential (assuming $O(1)$ complexity for solve queries), increasing the parameter size comes at a significant performance and communication cost.

Therefore, it is important to consider possible algorithmic countermeasures. Firstly, note that the attacker submits seemingly valid requests, so the server cannot stop such interactions. Even if the server did want to prevent these requests, it may not be able to detect them. This is because the attacker only submits the image curve and some torsion point images under an isogeny with chosen kernel.

However, the attack strongly depends on the attacker choosing the point V. If the input points V were randomized, the attack as such could not work. The OPRF protocol requires that such points are obtained via hashing the client's PRF input x, but it does not enforce it. Hence, a possible countermeasure to the proposed attack would be requiring the client to provide a zero-knowledge proof that the curve E_{mr} is not only the result of honest isogeny computations, but also that the kernel of ϕ_m is the result of some hash function. However, developing an ad-hoc and efficient proof that can prove such statements remains an open problem.

4.5 — Attack implementation

We implemented the subexponential attack of Section 4.3.3 in SageMath to demonstrate the correctness of the algorithm and prove its feasibility. The source code is freely available at https://github.com/isogenists/isogeny-OPRF. We remark that this implementation is to be regarded only as a proof-of-concept and that several subroutines can be further optimized. Improving their performance and using lower-level languages, such as C, as well as platform-specific instructions, such as AVX, could significantly reduce the running time of the attack.

The proposed attack has two distinguishing features that help its implementation: it can be easily parallelized, and it has very low memory requirements. Indeed, the computations to recover the generators of $\langle \phi_K(P_V) \rangle$, $\langle \phi_K(Q_V) \rangle$ and $\langle \phi_K(P_V + Q_V) \rangle$ are independent of each other. It is also possible to achieve a higher degree of parallelization. Within each computation to recover a single generator, the meet-in-the-middle operations within each layer of the tree are also independent of each other, and they can thus be parallelized. In this case, the tree is generated layer-by-layer in a breadth-first manner. Note that while this may require a sizeable amount of memory to fully store an entire layer, the memory requirements are hardly the bottleneck. An attack with 2^{20} queries requires to store, at most, 2^{19} curves. Since an elliptic curve can be represented by its j-invariant, the memory limit is $2^{19} \cdot 2 \log p$. With a prime of size $\approx 2^{1500}$, as proposed in the OPRF protocol, the memory limit is about 196 MB. Alternatively, it is possible to traverse the tree in a depth-first manner to further lower the memory requirement, but this may limit the degree of parallelization. We remark that while parallelization only provides a linear speed-up, its effects can be significant. Our implementation provides parallelized meet-in-the-middle computations with a configurable number of cores in parallel.

Results The majority of the attack's subroutines have polynomial complexity and they are optimized enough that their performance does not affect the overall running time. The building block that most affects the performance of the attack is the meet-in-the-middle computation. Indeed, the timings of the attack are directly correlated to the timing of a single meet-in-the-middle and the total number of queries. The memory requirements of the attack are given by the amount of memory needed for a single meet-in-the-middle, which in turn depends on the distance between the two curves. For parallelized implementations of the attack, the memory requirements correspond to as many meet-in-the-middle computations as there are parallel instances.

Table 4.1 shows the running times at different security levels on an Apple M1 CPU clocked at 3.20 GHz with 4 CPUs running in parallel. Up to 32 bits of security, the results come from running the entire attack, whereas for higher security levels the results are estimated based on those of a single MITM computation. The estimated time t is computed as

$$t = \frac{3(M+Q)2^q}{C}, \tag{4.1}$$

where M is the average running time of a MITM computation, Q is the average running time of a solve query computation, 2^q is the number of queries and C is the number of CPUs running in parallel. This formula follows from the fact that there are 2^q MITM computations and 2^q solve queries for each generator recovery, and three of those are needed. Moreover, parallelization gives a linear speed-up, and the remaining computations (such as those of Section 4.3.2) are extremely fast when compared to the rest of the attack, and thus negligible. Running computations at lower security levels and computing Eq. 4.1 does indeed estimate the running time accurately. It should be noted that this remains an estimate and the real results may vary to some degree.

We estimate that our non-optimized implementation running on a laptop with 4 CPUs can break 64 bits[2] of security in less than two days and 128 bits of security in about 5

[2]We report the results for $e_M = 169$, which corresponds to $\lambda = 67$. That is because our implementation requires $(q+2) \mid e_M$, and 169 allows choosing $q = 11$. Using $e_M = 160$ would have required using significantly more queries or a longer MITM, thus resulting in worse performance. Note that the requirement

years. If the same attack was performed with more powerful hardware and an optimized implementation, the running time could easily be reduced to a matter of months, if not weeks. We remark that if a server rotates its keys often, an attack that breaks the server one-more unpredictability after the key has changed still leads to significant attacks. For instance, in the case of OPRF-based private set intersection protocols, breaking the one-more unpredictability property allows the attacker to break the privacy property of the server's set at the time when that specific key was used.

Lastly, note that in the implementation solve queries are simulated locally. A real attack would interact with the server, and thus the "correct" attack time should not include the query computation times. For completeness, Table 4.1 reports the running time of the entire implementation, including the solve queries.

Parameters				**MITM**		**Running Time**
$\log p$	λ	e_M	q	Distance	Memory (kB)	(s)
112	8	20	3	8	3.5	15
216	16	40	6	10	13.8	212 (3.53 m)
413	32	80	8	16	211.4	1,371 (22.85 m)
859	67	169	11	26	14,073	163,869 (1.89 d)
1,614	128	320	18	40	3,384,803	174,709,440 (5.54 y)

Table 4.1: Results of our proof-of-concept implementation of the attack, running on an Apple M1 CPU clocked at 3.20 GHz with 4 CPUs in parallel and SageMath version 9.2. Results for $\lambda = 128$ are estimated based on the average running time of a meet-in-the-middle computation. Parameters include the size of the prime p, the security level λ, the degree of the isogeny written as $N_{\mathsf{M}} = 2^{e_M}$, and the number of queries 2^q. The MITM section reports the distance between the curves and memory needed to compute a single meet-in-the-middle.

4.6 — Trusted setup

In the OPRF protocol of Boneh et al., the authors suggest using a random supersingular elliptic curve as starting curve. However, there is currently no known algorithm to generate a random supersingular elliptic curve such that its endomorphism ring is unknown to

that $(q+2) \mid e_M$ is a limitation of the implementation and not of the attack itself.

the person who generated it. Some attempts to solve this problem have been proposed in [LB20] and further studied in [CPV20]. This motivates the following question:

Is a trusted third party needed to generate the starting curve E_0?

Phrased differently, would choosing the starting curve E_0 and therefore knowledge of its endomorphism ring allow a malicious server, client or third party to break security properties of the (V)OPRF?

We first discuss whether a server may know the endomorphism ring of the starting curve E_0. The security proof by Boneh et al.'s OPRF relies on the hardness of finding two distinct isogenies (up to isomorphism) of the same degree from E_0 to a second curve [BKW20, Lemma 29]. If the server chooses the starting curve and therefore knows its endomorphism ring, they are able to produce such collisions by breaking the collision resistance of the CGL hash function as in [PL17; EHLMP18]. To guarantee provable security, a server should therefore not choose the starting curve.

However, breaking the verifiability insured by the zero-knowledge proof [BKW20, Protocol 17] or the weak binding property [BKW20, Game 3] of the protocol seems harder than finding collisions. Indeed, the server would need to produce two isogenies of degree dividing N_{K} such that both isogenies have the same action on the N_{V}-torsion for a chosen starting curve. We leave adapting the security proofs or finding an attack on the zero-knowledge proof for future work.

We now argue that any other party, either the client or a third party, cannot choose the curve E_0 either without compromising the security of the protocol. In [QKLMP+21], the authors describe algorithms for finding a secret isogeny when torsion information is provided. Their algorithms can be split into two categories: one where the starting curve has j-invariant 1728 and one where the starting curve is a trapdoor curve from which one can solve the isogeny problem faster than generic meet-in-the-middle algorithms. Trapdoor curves are parameterized by a pair (A, B) where A corresponds to the degree of the secret isogeny and B to the order of torsion points whose image under the secret

isogeny is known. When $B \approx A^2$ or larger, then one can construct (A, B) trapdoor curves from which one can retrieve secret isogenies of degree A in polynomial time, if the action on the B-torsion is known [QKLMP+21, Theorem 15].

Attacks from the special starting curve with j-invariant 1728 do not apply here, since the starting curve cannot have j-invariant 1728 because the endomorphism ring needs to be unknown to the server. However, trapdoor curves have the property that without extra information they are difficult to distinguish from a random supersingular curve.

Suppose that a malicious party generates the starting curve E_0 in the following way. They generate a curve E' which is a trapdoor $(N_\mathsf{K}, N_\mathsf{V}N_\mathsf{R})$-curve and then perform a random walk of length $N_\mathsf{M}N_\mathsf{R}$ to obtain E_0 which is sent to the server. Now the malicious party poses as a client and instead of honestly complying with the protocol, they use E' as E_{mr}. They can prove knowledge of a suitable isogeny and torsion point images as they know an isogeny of the correct degree from E_0. Then the server computes E_{mrk} and reveals the action of the $N_\mathsf{V}N_\mathsf{R}$-torsion. Since E_{mr} was chosen to be a trapdoor curve and $N_\mathsf{V}N_\mathsf{R} \approx N_\mathsf{K}^2$, the malicious party can retrieve this isogeny in polynomial time.

Such an attack can be thwarted by applying a trusted setup in which E_0 is a truly random curve. In [BD21, §4], an efficient way to perform a distributed trusted setup is described, ensuring that, if at least one participant is honest, the setup can be trusted. In that case, torsion point attacks are not applicable. The attack can also be weakened by substantially increasing N_K. However, this might be susceptible to future improvements of trapdoor curve constructions.

4.7 — Conclusion

In this paper, we perform a thorough cryptanalysis of Boneh et al.'s SIDH-based oblivious pseudorandom function. The security of this OPRF is based on a new hardness assumption, the auxiliary one-more assumption. We investigate this assumption and we show how an attacker can win the corresponding security game in polynomial time, or with the

appropriate countermeasures in subexponential time.

The attack on the underlying hardness assumption leads to an attack on the pseudorandomness of the OPRF itself. We show how a malicious client can extract enough information from a number of initial executions of the OPRF protocol to subsequently evaluate the OPRF on arbitrary inputs without further interaction with the server. In particular, this attack breaks the security parameters provided by Boneh et al. As a proof of concept, we implement the attack in SageMath, verified its correctness and give timings for various security levels.

Furthermore, we discuss the security implications following from a lack of a trusted setup when generating the starting curve parameter in the SIDH-based OPRF. Note that Boneh et al. do not explicitly require a trusted setup. We show how a client or a third party generating the starting curve can backdoor it to retrieve the server's secret key, while a malicious server could generate the starting curve to break the supersingular-isogeny collision assumption.

This work leads to some open problems. On one hand, one could improve and extend the proposed attack, with a particular focus on reducing the complexity of the subexponential attack to polynomial time, as well as extending it to the CSIDH-based OPRF. On the other hand, further work is needed to develop efficient countermeasures against the subexponential attack or to design a novel SIDH-based VOPRF. Future research will also focus on understanding the implications of breaking the supersingular-isogeny collision assumption on the OPRF protocol itself, and whether it is possible to avoid a trusted setup.

Acknowledgments. We would like to thank Dan Boneh, Jesús Javier Chi Domínguez, Luca De Feo, Enric Florit, Dmitry Kogan and Simon Masson for fruitful discussions. Péter Kutas, Simon-Philipp Merz and Christophe Petit were supported by EPSRC and the UK government as part of the grant EP/S01361X/1 for Péter Kutas and Christophe Petit and the grant EP/P009301/1 for Simon-Philipp Merz. Further, Péter Kutas was supported by

the Ministry of Innovation and Technology and the National Research, Development and Innovation Office within the Quantum Information National Laboratory of Hungary.

Part II

New Protocols

Chapter 5

A Post-Quantum Oblivious PRF from Isogenies

Life is a series of trade-offs.
— Killing Eve, Season 3

This chapter is a verbatim reproduction of the following paper:
Andrea Basso. *A Post-Quantum Round-Optimal Oblivious PRF from Isogenies*. Cryptology ePrint Archive, Paper 2023/225. 2023. URL: https://eprint.iacr.org/2023/225

Abstract: An oblivious pseudorandom function, or OPRF, is an important primitive that is used to build many advanced cryptographic protocols. However, very few post-quantum solutions exist.

In this work, we propose a novel OPRF protocol that is post-quantum, verifiable, round-optimal, and moderately compact. Our protocol is based on a previous SIDH-based construction by Boneh et al., which was later shown to be insecure due to an attack on its one-more unpredictability. We propose an efficient countermeasure against this attack, and we demonstrate how to efficiently adapt the protocol to work with the countermeasures against the SIDH attacks. To achieve this, we also propose the first proof of isogeny knowledge that is compatible with masked torsion points, which may be of independent interest. We also design a novel non-interactive proof of knowledge of

parallel isogenies, which reduces the number of communication rounds of the OPRF to the theoretically-optimal two. Putting everything together, we obtain the most compact post-quantum verifiable OPRF protocol.

5.1 — Introduction

An oblivious pseudorandom function (OPRF) is a two-party protocol between a user and a server. The two parties obliviously evaluate a PRF on a user-controlled input with a secret key held by the server. After engaging in the protocol, the user learns only the output of the PRF on their chosen input, while the server does not learn anything, neither the user's input nor the output of the function. Oblivious PRFs can also satisfy a stronger property called *verifiability*: in a verifiable OPRF (vOPRF), the server initially commits to its secret key, and during the execution of the protocol it provides a proof that it has behaved honestly and it has used the previously committed secret key.

Oblivious PRFs have widespread applications: they can be used to build password-management systems [ECSJR15], adaptive oblivious transfers [JL09], password-protected secret sharing [JKK14], and private set intersection [JL09], which can in turn be used for privacy-preserving contact discovery in messaging services [DRRT18] or for checking compromised credentials [LPASC+19]. For instance, the web browser Microsoft Edge uses an OPRF-based protocol to detect compromised passwords. Another practical use case of OPRFs is the privacy-preserving authorisation mechanism known as Privacy Pass [DGSTV18]. Developed and currently deployed by Cloudflare, Privacy Pass reduces the number of CAPTCHAs that users need to complete by issuing a number of tokens, which users can spend to avoid solving a second CAPTCHA. To prevent the server (i.e. Cloudflare, in this case) from tracking users across websites, the user queries must be oblivious. OPRFs are also used within OPAQUE [JKX18], a strong asymmetric password-

authenticated key exchange that allows a user and a server to authenticate each other without transmitting a password and with strong security guarantees. For these reasons, there are ongoing efforts to integrate OPAQUE into TLS 1.3[1]. Overall, OPRFs are a fundamental tool for developing privacy-preserving solutions, and they are set to be standardized by the Crypto Forum Research Group (CFRG) [DFSW23].

It is possible to build an OPRF using generic multi-party computation techniques, but such solutions can be inefficient, and they require more rounds of communication than what an ad-hoc construction can achieve. Indeed, highly-efficient and round-optimal (i.e., with two rounds) constructions exist based on Diffie-Hellman [JKK14] or RSA blind signatures [Cha82]. All such constructions are classical, and in the post-quantum setting, very few OPRFs are reported in the literature. The first quantum-secure verifiable OPRF was proposed by Albrecht et al. [ADDS21]. The protocol is based on the module learning with errors problem and the short integer solution problem in one dimension, and it only requires two rounds of communication. However, the construction can be characterized as a feasibility result, as a single OPRF execution requires communicating hundreds of gigabytes of data. The only other post-quantum solutions were proposed by Boneh, Kogan, and Woo [BKW20]. The authors proposed two moderately-compact OPRFs based on isogenies, one relying on SIDH and one on CSIDH. The protocol based on CSIDH is a non-verifiable, three-round OPRF, which is obtained by combining a Naor-Reingold PRF [NR97] with a CSIDH-based oblivious transfer protocol [LGD21] to make the PRF evaluation oblivious. The OPRF based on SIDH is verifiable, but requires an even higher number of communication rounds, since the verifiability proof is highly interactive. A follow-up work by Basso et al. [BKMPS21] cryptanalyzed the SIDH-based OPRF by demonstrating two attacks against the one-more unpredictability of the protocol, i.e. it showed that a malicious user can recover sufficient information to independently evaluate the PRF on any input. The first attack is polynomial-time, but it can be easily prevented with a simple countermeasure; the second attack is subexponential but still practical, and

[1] https://blog.cloudflare.com/opaque-oblivious-passwords/

the authors argue that there is no simple countermeasure against it. More recently, a series of works [CD23; MMPPW23; Rob23] proposed an efficient attack on SIDH that also extends to the OPRF.

Contributions. In this work, we propose an OPRF protocol that is post-quantum secure, verifiable, round-optimal, and moderately compact (≈2MB per execution), with a security proof in the UC framework [Can01] in the random-oracle model. To do so, we show that the same high-level design as the SIDH-based OPRF by Boneh et al. [BKW20] is a viable solution, with the following changes:

- We propose an efficient countermeasure against the one-more unpredictability attack by Basso et al. [BKMPS21]. We modify the PRF definition, and in particular we change how the user's input is mapped to an elliptic curve, to prevent an attacker from independently evaluating the PRF. A malicious user can still recover some information, as in the Basso et al. attack, but this is no longer sufficient to determine the PRF output. Besides preventing the attack, this change results in a smaller prime and a more compact protocol.

- We discuss how to integrate MSIDH, a recently-proposed countermeasure [FMP23a] against the SIDH attacks that relies on masked torsion, into the OPRF protocol. This requires using longer isogenies and a larger prime, but a series of optimizations allow us to maintain a reasonable communication cost. To integrate MSIDH, we also propose the first zero-knowledge proof of knowledge that can guarantee the correctness of a MSIDH public key, which may be of independent interest. The protocol proves correctness of masked torsion images by building an SIDH square, as in previous proofs, and revealing masked torsion points on each side of the SIDH square, so that the composition of the masking values along the SIDH square gives the secret masking value.

- We propose a novel proof of knowledge that can guarantee that two isogenies are parallel, i.e. they are computed by applying the same secret key to two starting

curves and torsion points. The protocol is obtained by evaluating two proofs of isogeny knowledge with correlated randomness. Such a protocol can be used by the server to show that it is using a previously-committed secret key, which is the key ingredient to make the OPRF verifiable. Since the proof is a proof of knowledge, it can be made non-interactive through standard transformations; this makes the proposed OPRF the first isogeny-based OPRF to be round-optimal.

Paper organization. In Section 5.2, we introduce the main components needed for the rest of the paper, including the SIDH and the Boneh et al. construction. In Section 5.3, we present the OPRF ideal functionality, together with the security notions and assumptions needed to implement it. Section 5.4 presents the novel countermeasures against the one-more unpredictability attack by Basso et al., while Section 5.5 discusses how to prevent the SIDH attacks, and Section 5.6 presents the new proof of parallel isogeny used to achieve verifiability. In Section 5.7, we put everything together to obtain the new OPRF protocol, estimate its communication complexity, and compare it with the existing solutions.

5.2 — Preliminaries

In this section, we present the notation used in the rest of the paper, and we briefly introduce the SIDH protocol, the recent attacks on SIDH, the OPRF construction by Boneh et al. [BKW20], and the attack on the protocol by Basso et al. [BKMPS21].

5.2.1 – SIDH

The Supersingular Isogeny Diffie-Hellman (SIDH) [JD11] is a key-exchange protocol based on isogenies between supersingular elliptic curves. For information on elliptic curves and isogenies, we refer the reader to [Sil86]. The protocol parameters include a prime p of the form $p = ABf - 1$, where A and B are smooth coprime integers and

f a small cofactor, a supersingular curve E_0 defined over $\mathbb{F}_{p^2}$, and the basis P_A, Q_A and P_B, Q_B that span, respectively, $E_0[A]$ and $E_0[B]$. One party samples a secret key $a \xleftarrow{\$} \mathbb{Z}_A$, computes the isogeny $\phi_A : E_0 \to E_A \coloneqq E_0/\langle P_A + [a]Q_A \rangle$, and publishes $\mathsf{pk}_A = (E_A, R_A = \phi_A(P_B), S_A = \phi_A(Q_B))$. The second party, proceeds similarly by sampling a secret key $b \xleftarrow{\$} \mathbb{Z}_B$, computing $\phi_B : E_0 \to E_B \coloneqq E_0/\langle P_B + [b]Q_B \rangle$, and revealing $\mathsf{pk}_B = (E_B, R_B = \phi_B(P_A), S_B = \phi_B(Q_A))$. Then, both parties can agree on a shared secret by translating their secret isogeny to the starting curve in the other party's public key using the revealed torsion information. In other words, the first party computes $\phi'_A : E_B \to E_{AB} \coloneqq E_B/\langle R_B + [a]S_B \rangle$, and the second party computes $\phi'_B : E_A \to E_{BA} \coloneqq E_A/\langle R_A + [b]S_A \rangle$. The codomain curves E_{AB} and E_{BA} are isomorphic, and thus their j-invariant is the same and provides the shared secret of the key exchange. Note that it is possible to represent the points in the public keys more compactly than two x-coordinates, which requires $4 \log p$ bits. If the points are expressed in terms of a canonical basis , i.e. a basis deterministically computed from the curve, their coefficients only require $4 \log A$ or $4 \log B$ bits [CLN16]. In the rest of the paper, we write $P, Q = \mathcal{B}_N(E)$ for a canonical basis of order N on E. We also refer to the setup described above as the *SIDH square* (E_0, E_A, E_B, E_{AB}) with edges $(\phi_A, \phi_B, \phi'_A, \phi'_B)$.

Generally, isogenies do not commute, which means that two parties computing an SIDH-like exchange would not agree on a shared secret if they only revealed the isogeny codomain. To avoid the problem, SIDH reveals the image of a torsion basis that allows each party to translate their isogeny such that they commute. Torsion points are thus a key element of the SIDH protocol, but they also allow attackers to perform adaptive attacks against static-key SIDH [GPST16]. To prevent such attacks, both parties can provide a proof of torsion point correctness, such as the proof proposed in [BKW20; DDGZ22]. Unfortunately, revealing the torsion point images also enabled the recent passive attacks on SIDH.

The SIDH attacks. The security of the SIDH protocol hinges on the hardness of recomputing the secret isogenies given the public key. While the problem of finding an isogeny between two curves is believed to be hard, the presence of torsion point images in SIDH makes it easier since more information is revealed about the secret isogeny. In a series of works by Castryck and Decru [CD23], Maino, Martindale, Panny, Pope, and Wesolowski [MMPPW23], and Robert [Rob23], the authors propose a polynomial-time algorithm that can compute an isogeny of smooth degree d given the domain and codomain curves, the degree d, and the image of a torsion basis of order at least $\sqrt{d}$. This fully breaks the SIDH key exchange and all protocols based on it. Some counteremasures have been proposed [FMP23a], based on either secret-degree isogenies or on masked torsion images. We discuss these approaches in the context of the OPRF protocol in Section 5.5.

5.2.2 – The Boneh et al. OPRF construction

Boneh, Kogan, and Woo [BKW20] introduced a verifiable OPRF protocol based on SIDH, which uses a prime p of the form $p = N_{\mathsf{M}}N_{\mathsf{B}}N_{\mathsf{K}}N_1N_2f - 1$, where the values N_i are coprime smooth integers and f is a small cofactor. Initially, the server commits to its key k by publishing the curve E_C obtained as the codomain of the N_{K}-isogeny starting from $\tilde{E}$ with kernel $\langle \tilde{P} + [k]\tilde{Q} \rangle$, where the values $\tilde{E}, \tilde{P}, \tilde{Q}$ are protocol parameters. The commitment also include a zero-knowledge proof π_C of the correctness of the computation. Then, to evaluate the PRF on input $m \in \mathcal{M}$, where $\mathcal{M}$ defines the input space, the user computes an isogeny ϕ_m of degree N_{M} by hashing the input with a function H and computing $\phi_m : E_0 \to E_m \coloneqq E_0/\langle P + [H(m)]Q \rangle$, where the curve E_0 and the points P, Q are also protocol parameters. Then, the user blinds the curve E_m by computing a second isogeny $\phi_b : E_m \to E_{mb}$ of degree N_{B}. The user sends the curve E_{mb} and the torsion images $R_K = \phi_b \circ \phi_m(P_K), S_K = \phi_b \circ \phi_m(Q_K)$ to the server, where the points P_K, Q_K are also protocol parameters of order N_{K}. The user also provides a non-interactive zero-knowledge proof that torsion information was honestly computed. The server validates the proof, computes the isogeny $\phi_k : E_{mb} \to E_{mbk} \coloneqq E_{mb}/\langle R_K + [k]S_K \rangle$ based on its

secret key k, and sends the curve E_{mrk}, the image $\phi_k(E_{mb}[N_B])$, and a non-interactive zero-knowledge proof of correctness to the user. Then, the server and the user engage in an interactive protocol where the server proves that the isogeny ϕ_k has used the same key k as the committed value. If the user is convinced, they use the provided torsion information to undo the blinding isogeny, i.e. to compute the translation of the dual of the blinding isogeny, to obtain the curve E_{mk}. The output of the OPRF is then the hash $H\big(m, j(E_{mk}), (E_C, \pi_C)\big)$. The main exchange, without the commitments and the proofs, is represented in Fig. 5.1.

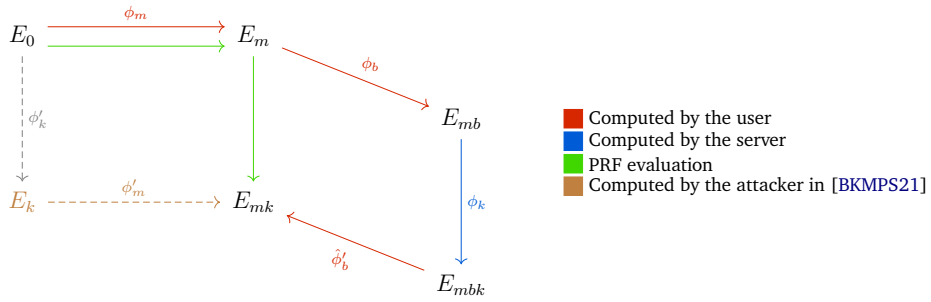

Figure 5.1: The Boneh et al. OPRF, based on [BKMPS21, Fig. 1].

5.2.3 – The Basso et al. attack

Basso et al. [BKMPS21] proposed two attacks against the one-more unpredictability of the Boneh et al. [BKW20] OPRF. In the first attack, an attacker who acts as a malicious user engages in the OPRF with a message isogeny ϕ_m with kernel generator a point M, of order ℓ^e. The attacker repeats the process with message isogenies with kernel generators $[\ell]M, [\ell^2]M, \ldots, [\ell^e]M$. The outputs of the PRF are the curves that lie on the isogeny path of $\phi'_m : E_k \to E_{mk}$ (see Fig. 5.1), which allows the attacker to compute a generator of the kernel of such isogeny. The recomputed generator is a scalar multiple $\phi'_k(M)$, where ϕ'_k is the isogeny parallel to the server's secret isogeny, i.e. its kernel is generated by $P_k + [k]Q_k$. By repeating this process three times with points M_1, M_2 and $M_3 \coloneqq M_1 + M_2$ (where M_1

and M_2 are linearly independent), the attacker obtains

$$R \coloneqq [\alpha]\phi'_k(M_0), \quad S \coloneqq [\beta]\phi'_k(M_1),$$
$$T \coloneqq [\gamma]\phi'_k(M_3) = [\gamma/\alpha]R + [\gamma/\beta]S,$$

for some unknown values α, β, γ. By expressing T in terms of R and S, the attacker obtains the values γ/α and γ/β and the points $R' \coloneqq [\gamma/\alpha]R = [\gamma]\phi'_k(M_0)$ and $S' \coloneqq [\gamma/\beta]S = [\gamma]\phi'_k(M_1)$. Finally, the attacker can evaluate the PRF on any input m by computing $E_K/\langle R' + [H(m)]S' \rangle$. The attack is polynomial time, but it crucially rely on using message isogenies ϕ_m of varying degree. The attack can be thwarted by server checking the order of the isogeny ϕ_m, which is possible because of the proof of knowledge provided by user.

The authors of [BKMPS21] also propose a second attack that cannot be easily prevented. The attack proceeeds in a similar way to the previous one, but the malicious user uses only isogenies of full degree. To obtain the curves on the path of ϕ_m, the attacker needs to find the middle curve between two PRF outputs. This introduces a trade-off between the complexity of the attack and the number of queries. Minimizing both yields a subexponential yet practical attack on the one-more unpredictability of the protocol.

5.3 — Oblivious pseudorandom functions

The security properties of an OPRF can be hard to define. Oblivious pseudorandom functions were originally proposed by Naor and Reingold [NR97], who defined an OPRF via an ideal functionality. Subsequent work [FIPR05; JL09] defined OPRFs in terms of the two-party computation $(k, x) \mapsto (\perp, f(k, x))$, but such a definition has several drawbacks. On one side, it is hard to build protocols that satisfy such a definition, because the security proof would require extracting the user's input, while at the same the definition is not secure enough, because it does not guarantee any security under composability. Since OPRFs are mainly used as builiding blocks in larger protocols, such a security guarantee

is highly needed. For these reasons, Jarecki et al. [JKK14; JKKX16; JKX18] proposed new definitions in the UC framework [Can01]. To avoid extracting the user's input, the ideal functionality introduces a ticketing system that increases a counter when the PRF is evaluated and decreases the counter when the user receives the PRF output. This captures the idea that a malicious user should learn only the PRF output for one input for each interaction. This results in the definition of Fig. 5.2, which is based on the definitions by Jarecki et al. [JKK14; JKKX17; JKX18].

Parameters: The PRF output ℓ, polynomial in the security parameter λ.

Convention: For every identifier S, the counter $\mathsf{tx}[S]$ is initially set to zero. For every value $\pi \in \{0,1\}^*$ and $x \in \{0,1\}^*$, the value $F(\pi, x)$ is initially undefined, and whenever such a value is referenced, the functionality assigns a random ℓ-bit string $F(\pi, x) \xleftarrow{\$} \{0,1\}^\ell$.

Initialization

- On message INIT from party S, forward (INIT, S) to the adversary $\mathcal{A}$.
- On message (PARAM, S, π) from adversary $\mathcal{A}$, and if $\mathsf{param}[S]$ is undefined, then set $\mathsf{param}[S] = \pi$.

Evaluation

- On message (EVAL, S, x) from $P \in \{U, \mathcal{A}\}$, record $\langle P, x\rangle$ and forward the message (EVAL, P, S) to $\mathcal{A}$.
- On message SERVERCOMPLETE from server S, send $(\text{SERVERCOMPLETE}, S)$ to $\mathcal{A}$ and increment $\mathsf{tx}[S]$.
- On message $(\text{USERCOMPLETE}, P, \pi)$ from $\mathcal{A}$, retrieve the record $\langle P, x\rangle$, delete it from the list of records, and decrement $\mathsf{tx}[S]$ if there exists an honest server S such that $\mathsf{param}[S] = \pi$; abort if no such record exists or if $\mathsf{tx}[S] = 0$. Then, send $(\text{EVAL}, \pi, F(\pi, x))$ to P.

Figure 5.2: Functionality $\mathcal{F}_{\mathsf{vOPRF}}$.

5.3.1 – Security assumptions

To prove that the OPRF protocol we propose implements the functionality of Fig. 5.2, we will make use of the properties listed in this section. Since our protocol and security

proof follows the same high-level structure as that of the OPRF protocol by Boneh et al. [BKW20], these properties are also based on those of the augmentable commitment framework proposed in [BKW20]. Unlike [BKW20], we avoid the abstraction of augmentable commitments due to its restrictiveness (the counteremasures of Section 5.4 would not be possible within that framework), and we prefer an explicit description throughout this work.

Correctness. Firstly, we require the OPRF to be correct, i.e. the output of the protocol is the output of function that deterministically depends only on the user's input and the server's secret key. In other words, we want that the blinding process that guarantees the obliviousness of the user's input does not affect the final output. In the context of our protocol, we want that the unblinding isogeny undoes the effect of the blinding isogeny. This is contained in the following lemma, whose proof follows from the correctness of the SIDH protocol [JD11].

Lemma 16 (Correctness)**.** *Let p be a prime of the form $p = N_B N_K f - 1$, where N_B, N_K, f are smooth coprime integers. Let E_0 be a supersingular elliptic curve defined over $\mathbb{F}_{p^2}$ and let P_B, Q_B and P_K, Q_K be respectively a basis of $E_0[N_B]$ and $E_0[N_K]$. Let also b and k be two values in $\mathbb{Z}_{N_B}$ and $\mathbb{Z}_{N_K}$. Then, consider the isogenies*

$$\phi_B : E_0 \to E_B \coloneqq E_0/\langle P_B + [b]Q_B \rangle,$$
$$\phi_K : E_0 \to E_K \coloneqq E_0/\langle P_K + [k]Q_K \rangle,$$
$$\phi'_k : E_B \to E_{BK} \coloneqq E_B/\langle \phi_B(P_K) + [k]\phi_B(Q_K) \rangle.$$

If R_B, S_B is a basis of $E_B[N_B]$ and the values b_0, b_1 satisfy $\ker \hat{\phi}_B = \langle [b_0]R_B + [b_1]S_B \rangle$, then we have

$$j\left(E_{BK}/\langle [b_0]\phi'_k(R_B) + [b_1]\phi'_k(S_B) \rangle\right) = j(E_K).$$

Input hiding. To ensure that the OPRF is oblivious, we want that the server does not learn the user's input. That holds in the strongest sense, i.e. the server should not learn

the user's input even when the input is randomly chosen between two inputs *chosen by the server*. In other words, the user must apply a blinding step that fully hides the chosen input. In the context of isogenies, we want the following problem to be hard.

Problem 1. *Let p be a prime of the form $p = N_B N_K f - 1$, where $N_B N_K$, f are smooth coprime integers. Let E_0 and E_1 be two supersingular elliptic curves defined over $\mathbb{F}_{p^2}$ and chosen by the adversary, and let P_0, Q_0 and P_1, Q_1 be a basis of $E_0[N_K]$ and $E_1[N_K]$, respectively, such that $e_{N_K}(P_0, Q_0) = e_{N_K}(P_1, Q_1)$. Let i be a random bit, i.e. $i \xleftarrow{\$} \{0, 1\}$, and B a random point in $E_i[N_B]$, and write $\phi : E_i \to E' := E_i/\langle B \rangle$. Output i given E' and $f(\phi(P_i), \phi(Q_i))$, where the latter is some auxiliary torsion information.*

The hardness of the problem clearly depends on the function f; if the torsion images were directly revealed, Problem 1 would be easy due to the SIDH attacks. We thus delay specifying the function f until Section 5.5, where we discuss the SIDH counteremasures to use within the OPRF protocol. In the section, we also state the variant of the Decisional Isogeny problem that Problem 1 reduces to.

One-more unpredictability. A key property of an OPRF is that the user learns the output of the PRF only on its input of choice. That means that a malicious user should not learn the output on more inputs than the number of OPRF executions. The attack by Basso et al. [BKMPS21] on the Boneh et al. [BKW20] OPRF targets the one-more unpredictability, since it shows that a malicious user can extract enough information to independently evaluate the OPRF on any input of their choice. We propose an efficient counteremasure against the one-more unpredictability attack in the next section; we thus delay until then a formalization of the isogeny-related assumption (see Problem 4) we need to guarantee the one-more unpredictability of the OPRF protocol.

Commitment binding. At the beginning of the OPRF protocol, the server committs to a secret key k, so that during each OPRF execution it can prove that the same key was used. To guarantee verifiability, we want a commitment scheme with an associated proof

of input reuse. We propose to commit to a key k by fixing a special curve $\tilde{E}$ with a basis $\tilde{P}, \tilde{Q}$ of $\tilde{E}[N_{\mathsf{K}}]$ and revealing $j(\tilde{E}/\langle \tilde{P} + [k]\tilde{Q} \rangle)$. The proof of input reuse, which in the context of isogenies becomes a proof of parallel isogenies, is presented in Section 5.5.2. To guarantee that the Commitment is binding, we want that the following problem to be hard.

Problem 2 (Collision finding problem)**.** *Let E_0 be a supersingular elliptic curve of unknown endomorphism ring. Find two distinct isogenies $\phi_0 : E_0 \to E$ and $\phi_1 : E_0 \to E'$ such that $j(E_0) = j(E_1)$.*

Problem 2 has been studied in the context of the CGL hash function [CLG09], and it has been shown to be heuristically equivalent to the following problem, which underpins every isogeny-based protocol [PL17; EHLMP18].

Problem 3 (Endomorphism Ring problem)**.** *Let E be a supersingular elliptic curve. Find the endomorphism ring $\mathrm{End}(E)$ of E.*

5.4 — Countermeasures against the one-more unpredictability attack

The original protocol by Boneh et al. starts by mapping an input m to an isogeny ϕ_m. If we denote with N_{M} the torsion space dedicated to the message, the protocol fixes a basis P, Q of $E_0[N_{\mathsf{M}}]$ and computes the isogeny ϕ_m given by

$$\phi_m : E_0 \to E_0/\langle P + [H(m)]Q \rangle =: E_m, \tag{5.1}$$

where $H(\cdot)$ maps the message m onto an element of $\mathbb{Z}_{N_{\mathsf{M}}}$.

The subexponential attack [BKMPS21] recovers the image P_k, Q_k of the torsion basis P, Q, up to scalar multiplication, under the secret isogeny $\phi'_k : E_0 \to E_k$. With such information, the attacker can evaluate the PRF on any input of their choice. The output

curve of the PRF is the curve computed as $E_k/\langle P_k + [H(m)]_k \rangle$. Any countermeasures against such an attack need to prevent the attacker from evaluating the OPRF without interacting with the server. A first approach might try to prevent the attacker from recovering the points P_k, Q_k altogether, but it appears to be hard since the curve E_k is fixed, because the server needs to hold a long-term static key to satisfy the OPRF definition, which in turn also fixes the curve E_k. Moreover, the attack by Basso et al. could be prevented by requiring the user to send only honestly-generated queries. The attacker needs to send carefully-chosen queries, where the kernel of the message isogeny does not necessarily satisfy Eq. (5.1). However, there is no simple way to prove in zero-knowledge that the kernel was honestly computed, besides using very expensive generic techniques. Another approach might require to simply increase the parameters. The attack is subexponential, and it is possible to obtain λ bits of security if the isogeny ϕ_m has degree 2^{λ^2} (this can be reduced if we limit the number of queries the attacker can make). This would require using very long isogenies (the degree would be $2^{16,384}$ for $\lambda = 128$) and very large primes.

Instead, in this section we propose a novel and efficient countermeasure that sidesteps these issues. Our main idea is to accept that an attacker may recover the curve E_k and points P_k, Q_k on it, but to prevent those points from being sufficient to evaluate the desired isogeny. To do so, we require that the isogeny ϕ_m has an irrational kernel, i.e. its kernel is defined over a sufficiently-large extension field. Such an isogeny can be efficiently computed as a composition of rational isogenies. More formally, assume that $N_{\mathsf{M}} = \ell^e$, and e is the highest power of ℓ that divides $p + 1$. Then, given an input $m \in \mathcal{M}$, we compute the isogeny ϕ_m in the following way:

1. We first map the message m to two elements in $\mathbb{Z}_{\ell^e}$ through two hash functions H_0, H_1 that are collision resistant. We thus have $m_0 = H_0(m)$ and $m_1 = H_1(m)$.

2. Given the starting curve E_0 and two points P_0, Q_0 spanning $E_0[\ell^e]$, we compute the isogeny

$$\phi_0 : E_0 \to E_1 \coloneqq E_0/\langle P_0 + [m_0]Q_0 \rangle.$$

3. We determine a canonical basis P_1, Q_1 of $E_1[\ell^e]$ and compute the isogeny

$$\phi_1 : E_1 \to E_m := E_1/\langle P_1 + [m_1]Q_1 \rangle,$$

4. The isogeny $\phi_m : E_0 \to E_m$ is the composition $\phi_1 \circ \phi_0$.

An attacker may still try to apply the one-more unpredictability attack. In the original case, the attacker recovers three isogenies from E_k to E_{mk} and they combine their kernel generators to obtain the image points P_k, Q_k. In the proposed construction, the attacker can still recover three (or more) isogenies from E_k to E_{mk}. However, the kernel generators of these isogenies are points of order ℓ^{2e}, and thus they are defined only over the extension field $\mathbb{F}_{p^{2\ell^e}}$. This is an exponentially large field, and even just representing such a point—let alone doing any computation—would be exponential in the security parameter. To guarantee security, it is important that the degree of ϕ_m is a prime power. If the degree were a product of prime powers, it is possible to represent a large extension by working over several smaller extensions because of the Chinese Remainder Theorem. This can reduce the complexity of working over a large extension and thus reduce the security of the proposed countermeasures.

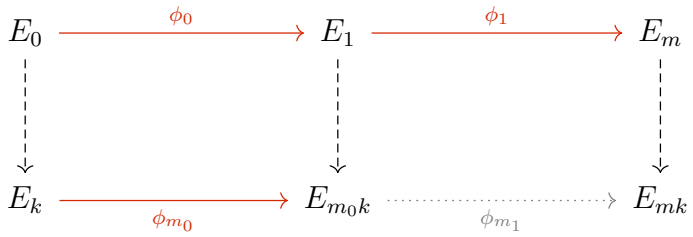

Figure 5.3: Summary of the proposed countermeasure (this does not depict the blinding/unblinding phase). Isogenies in red are known or can be computed by the attacker, isogenies in black are unknown to the attacker, and the dotted isogeny represents the missing isogeny that the attacker needs to compute to succeed in the attack.

The attacker can work with the kernel generators of only the first half of the isogenies and obtain a basis P_k, Q_k of order ℓ^e (see Fig. 5.3). This allows them to evaluate the first isogeny ϕ_{m_0} to obtain the curve E_{m_0k} for any message m. However, the attacker has no way of computing the remaining isogeny ϕ_{m_1}. To do so, the attacker would need to map

the canonical basis on E_1 to E_{m_0k}, which does not seem to be possible without knowing the server secret key. Alternatively, the attacker could map the points P, Q and P_k, Q_k under the isogenies ϕ_0 and ϕ'_k. At least one of the image points on each curve has full order, and the point of full order on E_{m_0k} is the image of the point of full order on E_1. This suggest such an approach could be used to find a basis, but the second point on each curve is always a scalar multiple of the first point[2]. Hence, guessing the remaining point has exponential complexity ℓ^e. Lastly, the attacker cannot use a similar strategy as the one-more unpredictability attack to recover a basis on E_{m_0k} because the curve E_{m_0k} depends on the message m. It thus changes at every interaction, and it is hard for an attacker to find two messages that have the same first curve E_1 and E_{m_0k} since we assume that the hash function H_0 is collision-resistant. Note that we require H_0 and H_1 to be collision-resistant, but we conjecture that only H_0 needs to be. Overall, the knowledge of E_{m_0k} does not help the attacker learn any information on the curve E_{mk}, which successfully prevents the the one-more unpredictability attack.

Optimizations. We can extend this approach to obtain a more compact protocol. Rather than limiting ourselves to two isogenies, we can extend this to an arbitrary number. Let I be an integer greater than one, and let H_i be distinct functions for every $i \in \{1, \dots, I\}$, which are modelled as random oracles. Then, given an input m and a starting curve E_0, the isogeny ϕ_m and the curve E_m can be computed as in Algorithm 3. This modification can result in a more compact OPRF protocol because only the smaller isogenies ϕ_i need to be defined over $\mathbb{F}_{p^2}$; thus, using more isogenies can result in a smaller prime p while maintaining the same degree of the isogeny ϕ_m. In this case, note that the functions H_i cannot be collision-resistant if their output space becomes smaller than $2^{2\lambda}$; however, the case where $I > 2$ is clearly more secure than $I = 2$ because an attacker can recover even less information. Since we require H_0 to be collision-resistant when $I = 2$, in the case $I > 2$ it is thus sufficient to ask that the concatenation $H_0(x)||H_1(x)|| \dots ||H_n(x)$ is

[2] If $\ker \phi = \langle P + \alpha Q \rangle$, it follows that $\phi(P) = -\alpha\phi(Q)$.

collision-resistant where n is the smallest value such that the concatenation output is larger than $2^{2\lambda}$. In other words, the functions H_i can be obtained by splicing the output a collision-resistant hash function.

In the rest of the paper, we write $(\phi_m, E_m) = \mathcal{H}_I(x)$ to refer to the function in Algorithm 3; we also write $[P_0, P_1, \ldots, P_{I-1}]_{E,N}$ to denote a list points of order N where the point P_0 belongs to E, and the point P_i belongs to $E_i := E_{i_1}/\langle P_{i-1}\rangle$. We refer to this as a *sequence*, whose associated isogeny is the composition of the isogenies $E_i \to E_i/\langle P_i\rangle$.

Algorithm 3 Function $\mathcal{H}_I$ mapping the input m to the curve E_m

for $i \leftarrow 0$ to $I-1$ **do**
    Set $m_i = H_i(m)$;
    Set $P_i, Q_i = \mathcal{B}_M(E)$;
    Compute $\phi_i : E_i \to E_{i+1} := E_i/\langle P_i + [m_i]Q_i\rangle$;
Set $\phi_m = \phi_{I-1} \circ \ldots \circ \phi_0$;
Set $E_m = E_{I-1}$;
return ϕ_m, E_m;

A new assumption. We proposed a modified protocol that prevents the existing one-more unpredictability attacks. As in the original construction, the one-more unpredictability of the resulting protocol relies on the hardness of a novel problem, which is the following.

Problem 4 (One-more unpredictability)**.** *Let p be a prime of the form $p = N_M N_K f - 1$, where N_M and N_K are smooth coprime integers, and f a cofactor. Let $\mathcal{H}_I$ be a function as in Algorithm 3. Let E_0 be a supersingular curve defined over $\mathbb{F}_{p^2}$, and let K be a point on E_0 of order N_K. Write ϕ_K for the isogeny $\phi_K : E_0 \to E_K := E_0/\langle K\rangle$. Given the curves E_0, E_K and an oracle that responds to the following queries:*

- *challenge: returns a random sequence $[M_0, \ldots, M_{I-1}]_{E_0,N_M}$,*
- *solve$([V_0, \ldots, V_{I-1}]_{E_0,N_M}$): returns $j(E_V/\langle\phi_V(K)\rangle)$, where ϕ_V is the isogeny associated to the input sequence,*
- *decide(i, j): returns true if j is equal to the output of a solve query with input the response of the i-th challenge query, and false otherwise,*

For any value n, produce n pairs (i, j) such that decide$(i, j) =$ true with less than n solve queries.

The problem is based on Game 12 of [BKW20], but compared to it, this game involves multiple points during the challenge and solve query to abstract the behavior described in the previous section. Moreover, the problem includes the countermeasures against the polynomial time attack of [BKMPS21], i.e. the attacker can only query points of the correct order. This can be replicated in the OPRF setting by checking the order of the isogenies in the proof of isogeny knowledge. We included these countermeasures to prevent possible attacks since they are inexpensive. However, we conjecture that the problem remains hard even if the adversary is allowed to submit solve queries with points of arbitrary order. Furthermore, the problem remains hard after the SIDH attacks since it does not involve exchanging any torsion points.

Countermeasure costs. We briefly discuss the impact of the proposed countermeasures on the performance of the OPRF protocol. Firstly, we need to determine the parameters ℓ, e, and I. In the previous section, we presented two possible attacks on the Auxiliary One-More SIDH assumption: the first requires to work over the extension field $\mathbb{F}_{p^{2\ell^e}}$, while the second obtains a point of full order on E_1 and its image on $E_{m_0 k}$, fixes a linearly-independent point on E_1, and then guesses its image $E_{m_0 k}$. The first attack involves using points with coordinates with ℓ^e values in $\mathbb{F}_p$, while the second requires guessing the correct image out of the ℓ^e possibilities. It may be tempting to set $\ell^e \approx 2^\lambda$, but when $I = 2$ we require that the hash functions H_0, H_1 are collision-resistant, so their output space must be larger than $2^{2\lambda}$. Hence, we set $\ell^e \approx 2^{2\lambda}$ and the degree of ϕ_m to be $2^{4\lambda}$. If we want to minimize the bandwidth consumption of the protocol, we can set $e = 1$ and $\ell^I \approx 2^{4\lambda}$. Moreover, we can choose ℓ such that isogenies of degree ℓ are defined over a small extension, such $\mathbb{F}_{p^4}$, rather $\mathbb{F}_{p^2}$.

The choice of e thus determines the size of the message component N_{M} and the prime p: if $I = 2$, the message component N_{M} is already smaller than the value N_{M} in the

original construction, which used $N_{\mathsf{M}} \approx 2^{5/2\lambda}$. If $e = 1$, the message component N_{M} is one, since the prime p does not need to change to allow computations of the message isogeny. This means that not only do the proposed countermeasures protect against existing attacks, but also they reduce the prime size leading to a more compact and efficient protocol.

5.5 — Countermeasures against the SIDH attacks

The recent series of attacks by Castryck and Decru [CD23], Maino, Martindale, Panny, Pope, and Wesolowski [MMPPW23], and Robert [Rob23] exploits torsion-point information to break SIDH. These attacks trivially translate to the OPRF, where any third party can recover both the user's hashed input (which breaks obliviousness) and the server's secret key. In this section, we discuss how to adapt the existing SIDH countermeasures to work in the OPRF setting. After modifying the main exchange, we propose a novel proof of isogeny knowledge that works together with the countermeasures, which may be of independent interest since it is the first proof to prove the correctness of torsion point images in the SIDH-with-countermeasure setting. This proof can be used together with the patched SIDH to obtain a post-quantum non-interactive key-exchange.

Combining the countermeasures together with the novel proof of torsion point correctness, we obtain an SIDH-based OPRF that is resistant against the SIDH attacks. While the countermeasures impose larger parameters, the resulting protocol remains the most compact post-quantum vOPRF.

5.5.1 – Protecting the exchange

The OPRF exchange is based on SIDH, but it has some differences from a simple SIDH key exchange. In particular, in the OPRF protocol the two parties compute isogenies of different lengths and need to prove the correctness of their outputs. Moreover, the server starts its computation from a curve provided by the user and also needs to prove that it

used the same key it has previously committed.

The attacks on SIDH recover an isogeny $\phi : E \to E'$ of degree d when provided with the curves E, E', the degree d and the image of the n-torsion $\phi(E[n])$, where the size of n satisfies at least $n \approx d^{1/2}$. This suggests three possible countermeasures, as discussed in [FMP23a]:

1. Hide the degree d of the isogeny ϕ by choosing a random $d' \mid d$.

2. Increase the degree d of the isogeny ϕ, such that $d \gg n^2$,

3. Mask the exact torsion images by providing a scalar multiple.

In the OPRF setting, the hidden-degree countermeasure does not appear to work. Both parties need to prove the correctness of their revealed torsion points, and all the proofs of isogeny knowledge in the literature rely on constructing an SIDH square and revealing some sides. This inevitably leaks the degree of the secret isogeny, which makes the hidden-degree countermeasure ill-suited to work with zero-knowledge proofs.

Relying only on longer isogeny may seem like a valuable approach, as it does not require on any new assumption. In the SIDH setting, it is not possible to protect both parties with such a strategy because it would require both isogenies to be longer than the other. In the OPRF, however, the user computes the message and blinding isogenies in the first round, provides enough torsion information for the server to compute its isogeny, and the server reveals the torsion information needed for the user to invert the blinding isogeny. This suggests that the protocol could be secure if the server's isogeny is sufficiently longer than the blinding isogeny, and if the composition of the message and blinding isogeny is sufficiently longer the server's isogeny. This approach could lead to a compact and fairly efficient protocol, but unfortunately it does not guarantee the input hiding property. The server should not distinguish the user's input even when the user chooses between two server-controlled messages. This approach is thus inadequate for an OPRF, but it might still be useful for specific applications where the message space has sufficiently-large entropy and such a strong security assumption is not needed.

Thus, to guarantee the security of the SIDH-based OPRF we need to rely on the masked-torsion countermeasure, as in masked SIDH (M-SIDH) [FMP23a]. Let $\phi : E \to E'$ be the isogeny we want to protect, and let P, Q be a basis of $E[n]$, for some n coprime with d. Given a basis $P' = \phi(P), Q' = \phi(Q)$, the other party computes their isogeny with kernel $\langle P' + [x]Q' \rangle$, where x is the secret key. Thus, it is possible to reveal $[\alpha]P', [\alpha]Q'$, for some random α coprime with the torsion order n, without affecting the correctness of the protocol. However, an attacker can recover the value α^2 from the Weil pairing, since $e([\alpha]P', [\alpha]Q') = e(P, Q)^{\alpha^2 \deg \phi}$. To ensure that the attacker cannot recover the value α, we want that any value has at least 2^λ square roots modulo n, hence n needs to be the product of at least λ prime powers. This, however, is not enough to guarantee security, as an attacker can guess the correct square root modulo some $n' \mid n$ with $n' > d^{1/2}$ in less than $\mathcal{O}(2^\lambda)$ guesses. We thus also require that $d > n'$, where n' is the product of the powers of the λ largest primes dividing n. From now on, we write $n = f_{\mathsf{MSIDH}}(\lambda, d)$ to denote the smallest value n that can guarantee λ bits of security when used in M-SIDH with an isogeny of degree d. Lastly, the countermeasure analysis in [FMP23a] shows that an attack is possible for certain parameters when the starting curve has a small endomorphism. In our case, such an attack does not apply even if the OPRF starting curve E_0 has a known endomorphism ring with a small endomorphism ι. The composition of the message and blinding isogeny $\phi_x \circ \phi_m$ is sufficiently long that the attack does not apply, while considering the blinding isogeny ϕ_x alone (remember that in the security game the attacker can control the messages) does not help either. Even if the attacker can guess the input message, the smallest endomorphism known on the domain of the blinding isogeny is $\hat{\phi}_m \circ \iota \circ \phi_m$, which is too large. The server computes its isogeny starting from a curve E_{mx} that is sent by the user, which generally could be an avenue for attack since MSIDH is insecure for special starting curves. However, the user also submits a proof that the user knows an isogeny of long degree between E_0 and E_{mx}. This guarantees that the smallest known endomorphism is again sufficiently large, and thus the attack does not apply to the server's isogeny as well.

We can now formulate the following problem, on whose hardness the input hiding property of the OPRF is based.

Problem 5 (Decisional M-SIDH isogeny problem)**.** *Let E_0 be a supersingular elliptic curve, with a basis P, Q be of $E_0[n]$. Distinguish between the following distributions:*

- *(E_1, R, S), where E_1 is the codomain of a d-isogeny $\phi : E_0 \to E_1$, where d is coprime with n, and the points R, S are the masked images of P, Q, i.e. $R = [\alpha]\phi(P)$ and $S = [\alpha]\phi(Q)$ for some $\alpha \xleftarrow{\$} \mathbb{Z}_n^*$;*
- *(E_1, R, S), where E_1 is a random supersingular elliptic curve and the points R, S are a random basis of $E_1[n]$ such that $e(R, S) = e(P, Q)^{\alpha^2 d}$, for some value α.*

The hardness of the problem clearly depends on the choices of n and d; the problem (conjecturally) requires $O(2^\lambda)$ operations to solve when $n > f_{\mathsf{MSIDH}}(\lambda, d)$, i.e. the product of the λ largest prime powers dividing n is smaller than $\sqrt{d}$.

Concrete cost. We have shown it is possible to protect the OPRF protocol from the SIDH attacks. Unfortunately, the proposed countermeasure do come at a significant cost. The degrees of the blinding isogeny and the server's isogeny are the same as in SIDH with the same countermeasures. At security level $\lambda = 128$, that corresponds to isogenies of degree $\approx 2^{2956}$. More generally, we see experimentally that the degree of the isogenies scales log-linearly in the security parameter with a constant of ≈ 6.7. We thus have that the degree of the blinding isogeny and the server's isogeny must be $\approx 2^{6.7\lambda \log \lambda}$ to guarantee the security of the protocol.

5.5.2 – Adapting the proof of isogeny knowledge

In the previous section, we showed how it is possible to protect the OPRF against the SIDH attacks using masked torsion points. However, in the OPRF protocol both parties need to prove the correctness of their torsion images to prevent adaptive attacks and guarantee the verifiability of the execution. This leads to an issue, because both the user

and the server want to prove that their torsion points were honestly generated, but these points are also scaled by a secret value. Thus, two parties want to prove that both points were honestly generated and scaled by the same value.

In this section, we propose a zero-knowledge proof of isogeny knowledge that can guarantee the correctness of torsion points up to a scalar, i.e. a proof for the following relation:

$$\mathcal{R}_{\mathsf{iso}} = \left\{ ((E_0, P_0, Q_0, E_1, P_1, Q_1), (\phi, \alpha)) \;\middle|\; \begin{array}{c} \phi : E_0 \to E_1 \text{ is a cyclic } d\text{-isogeny}, \\ P_1 = [\alpha]\phi(P_0), \\ Q_1 = [\alpha]\phi(Q_0). \end{array} \right\}.$$

In the literature, we can find two proofs of isogeny knowledge that also guarantee the correctness of torsion point images. The first proof constructs an SIDH square and explicitly maps the torsion images through all the sides of the square. This proof was proposed by Boneh et al. [BKW20] for the OPRF protocol, based on a previous idea by Galbraith [Gal18]. The second proof [DDGZ22], instead, is an extension of the simpler proof of isogeny knowledge by De Feo and Jao [JD11], but it does not seem to be malleable enough to support masked torsion, because the correctness of the torsion is implicitly proven through the existence of two parallel SIDH squares. In the first proof, however, the torsion images are more explicit, which makes it more suitable to support masked torsion. We thus propose a new proof based on the same approach as [BKW20] and [Gal18], although with some notable differences.

The main idea is that the masking constant α can be split into three shares $\alpha = \alpha_1\alpha_2\alpha_3$. The prover can mask the torsion points with α_i when computing the i-th side of the SIDH square, so that the composition of the three side isogenies, together with their masking values, forms a commutative diagram with the isogeny ϕ with masking value α. The proof remains zero-knowledge because each single value α_i is independent of α. More formally, let E_0 and E_1 be supersingular elliptic curves with points $P_0, Q_0 \in E_0[n]$ and $P_1, Q_1 \in E_1[n]$. The prover wants to prove knowledge of a d-isogeny $\phi : E_0 \to E_1$ and

a value $\alpha \in \mathbb{Z}_n$ such that $P_1 = [\alpha]\phi(P_0)$ and $Q_1 = [\alpha]\phi(Q_0)$. This only makes sense if ϕ is secret, thus let us assume $n = f_{\mathsf{MSIDH}}(\lambda, d)$. The prover generates a random isogeny $\psi : E_0 \to E_2$ of degree s, where $s \approx n$ is a smooth number coprime with both n and d, and generates the SIDH square (E_0, E_1, E_2, E_3) with edges $(\phi, \psi, \phi', \psi')$. To guarantee soundness, the prover needs to show that ψ and ψ' are parallel: the prover thus generates a s-basis R_2, S_2 on E_2, maps it to E_3 to obtain R_3, S_3, and expresses the kernels of $\hat{\psi}$ and $\hat{\psi}'$ in terms of R_2, S_2 and R_3, S_3 with the same linear coefficients. The prover also splits α in three shares $\alpha = \alpha_1\alpha_2\alpha_3$ and maps the points P_0, Q_0 through ψ and ϕ' with masking values α_1 and α_2 to obtain the points

$$P_2 = [\alpha_1]\psi(P_0),\ Q_2 = [\alpha_1]\psi(Q_0),$$
$$P_3 = [\alpha_2]\phi'(P_2),\ Q_3 = [\alpha_2]\phi'(Q_2),$$

which implies that P_3 and Q_3 also satisfy the relation

$$[\alpha_3]P_3 = \psi'(P_1),\ [\alpha_3]Q_3 = \psi'(Q_1).$$

Hence, the SIDH square commutes with respect to the points P_i, Q_i, i.e. if we restrict ourselves to the n-torsion, we have

$$[\alpha][s]\phi = [\alpha_3]\hat{\psi}' \circ [\alpha_2]\phi' \circ [\alpha_1]\psi.$$

Thus, the witness can be split into three components, and hence we obtain a proof with ternary challenges. The prover initially commits to the curves E_2, E_3 and the relevant points on them with a commitment scheme $\mathsf{C}(\cdot)$. Then, depending on the challenge, the prover responds with one edge of the SIDH square, the relevant curves and points, and the corresponding commitment openings. The proof is described in Fig. 5.4. Since each iteration has soundness error $2/3$, the proof must be repeated $-\lambda \log_{2/3}(2) \approx 1.71$ times to achieve a soundness error of $2^{-\lambda}$.

Remark 6. *If the kernel of the isogeny ϕ is not defined over a small extension field, as in the case of the message isogeny, the proof can be computed by gluing together multiple SIDH squares, as shown in [BCCDF+23].*

We now sketch the proofs of correctness, three-special soundness and zero-knowledge. Given the similarity of the zero-knowledge proof with those in [BKW20], the proofs also follow a similar approach.

- **Correctness.** A honest prover always generates proofs that are accepted by the verifier. The verifier recomputes the same operations as the prover and checks that the outputs match. The only difference is in the $\mathsf{chall} = \pm 1$ cases, where the verifier computes the dual of ψ and ψ', which then introduces the s factor in the point equality check.

- **Three-special soundness.** The protocol is three-special sound because there exists an extractor that extracts the witness given three accepting transcripts with the same commitments and different challenges. The isogeny ϕ can be computed by mapping the kernel of ϕ' (from $\mathsf{chall} = 0$) under the isogeny $\hat{\psi}$ (from $\mathsf{chall} = -1$). Since the isogenies ψ and ψ' are parallel (from all the challenges combined), this guarantees that ϕ is a d-isogeny from E_0 to E_1. The masking value α can be recomputed as the product of α_1, α_2, and α_3.

- **Zero-knowledge.** We sketch a simulator that given a statement $(E_0, P_0, Q_0, E_1, P_1, Q_1)$ and a challenge chall can simulate a valid transcript without knowledge of the witness. For the case $\mathsf{chall} = -1$, the simulator behaves like an honest prover. For $\mathsf{chall} = +1$, the situation is similar: the simulator can compute a d-isogeny ψ', pick a random basis R_3, S_3 of $E_3[d]$ and a random value $\alpha_3 \in \mathbb{Z}_n^*$, and compute the values a, b and points P_3, Q_3 that pass verification. Note that the points R_3, S_3 are uniformly random among the bases of $E_3[d]$, and the value α_3 is uniformly random and independent of α; the simulated values are thus distributed as the honestly-generated ones. The case of $\mathsf{chall} = 0$ is more complicated: the simulator

$\mathsf{P}_1((E_0, P_0, Q_0), (E_1, P_1, Q_1), \phi, \alpha)$:

Sample a random cyclic isogeny $\psi : E_0 \to E_2$ of degree s;
Construct the SIDH square $(E_0, E_1, E_2, E_3, \phi', \psi')$ on (ϕ, ψ);
Sample random units α_1, α_2 mod n and set $a_3 \coloneqq \alpha/\alpha_1\alpha_2$;
Set $P_2, Q_2 \coloneqq [\alpha_1]\psi(P_1), [\alpha_1]\psi(Q_1)$, and $P_3, Q_3 \coloneqq [\alpha_2]\phi'(P_2), [\alpha_2]\phi'(Q_2)$;
Let R_2, S_2 be a basis of $E_2[d]$ and set $R_3, S_3 \coloneqq \phi'(R_2), \phi'(R_3)$;
Write $K = [a]R_2 + [b]S_2$ for K a random generator of $\ker \hat{\psi}$
Sample random strings $r_1, \ldots, r_7$;
return $\big(\mathsf{st}, \mathsf{C}(E_2, R_2, S_2, P_2, Q_2; r_1)$, $\mathsf{C}(E_3, R_3, S_3, P_3, Q_3; r_2)$,
$\mathsf{C}(a, b; r_3)$, $\mathsf{C}(\phi'; r_4)$, $\mathsf{C}(\alpha_1; r_5)$, $\mathsf{C}(\alpha_2; r_6)$, $\mathsf{C}(\alpha_3; r_7)\big)$.

$\mathsf{P}_2(\mathsf{st}, \mathsf{chall})$:

if $\mathsf{chall} == -1$ **then**
    return $((E_2, R_2, S_2, P_2, Q_2, r_1), (a, b, r_3), (\alpha_1, r_5))$;
else if $\mathsf{chall} == 0$ **then**
    return $((E_2, R_2, S_2, P_2, Q_2, r_1), (E_3, R_3, S_3, P_3, Q_3, r_2), (\phi', r_4), (\alpha_2, r_6))$;
else if $\mathsf{chall} == 1$ **then**
    return $((E_3, R_3, S_3, P_3, Q_3, r_2), (a, b, r_3), (\alpha_3, r_7))$;

$\mathsf{V}((E_0, P_0, Q_0), (E_1, P_1, Q_1), (\mathsf{com}_1, \ldots, \mathsf{com}_9), \mathsf{chall}, \mathsf{resp})$:

if $\mathsf{chall} == -1$ **then**
    $((E_2, R_2, S_2, P_2, Q_2, r_1), (a, b, r_3), (\alpha_1, r_5)) = \mathsf{resp}$;
    Check $\mathsf{com}_1 = \mathsf{C}(E_2, R_2, S_2, P_2, Q_2; r_1)$,
        $\mathsf{com}_3 = \mathsf{C}(a, b; r_3)$, $\mathsf{com}_5 = \mathsf{C}(\alpha_1; r_5)$;
    Let $\hat{\psi}$ be the isogeny with kernel $\langle [a]R_2 + [b]S_2 \rangle$;
    Check $\hat{\psi}$ is an s-isogeny from E_2 to E_0;
    Check $[\alpha_1 s]P_0 = \hat{\psi}(P_2)$ and $[\alpha_1 s]Q_0 = \hat{\psi}(Q_2)$;
else if $\mathsf{chall} == 0$ **then**
    $((E_2, R_2, S_2, P_2, Q_2, r_1), (E_3, R_3, S_3, P_3, Q_3, r_2), (\phi', r_4), (\alpha_2, r_6)) = \mathsf{resp}$;
    Check $\mathsf{com}_1 = \mathsf{C}(E_2, R_2, S_2, P_2, Q_2; r_1)$,
        $\mathsf{com}_2 = \mathsf{C}(E_3, R_3, S_3, P_3, Q_3; r_2)$,
        $\mathsf{com}_4 = \mathsf{C}(\phi'; r_4)$, $\mathsf{com}_6 = \mathsf{C}(\alpha_2; r_6)$;
    Check ϕ' is a d-isogeny from E_1 to E_2;
    Check $R_3, S_3 = \phi'(R_2), \phi'(R_3)$;
    Check $P_3, Q_3 = [\alpha_2]\phi'(P_2), [\alpha_2]\phi'(Q_2)$;
else if $\mathsf{chall} == 1$ **then**
    $((E_3, R_3, S_3, P_3, Q_3, r_2), (a, b, r_3), (\alpha_3, r_7)) = \mathsf{resp}$;
    Check $\mathsf{com}_2 = \mathsf{C}(E_3, R_3, S_3, P_3, Q_3; r_2)$,
        $\mathsf{com}_3 = \mathsf{C}(a, b; r_3)$, $\mathsf{com}_7 = \mathsf{C}(\alpha_3; r_7)$;
    Check $\langle R_3, S_3 \rangle = E_3[s]$;
    Let $\hat{\psi}'$ be the isogeny with kernel $\langle [a]R_3 + [b]S_3 \rangle$;
    Check $\hat{\psi}$ is an s-isogeny from E_3 to E_1;
    Check $[\alpha_3 s]P_1 = \hat{\psi}'(P_3)$ and $[\alpha_3 s]Q_1 = \hat{\psi}'(Q_3)$;

Figure 5.4: Interactive proof of knowledge for the relation $\mathcal{R}_{\mathsf{iso}}$.

can sample a random curve E_2, generate a random basis P_2, Q_2 of $E_2[n]$ that satisfies $e(P_2, Q_2) = e(P_0, Q_0)^{x^2 s}$ for some random x, pick a random d-isogeny $\phi' : E_2 \to E_3$, and compute the image points on E_3. In this case, the indistinguishability of the simulator's output is only computational. It is thus based on the conjectured hardness of the following problem, which is a modified version of the Decisional Supersingular Product (DSSP) problem introduced in [JD11].

Problem 6 (DSSP with Torsion (DSSPwT) problem)**.** *Given an isogeny $\phi : E_0 \to E_1$ and points $P_0, Q_0 \in E_0[n]$, where $n = f_{MSIDH}(\lambda, d)$, distinguish between the following distributions:*

- $\mathcal{D}_0 = \{(E_2, P_2, Q_2, E_3, \phi')\}$*, where E_2 is the codomain of an s-isogeny $\psi : E_0 \to E_2$, the points P_2, Q_2 satisfy $P_2 = [\alpha]\psi(P_0)$, $Q_2 = [\alpha]\psi(Q_0)$ for some $\alpha \in \mathbb{Z}_n^*$, and $\phi' : E_2 \to E_3$ is a d-isogeny with kernel $\ker \phi' = \psi(\ker \phi)$.*
- $\mathcal{D}_1 = \{(E_2, P_2, Q_2, E_3, \phi')\}$*, where E_2 is a random supersingular curve with the same cardinality as E_0, P_2 and Q_2 are two random points of order n such that $e(P_2, Q_2) = e(P_0, Q_0)^s$, and the isogeny ϕ' is a d-isogeny between E_2 and E_3.*

Note that [BKW20] argues that a similar proof can only reveal one torsion point (either P_i or Q_i) at a time to prevent a distinguishing attack on the simulator. The attack they present relies on computing the Weil pairing between two points of coprime order, and thus their pairing is always one. The attack thus does not apply, and the simulated transcript remains undistinguishable under Weil pairing checks because the sampled points P_2, Q_2 are guaranteed to have the same pairing as the honestly-generated points. By revealing both points P_i and Q_i we obtain a significantly more efficient proof, since it has $1/3$ soundness rather than $1/6$.

Optimizations. For simplicity, the proof in Fig. 5.4 contains a schematic description of the protocol, but the proof can be made more efficient through a series of optimizations.

In the commitment phase, the value α_2 is only revealed together with the isogeny ϕ', and thus they can be committed together. Note that we have the prover commit to ϕ' to make the proof online-extractable without recursion, which is necessary to achieve a proof in the UC model. For applications of this proof outside of the OPRF context, the prover can avoid committing to ϕ'. The masking values α_1 and α_3 are independent of α, even when considered together, because α_2 is uniformly random. They can then be committed together and revealed both in the response to challenges $\mathsf{chall} = \pm 1$. Since the commitment for a, b is also revealed when $\mathsf{chall} = \pm 1$, the values a, b, α_1, α_3 can all be committed together. When $\mathsf{chall} = -1$, the curve E_3 and the points P_3, Q_3 are not revealed, and thus learning α_3 does not provide any information. The same applies to α_1 when $\mathsf{chall} = +1$. This allow us to reduce the number of commitments to four.

To further reduce the communication between prover and verifier, the basis R_2, S_2 on E_2 can be chosen canonically, so that it can be recomputed from E_2. Moreover, for the challenge $\mathsf{chall} = -1$, the prover can avoid revealing the curve E_2, the points P_2, Q_2 and the coefficients a, b by revealing instead a kernel generator of ψ. The prover can recompute E_2, P_2, Q_2 and obtain a, b by writing a kernel generator of $\hat{\psi}$ in terms of the canonical basis R_2, S_2. Normally, the recomputed a, b would not be the same as those computed by the verifier since they are not unique. The problem can be avoided by fixing a canonical way to compute the coefficients, such as prescribing that one of the two coefficients must be one, and that a must be one if both coefficients are invertible mod s. The same approach holds for $\mathsf{chall} = +1$, except that the points R_3, S_3 have to be revealed by the prover. In the case of the horizontal isogeny, the prover can avoid revealing E_3 and the points R_3, S_3 and P_3, Q_3. They can all be recomputed from the remaining values.

Concrete cost. Each repetition of the proof requires two commitments, which are 2λ-bit long and use a λ-bit long opener. When $\mathsf{chall} = -1$, the prover reveals one s-isogeny, a masking value, and two commitment openers, which requires $\log n + \log s + 2\lambda$ bits. When $\mathsf{chall} = +1$, the prover also reveals two torsion points of order s: if they are compressed

as in [AJKKL16], the response requires $5\log s + \log n + 2\lambda$ bits. Lastly, for $\mathsf{chall} = 0$, the prover reveals a curve, a d-isogeny, two points of order n, a masking value, and three openers; thus, the answer requires $2\log p + \log d + 5\log n + 3\lambda$ bits.

Hence, if we assume $d \approx n \approx s \approx \sqrt[3]{n}$, an average proof where the three challenges appear equally requires $\approx 1.71\lambda(20/9\log p + 7/3\lambda)$ bits, while a worst-case proof, with only $\mathsf{chall} = 0$ challenges, requires $\approx 1.71\lambda(4\log p + 3\lambda)$ bits.

5.6 — Verifiability

Oblivious PRFs can satisfy a stronger security property called *verifiability*. Informally, this guarantees that the server behaves honestly and always uses the same long-term static key. This is needed to guarantee the privacy of the user in those instances where the user may later reveal the output of the OPRF. A malicious server may behave "honestly" while also using different secret keys on different interactions. After learning the OPRF output of the user, the server can then test which secret key was used to produce that specific output and thus link the user to a specific user-server interaction.

The Boneh et al. construction. The OPRF protocol by Boneh et al. achieves verifiability by introducing three components. First, the server initially commits to a secret key k. The commitment is in the form of an elliptic curve $E_C \coloneqq E/\langle P + [k]Q\rangle$, where the curve E and the points P, Q are fixed parameters. Second, during the OPRF execution, the server provides a zero-knowledge proof that its computations used the same key as the one in the commitment. We refer to this proof as a *proof of parallel isogeny* (PoPI). Lastly, the server also provides two *proofs of isogeny knowledge* (PoIKs) that guarantee the correctness of the computations during both the commitment stage and the OPRF execution.

The proof of parallel isogeny proposed by Boneh et al. relies on the user and the server engaging in an SIDH exchange, where one of the sides is either the commitment isogeny or the the secret server isogeny in the OPRF protocol. The user only reveals the codomain of an isogeny starting from either starting curve, which means the server cannot distinguish

the two cases. Thus, if the keys used in the protocol and the commitment were different, the server could not do better than randomly guessing which starting curve the user used. By repeating the protocol λ times, the server can prove the parallelness of the isogenies with soundness $2^{-\lambda}$. However, this proof is inherently interactive. Since the server also needs to defend against adaptive attacks [GPST16], the proof uses an approach similar to the Fujisaki-Okamato transform, which requires five rounds of interaction. Moreover, the proof relies on multiple SIDH exchanges, and it is thus broken by the attacks on SIDH [CD23; MMPPW23; Rob23]. It may be possible to avoid some of the issues, for instance by starting from a curve of unknown endomorphism ring using a trusted setup and by switching to an SIDH version that is resistant to the recent attacks. However, it seems impossible to obtain a non-interactive proof using a similar approach.

Our proposal. We introduce a novel public-coin proof protocol of parallel isogeny that sidesteps the problems discussed above. Since the proof does not rely on private randomness, we obtain a proof *of knowledge* that can be made non-interactive via the Fiat-Shamir transform [FS87] or the Unruh transform [Unr15]. In the OPRF setting, we will rely on the latter to achieve the online-extractability without rewinding needed to get a proof in the UC model. Our main approach relies on executing two proofs of isogeny knowledge in parallel *with correlated randomness*. Since part of the randomness used is shared, we can obtain a proof of parallelness without needing additional computations.

Firstly, we formalize the notion of parallelness. We say that two d-isogenies $\phi : E_0 \to E_1$ and $\tilde{\phi} : \tilde{E}_0 \to \tilde{E}_1$ are parallel with respect to the bases $R, S \in E_0[d]$ and $\tilde{R}, \tilde{S} \in E_0'[d]$ if there exists coefficients $a, b \in \mathbb{Z}_d$ such that $\ker \phi = \langle [a]R + [b]S \rangle$ and $\ker \tilde{\phi} = \langle [a]\tilde{R} + [b]\tilde{S} \rangle$. This suggests that the parallelness relation that we are proving is the following:

$$\mathcal{R}_{\mathsf{par}} = \left\{ ((E_0, R, S, E_1, \tilde{E}_0, \tilde{R}, \tilde{S}, \tilde{E}_1), k_0, k_1) \;\middle|\; \begin{array}{l} E_0/\langle [k_0]R + [k_1]S \rangle \cong E_1, \\ \tilde{E}_0/\langle [k_0]\tilde{R} + [k_1]\tilde{S} \rangle \cong \tilde{E}_1 \end{array} \right\}.$$

However, as discussed before, we are combining several proofs together to obtain a

larger proof that simultaneously proves knowledge of two isogenies and guarantees the two isogenies are parallel. We thus obtain a proof for the following relation, where we consider the case of a secret key with two coefficients for completeness. For practical reasons, the OPRF will fix $k_0 = 1$ without any loss of security.

$$\mathcal{R}^*_{\mathsf{par}} = \left\{ \begin{array}{l} ((E_0, R, S, P_0, Q_0, E_1, P_1, Q_1, \\ \quad \tilde{E}_0, \tilde{R}, \tilde{S}, \tilde{P}_0, \tilde{Q}_0, \tilde{E}_1, \tilde{P}_1, \tilde{Q}_1), \\ \quad (k_0, k_1, \alpha, \alpha')) \end{array} \middle| \begin{array}{c} \ker \phi = \langle [k_0]R + [k_1]S \rangle, \\ \ker \phi' = \langle [k_0]\tilde{R} + [k_1]\tilde{S} \rangle, \\ (E_0, P_0, Q_0, E_1, P_1, Q_1), (\phi, \alpha) \in \mathcal{R}_{\mathsf{iso}}, \\ (\tilde{E}_0, \tilde{P}_0, \tilde{Q}_0, \tilde{E}_1, \tilde{P}_1, \tilde{Q}_1), (\phi', \alpha') \in \mathcal{R}_{\mathsf{iso}} \end{array} \right\}.$$

Now, let the curve $\tilde{E}_0$ with a d-basis $\tilde{R}, \tilde{S}$ be fixed protocol parameters. Using the same notation as before, assume that server has committed to its key (k_0, k_1) by publishing the codomain of the d-isogeny $\tilde{\phi}$ that has kernel $\langle [k_0]\tilde{R} + [k_1]\tilde{S} \rangle$. The server may also reveal some torsion information in its commitment, but as we will discuss later, this is not strictly needed. During the OPRF execution, the server receives a curve E_0 with a d-basis R, S on it, and it computes $\phi : E_0 \to E_1 \coloneqq E_0/\langle [k_0]R + [k_1]S \rangle$. The server then wants to prove that it knows the isogenies ϕ and $\tilde{\phi}$ and that they are parallel.

If the server simply ran two instances of the PoIK from Fig. 5.4 in parallel, there would be no way to convince the prover that the isogenies are indeed parallel. If the proofs share the same challenges, i.e. the verifier sends the same challenges to both proofs, the server would respond with both ϕ and $\tilde{\phi}'$ when $\mathsf{chall} = 0$. However, the isogenies ϕ and $\tilde{\phi}'$ are not parallel with respect to the bases R_2, S_2 and $\tilde{R}_2, \tilde{S}_2$ since they are randomly generated. We thus to want to modify the proof such that the bases R_2, S_2 and $\tilde{R}_2, \tilde{S}_2$ are related to R_0, S_0 and $\tilde{R}_0, \tilde{S}_0$, so that when ϕ and $\tilde{\phi}$ are parallel, so are ϕ' and $\tilde{\phi}'$. One way to do this is by computing the basis R_2, S_2 as $R_2, S_2 = \psi(R_0), \psi(S_0)$ (and similarly for $\tilde{R}_2, \tilde{S}_2$) in both proofs, where ψ is the vertical isogeny used in the proof of knowledge. This is however not zero-knowledge, because when $\mathsf{chall} = 0$, the verifier could recompute the secret isogeny ϕ. Instead, we propose that the prover generates four random coefficients

$w, x, y, z \in \mathbb{Z}_d$ such that $wz - xy \neq 0 \bmod d$, and computes R_2 and S_2 as the solution of

$$R_0 = [w]\psi(R_2) + [x]\psi(S_2), \quad S_0 = [y]\psi(R_2) + [z]\psi(S_2).$$

This is then secure, because the basis R_2, S_2 is uniformly random. Thus, for a single proof, this change only affects how the random points R_2, S_2 are generated, but does not affect the security of the proof. The rest of the proof needs to be modified to ensure that the process is followed correctly, i.e. we want the prover to reveal the values w, x, y, z together with ψ so that the verifier can verify the correctness of R_2 and S_2. The modified proof is denoted by $\mathcal{P}^*_{\mathsf{iso}}$, and it is represented explicitly in Fig. 5.6 in Section 5.A.

Now, if the prover executes the modified proof of isogeny knowledge for ϕ and $\tilde{\phi}$ in parallel, with the same challenges, and with the same values x, w, y, z, the isogenies $\phi', \tilde{\phi}'$ revealed when $\mathsf{chall} = 0$ are parallel when the isogenies $\phi, \tilde{\phi}$ are also parallel, as shown in the following lemma.

Lemma 17. *Let notation be as above. The isogenies $\phi, \tilde{\phi}$ are parallel if and only if the isogenies $\phi', \tilde{\phi}'$ are also parallel.*

Proof. Assume the isogeny ϕ has kernel $\langle [k_0]R_0 + [k_1]S_0 \rangle$ and the isogeny $\tilde{\phi}$ has kernel $\langle [\tilde{k}_0]\tilde{R}_0 + [\tilde{k}_1]\tilde{S}_0 \rangle$. The kernel of ϕ' is the image of the kernel of ϕ under ψ, i.e. $\ker \phi' = \psi(\ker \phi)$. Since $\ker \phi = \langle [k_0]R_0 + [k_1]S_0 \rangle$, it follows that

$$\ker \phi' = \langle [k_0]\psi(R_0) + [k_1]\psi(S_0) \rangle = \langle [wk_0 + yk_1]R_2 + [xk_0 + zk_1]S_2 \rangle.$$

Similarly, we obtain

$$\ker \tilde{\phi}' = \langle [w\tilde{k}_0 + y\tilde{k}_1]\tilde{R}_2 + [x\tilde{k}_0 + z\tilde{k}_1]\tilde{S}_2 \rangle.$$

Since the coefficients w, x, y, z were chosen such that the matrix $\left(\begin{smallmatrix} w & x \\ y & z \end{smallmatrix}\right)$ is invertible, we

obtain that

$$(k_0 = \tilde{k}_0) \wedge (k_1 = \tilde{k}_1) \iff (wk_0 + yk_1 = w\tilde{k}_0 + y\tilde{k}_1) \wedge (xk_0 + zk_1 = x\tilde{k}_0 + z\tilde{k}_1).$$

□

We can now use the proof $\mathcal{P}^*_{\mathsf{iso}}$ from Fig. 5.6 to construct our proof of parallel isogeny knowledge. The prover runs two such proofs in parallel, with the same randomness (w, x, y, z), and responds to the verifier's challenges with the responses of the individual proofs. The resulting proof is represented explicitly in Fig. 5.6 in Section 5.A. The security proofs follow closely those of the PoIK $\mathcal{P}_{\mathsf{iso}}$ in Section 5.5.2: correctness of $\mathcal{P}_{\mathsf{iso}}$ implies correctness of $\mathcal{P}_{\mathsf{par}}$, while the soundness of $\mathcal{P}_{\mathsf{par}}$ follows from the soundness of $\mathcal{P}_{\mathsf{iso}}$ and Lemma 17. The argument for zero-knowledge is also similar, but it is based on the hardness of the following problem, which takes into consideration that the two parallel instance partially share the same randomness.

Problem 7 (Double DSSP with Torsion (DDSSPwT) problem)**.** *Let $\mathcal{D}_0$ and $\mathcal{D}_1$ be as in Problem 6. Given:*

1. *two d-isogenies $\phi : E_0 \to E_1$, $\tilde{\phi} : \tilde{E}_0 \to \tilde{E}_1$,*
2. *the points $R_0, S_0 \in E_0[d]$ and $\tilde{R}_0, \tilde{S}_0 \in \tilde{E}_0[d]$,*
3. *the points $P_0, Q_0 \in E_0[n]$ and $\tilde{P}_0, \tilde{Q}_0 \in \tilde{E}_0[n]$, where $n = f_{\mathsf{MSIDH}}(\lambda, d)$,*

distinguish between the following distributions:

- $\mathcal{D}^*_0 = \left\{ \begin{array}{l} (E_2, R_2, S_2, P_2, Q_2, E_3, \phi'), \\ (\tilde{E}_2, \tilde{R}_2, \tilde{S}_2, \tilde{P}_2, \tilde{Q}_2, \tilde{E}_3, \tilde{\phi}') \end{array} \right\}$*, where the curves and the n-torsion points follow the $\mathcal{D}_0$-distribution, i.e. we have $(E_2, P_2, Q_2, E_3, \phi') \leftarrow \mathcal{D}_0$, and $(\tilde{E}_2, \tilde{P}_2, \tilde{Q}_2, \tilde{E}_3, \tilde{\phi}') \leftarrow \mathcal{D}_0$, and moreover*

$$\begin{bmatrix} R_2 \\ S_2 \end{bmatrix} = B \begin{bmatrix} \psi(R_0) \\ \psi(S_0) \end{bmatrix}, \quad \text{and} \quad \begin{bmatrix} \tilde{R}_2 \\ \tilde{S}_2 \end{bmatrix} = B \begin{bmatrix} \tilde{\psi}(\tilde{R}_0) \\ \tilde{\psi}(\tilde{S}_0) \end{bmatrix},$$

for some $B \in \mathrm{GL}_2(\mathbb{Z}_n)$, and ψ and $\tilde{\psi}$ being respectively the s-isogenies between E_0 and E_2 and $\tilde{E}_0$ and $\tilde{E}_2$ that are guaranteed to exist because of the $\mathcal{D}_0$ distribution;

- $\mathcal{D}_1^* = \left\{ \begin{array}{l} (E_2, R_2, S_2, P_2, Q_2, E_3, \phi'), \\ (\tilde{E}_2, \tilde{R}_2, \tilde{S}_2, \tilde{P}_2, \tilde{Q}_2, \tilde{E}_3, \tilde{\phi}') \end{array} \right\}$, *where the curves and the n-torsion points follow the $\mathcal{D}_1$-distribution, i.e. we have $(E_2, P_2, Q_2, E_3, \phi') \leftarrow \mathcal{D}_1$, and $(\tilde{E}_2, \tilde{P}_2, \tilde{Q}_2, \tilde{E}_3, \tilde{\phi}') \leftarrow \mathcal{D}_1$, and moreover the points R_2, S_2 and $\tilde{R}_2, \tilde{S}_2$ form a random basis of $E_2[d]$ and $\tilde{E}_2[d]$, respectively.*

The proof $\mathcal{P}_{\mathsf{par}}$ is a proof of knowledge, and it can be made non-interactive with standards transformations, such as the Fiat-Shamir [FS87] or the Unruh [Unr15] transform. This is the first non-interactive proof of parallelness.

Optimizations. For simplicity, the presentation of the proof $\mathcal{R}^*_{\mathsf{par}}$ preferred a schematic description, but it is possible to improve the protocol to make it more compact. Besides the optimizations applicable to the proof $\mathcal{P}_{\mathsf{iso}}$ described in Section 5.5.2, we remark that parallelness is independent of torsion images. Thus, the proofs of isogeny knowledge do not need to guarantee the correctness of torsion images to prove parallelness. However, in the OPRF context, the correctness of the torsion images revealed by the server is needed to guarantee verifiability: a malicious server might otherwise reveal incorrect torsion points to different users and use that information to match OPRF outputs to specific interactions. Hence, the proof can be made more efficient by avoiding proving the correctness of torsion images for the commitment isogeny.

Concrete cost. The proof described in Fig. 5.6 adds the communication of the values w, x, y, z when $\mathsf{chall} = -1$. In that case, the prover's response requires $\log n + \log s + 4 \log d + 2\lambda$ bits; when $\mathsf{chall} = 0$, the response is also larger because the points R_2, S_2 need to be communicated explicitly. The answers to the other challenge remains unchanged.

The same proof, when used for the commitment isogeny, can avoid proving correctness of the torsion images, resulting in a smaller proof. In particular, no masking values

are ever revealed, and when $\mathsf{chall} = 0$ the response does not contain the points P_2, Q_2 on E_2. Setting $d \approx n \approx s \approx \sqrt[3]{n}$, we obtain that an average proof $\mathcal{P}_{\mathsf{par}}$ requires $\approx 1.71\lambda(49/9 \log p + 14/3\lambda)$ bits, while a worst-case proof would require $\approx 1.71\lambda(9 \log p + 6\lambda)$ bits.

5.7 — A new OPRF protocol

In this section, we combine the countermeasures presented in Section 5.4, the SIDH countermeasures and the novel proof of isogeny knowledge discussed in Section 5.5, and the non-interactive proof of parallel isogeny introduced in Section 5.6 to obtain a verifiable OPRF protocol that is post-quantum secure, round-optimal, and moderately compact.

The OPRF protocol is a two-party protocol between a user U and a server S. Let N_{M}, N_{B}, N_{K} be coprime numbers representing the degrees of the message isogeny, the blinding isogeny, and the server's isogeny, respectively. Let p be a prime of the form $p = N_{\mathsf{M}} N_{\mathsf{B}} N_{\mathsf{K}} f - 1$, for some cofactor f, and let $E_0, \tilde{E}$ be two supersingular elliptic curves defined over $\mathbb{F}_{p^2}$. Moreover, let P, Q be a fixed basis of $E_0[N_{\mathsf{M}}]$ and let $\tilde{P}, \tilde{Q}$ be a fixed basis of $\tilde{E}[N_{\mathsf{K}}]$. The first curve is used to compute the PRF, while the second is used within the server's commitment.

At a high-level, to evaluate the OPRF on an input x, the user maps the input to a curve E_m according to Algorithm 3 and computes a blinding isogeny $\phi_b : E_m \to E_{mb}$. The user then sends the codomain curve, together with torsion images and a proof of their correctness, to the server, which computes a second isogeny $\phi_k : E_{mb} \to E_{mbk}$. The torsion information is appropriately masked to avoid the SIDH attacks. The server then responds with the curve E_{mbk}, some torsion information, a proof of their correctness, and a proof that it has used the previously-committed secret key. The user then concludes by using the torsion information provided by the server to undo the blinding isogeny and compute the curve E_{mk}. Its j-invariant is then hashed together with the input and the

server's public key to form the PRF output. The protocol is described in Fig. 5.5 and it realized the OPRF ideal functionality of Fig. 5.2, which allows us to state the following theorem.

Theorem 4. *The protocol described in Fig. 5.5 realizes the ideal functionality $\mathcal{F}_{vOPRF}$ of Fig. 5.2 in the random oracle model.*

The proof follows the same line as the security proof of the OPRF protocol by Boneh et al. [BKW20, Theorem 20], since the hardness assumption of Problem 4 and the proof $\mathcal{P}_{\mathsf{iso}}$ are a drop-in replacement for the Auxiliary One-More SIDH assumption and the NIZKPK proof used in [BKW20], respectively. At a high level, the case of an honest user and malicious server in the proof is simple because the server only interacts with the user through their first query, and in that case the user's security corresponds to the input hiding property, guaranteed by the hardness of Problem 1. The case of a malicious user is more complicated, because the user has output. The server can be simulated as a honest server, but to ensure that the malicious user output is indistinguishable from the ideal-world, the random oracle $\bar{H}$ can be programmed to output the ideal-world output. This would create a problem with the ticketing system of the ideal functionality if the adversary could produce more OPRF outputs than the number of interactions, but the one-more unpredictability property prevents that. The main difference between this proof and that of [BKW20] is the use of a non-interactive proof of parallel isogeny that can be simulated in the proof, which results in a simpler proof since the proof of knowledge can be simulated. Note that the proof in [BKW20] is written in terms of the augmentable commitment abstraction, which we preferred avoiding; since the same security properties can be directly expressed in terms of the OPRF protocol, as shown in Section 5.3, the difference is purely syntactical.

Parameter selection. Firstly, we discuss how to select the starting curves E_0 and $\tilde{E}$. As mentioned in Section 5.5, the cryptanalysis on masked-torsion SIDH with a starting

[3]The proof algorithm does not receive torsion points because, as discussed in Section 5.6, they are not necessary to prove parallelness.

Parameters. A prime p of the form $p = N_{\mathsf{M}} N_{\mathsf{B}} N_{\mathsf{K}} f - 1$, where $N_{\mathsf{M}}, N_{\mathsf{B}}, N_{\mathsf{K}}$ are smooth coprime integers and f a smooth cofactor. E_0 and $\tilde{E}$ are supersingular elliptic curves defined over $\mathbb{F}_{p^2}$, where $\mathrm{End}\tilde{E}$ is unknown, and $P, Q \in E_0[N_{\mathsf{M}}]$ and $\tilde{P}, \tilde{Q} \in E[N_{\mathsf{K}}]$ are fixed bases.
The protocol also relies on several functions:

- $H_i : \{0,1\}^* \to \mathbb{Z}_M$ for $i \in \{1, \ldots, I\}$, where I is such that $N_{\mathsf{M}}^I > 2^{4\lambda}$, to use within $\mathcal{H}_I$,
- $\bar{H} : \{0,1\}^* \to \{0,1\}^\lambda$, to hash the final PRF output,

and two non-interactive proofs of knowledge:

- $\mathcal{P}_{\mathsf{iso}}$, for the user to prove correctness of torsion images,
- $\mathcal{P}_{\mathsf{par}}$, for the server to prove it computed honestly with the committed key.

Initialization. On input INIT from the environment, the server S:

- sample $k \leftarrow \mathbb{Z}_K$ and stores it,
- computes the curve $\tilde{E}_C = \tilde{E}/\langle \tilde{P} + [k]\tilde{Q} \rangle$,
- stores $\mathsf{pk} = (j(E_C))$ and outputs $(\text{INIT}, \mathsf{pk})$.

Evaluation. On input INIT from the environment, the server S:

- On input (EVAL, S, x), the user U proceeds as follows:
  1. Sample $\alpha \leftarrow \mathbb{Z}_N^*$ and $b \leftarrow \mathbb{Z}_B$,
  2. Compute $(\phi_m, E_m) = \mathcal{H}_I(x)$;
  3. Compute $\phi_b : E_m \to E_{mb} \coloneqq E_m/\langle P_m + [b]Q_m \rangle$, where $P_m, Q_m = \mathcal{B}_B(E_m)$,
  4. Set $\phi_{mb} = \phi_b \circ \phi_1 \circ \phi_0$, $R = [\alpha]\phi_{mb}(P)$, $S = [\alpha]\phi_{mb}(Q)$,
  5. Compute $\pi_c \leftarrow \mathcal{P}_{\mathsf{iso}}(E_0, P, Q, E_{mb}, R, S, \phi_{mb}, \alpha)$,
  6. Send message (E_{mb}, R, S, π_c) to the server and store ϕ_b
- On input SERVERCOMPLETE from the environment and message (E_{mb}, R, S, π_c) from the user U, the server S proceeds as follows:
  1. Verify the proof π_c,
  2. Sample $\alpha_k \leftarrow \mathbb{Z}_n^*$,
  3. Compute $\phi_k : E_{mb} \to E_{mbk} \coloneqq E_{mb}/\langle R + [k]S \rangle$,
  4. Compute $R_k = [\alpha_k]\phi_k(P_b)$, $S_k = [\alpha_k]\phi_k(Q_b)$, where $P_b, Q_b = \mathcal{B}_B(E_{mb})$,
  5. Compute $\pi_k \leftarrow \mathcal{P}_{\mathsf{par}}((E_{mb}, P_b, Q_b, E_{mbk}, R_k, S_k), (\tilde{E}, \tilde{P}, \tilde{Q}, \tilde{E}_C), k, \alpha_k)$[3],
  6. Send $(\mathsf{pk}, E_{mbk}, R_k, S_k, \pi_k)$ to the user U
- On input $(\mathsf{pk} = j(E_c), E_{mbk}, R_k, S_k, \pi_k)$ from the server S, the user U proceeds as follows:
  1. Verify the proof π_k,
  2. Compute b_0, b_1 such that $\langle [b_0]P_b + [b_1]Q_b \rangle = \ker \hat{\phi}_b$, where $P_b, Q_b = \mathcal{B}_d(E_{mb})$,
  3. Compute $\phi_u : E_{mbk} \to E_{mk} \coloneqq E_{mbk}/\langle [b_0]R_k + [b_1]S_k \rangle$,
  4. Compute $y = \bar{H}(x, \mathsf{pk}, j(E_m k))$ and output $(\text{EVAL}, \mathsf{pk}, y)$.

Figure 5.5: The verifiable OPRF protocol.

curve with small endomorphism [FMP23a, Section 4.2] does not apply here, since the message isogeny removes this property from the starting curve of the blinding isogeny.

Hence, the curve E_0 does not need to have unknown endomorphism ring. However, the situation is different for $\tilde{E}$: as observed in [BKMPS21], knowledge of $\mathrm{End}\tilde{E}$ allows to find collisions in the server's commitment. Thus, knowing $\mathrm{End}\tilde{E}$ would allow the server to break verifiability, since it could prove parallelness to two distinct isogenies. It is thus necessary that the curve $\tilde{E}$ is generated by a trusted party or through a multiparty trusted setup ceremony, such as the one presented in [BCCDF+23].

The main parameter of the OPRF protocol is the prime p. Firstly, if the message isogeny is the composition of many isogenies whose kernel is defined over $\mathbb{F}_{p^4}$, the value $p+1$ does not need have a dedicated factor. Then, for the main exchange, i.e. the blinding, server's isogeny, unblinding part, we need to smooth coprime integers N_B and N_K that are highly composite to prevent the SIDH attacks. Following the analysis of Section 5.5, we have $N_\mathsf{B} \approx N_\mathsf{K} \approx 2^{2956}$. Lastly, the proofs of knowledge $\mathcal{P}_{\mathsf{iso}}$ and $\mathcal{P}_{\mathsf{par}}$ require a third cofactor N_S that is coprime with both N_B and N_K. To guarantee the hardness of Problems 6 and 7, the integer N_S needs to be of the same length as N_B and N_K. However, since torsion points of order N_S do not need to be masked, the value N_S can be a prime power. Putting this together, we obtain that the prime p needs to be of the form $p = N_\mathsf{B}N_\mathsf{S}N_\mathsf{K}f - 1$ and be at least 8868-bit long to guarantee $\lambda = 128$ bits of security. Note that the new computation of the message isogeny and the new proofs of knowledge has significantly reduced the size of the prime; compared to the OPRF protocol by Boneh et al., we use a prime that is $5.8\times$ larger, while relying on an SIDH protocol with isogenies that are $9.2\times$ longer.

Efficiency. We can now estimate the communication cost of the OPRF protocol. The two main contributors are the non-interactive proofs of knowledge. If we set $N_\mathsf{B} \approx N_\mathsf{S} \approx N_\mathsf{K} \approx 2^{2956}$, and we consider that the user proves knowledge of the isogeny $\phi_b \circ \phi_m$, which has degree $\approx N_\mathsf{S}2^{4\lambda}$, we obtain that the user's proof takes 550 kB on average, while a worst-case proof requires 908 kB. For the proof of parallelness, we obtain that an average proof requires 1.3 MB on average, while a worst-case proof requires 2.2 MB. Putting altogether, a single execution of the verifiable OPRF requires 1.9 MB, but it can take up

Table 5.1: Comparison of existing post-quantum OPRF protocols.

Protocol	Rounds	Bandwidth (avg.)	Verifiable	Secure
[ADDS21] (LWE)	2	>128 GB	✓	✓
[BKW20] (SIDH)	5	1.4 MB	✓	✗
[BKW20] (CSIDH)	3	424 kB	✗	✓
[This work]	2	1.9 MB	✓	✓

3.2 MB, depending on the distribution of challenges within the proofs of knowledge.

If we compare our results with the protocol by Boneh et al., we see that our protocol is only moderately larger despite including countermeasures against two types of attacks and reducing the round complexity. Boneh et al. reports only the communication cost of worst-case proofs, but we recompute that an average user-side proof requires 441 kB, while the server-side proofs require 979 kB. Altogether, our protocol requires 30% more communication. This is due to the use of significantly longer isogenies ($\approx 2^{2956}$ vs $\approx 2^{320}$) to protect against the SIDH attacks; any future improvement in the SIDH countermeasures might significantly reduce the communication cost of the OPRF protocol.

We summarize the state of post-quantum OPRF protocols in Table 5.1. When compared to the CSIDH-based OPRF by Boneh et al., our OPRF offers verifiability and a lower number of communication rounds. While the communication cost is higher, if we remove the large server-side proof needed for verifiability, our protocol requires less than a 30% increase in communication.

5.8 — Conclusion

In this work, we presented a post-quantum verifiable oblivious PRF protocol that is moderately compact and round-optimal. The protocol is the first round-optimal OPRF based on isogenies, and it is several orders of magnitude more compact than the existing round-optimal protocol. To obtain this protocol, we started from an insecure protocol by Boneh et al., and we proposed an efficient countermeasure against the one-more unpredictability attack, integrated the existing SIDH countermeasures, developed a

new zero-knowledge proof of isogeny that works with the SIDH countermeasures, and proposed a novel non-interactive proof of parallel isogeny that reduced the number of rounds to two.

The protocol is an important stepping stone towards fully practical post-quantum OPRFs, but its performance is hindered by the inefficiency of the SIDH countermeasures. In future work, we aim at developing more efficient solutions: a moderate reduction in the degree of the isogenies would significantly improve the efficiency of the protocol. It is also interesting to improve the proof of parallel isogeny by avoiding validating the commitment isogeny at every interaction.

Acknowledgements. The author would also like to thank Christophe Petit and Luca de Feo for various suggestions, and Tako Boris Fouotsa, Christophe Petit, and the anonymous reviewers of the workshop PQCifris 2022 for feedback on earlier drafts of this work. The author thanks Luca de Feo, Antonin Leroux, and Benjamin Wesolowski for fruitful discussions on isogeny-based zero-knowledge proofs at the Banff International Research Station workshop “Supersingular Isogeny Graphs in Cryptography”.

5.A — Additional material from Section 5.6

$\mathsf{P}_1^*\Big(\big((E_0, R_0, S_0, P_0, Q_0), (E_1, P_1, Q_1), \phi, \alpha\big), (w, x, y, z)\Big)$:

1-4: Same as P_1 in Fig. 5.4.
5: Set $R_2 := [w]\psi(R_0) + [x]\psi(S_0), S_2 := [y]\psi(R_0) + [z]\psi(S_0)$;
6-8: Same as P_1 in Fig. 5.4.

$\mathsf{P}_2^*(\mathsf{st}, \mathsf{chall})$:

1: **if** $\mathsf{chall} == -1$ **then**
2: **return** $((E_2, R_2, S_2, P_2, Q_2, r_1), (a, b, r_3), (\alpha_1, r_5), (w, x, y, z))$;
3: **else**
4: Same as P_2 in Fig. 5.4.

$\mathsf{V}^*\Big(\big(\big((E_0, R_0, S_0, P_0, Q_0), (E_1, P_1, Q_1), \phi, \alpha\big), \mathsf{com}, \mathsf{chall}, \mathsf{resp}\Big)$:

1: **if** $\mathsf{chall} == -1$ **then**
2: $((E_2, R_2, S_2, P_2, Q_2, r_1), (a, b, r_3), (\alpha_1, r_5), (w, x, y, z)) = \mathsf{resp}$;
3-6: Same as V in Fig. 5.4.
7: Check $R_2 := [w]\psi(R_0) + [x]\psi(S_0), S_2 := [y]\psi(R_0) + [z]\psi(S_0)$;
8: **else**
9: Same as V in Fig. 5.4.

$\mathsf{P}_1((E_0, R, S, P_0, Q_0, E_1, P_1, Q_1), (\tilde{E}_0, \tilde{R}, \tilde{S}, \tilde{P}_0, \tilde{Q}_0, \tilde{E}_1, \tilde{P}_1, \tilde{Q}_1), k_0, k_1, \alpha, \alpha')$:

1: Sample random coefficients w, x, y, z such that $wz - xy \neq 0 \bmod d$;
2: Compute $\phi : E_0 \to E_0/\langle [k_0]R + [k_1]S\rangle \cong E_1$
3: Compute $\phi' : \tilde{E}_0 \to \tilde{E}_0/\langle [k_0]\tilde{R} + [k_1]\tilde{S}\rangle \cong \tilde{E}_1$
4: Run $\mathsf{P}_1^*((E_0, P_0, Q_0, E_1, P_1, Q_1), \phi, \alpha, (w, x, y, z))$ to get $\mathsf{st}, \mathsf{com}$;
5: Run $\mathsf{P}_1^*((\tilde{E}_0, \tilde{P}_0, \tilde{Q}_0, \tilde{E}_1, \tilde{P}_1, \tilde{Q}_1), \tilde{\phi}', \tilde{\alpha}', (w, x, y, z))$ to get $\tilde{\mathsf{st}}, \tilde{\mathsf{com}}$;
6: **return** $\big((\mathsf{st}, \tilde{\mathsf{st}}), (\mathsf{com}, \tilde{\mathsf{com}})\big)$;

$\mathsf{P}_2((\mathsf{st}, \tilde{\mathsf{st}}), \mathsf{chall})$:

1: **return** $(\mathsf{P}_2^*(\mathsf{st}, \mathsf{chall}), \mathsf{P}_2^*(\tilde{\mathsf{st}}, \mathsf{chall}))$;

$\mathsf{V}((E_0, R, S, P_0, Q_0, E_1, P_1, Q_1), (\tilde{E}_0, \tilde{R}, \tilde{S}, \tilde{P}_0, \tilde{Q}_0, \tilde{E}_1, \tilde{P}_1, \tilde{Q}_1),$
$(\mathsf{com}, \tilde{\mathsf{com}}), \mathsf{chall}, (\mathsf{resp}, \tilde{\mathsf{resp}}))$:

1: Set $v := \mathsf{V}^*((E_0, R, S, P_0, Q_0, E_1, P_1, Q_1), \mathsf{com}, \mathsf{chall}, \mathsf{resp})$;
2: Set $\tilde{v} := \mathsf{V}^*((\tilde{E}_0, \tilde{R}, \tilde{S}, \tilde{P}_0, \tilde{Q}_0, \tilde{E}_1, \tilde{P}_1, \tilde{Q}_1), \tilde{\mathsf{com}}, \mathsf{chall}, \tilde{\mathsf{resp}})$;
3: **return** $v \wedge \tilde{v}$;

Figure 5.6: **Top:** Modified proof of knowledge for the relation $\mathcal{R}_{\mathsf{iso}}$ where the basis randomness is explicit. The expressions in magenta denote the changes from Fig. 5.4. **Bottom:** Interactive proof of knowledge for the relation $\mathcal{R}^*_{\mathsf{par}}$.

Chapter 6

Supersingular Curves You Can Trust

Trust, but verify.
— Russian proverb,
popularized by Ronald Raegan.

This chapter is a verbatim reproduction of the following paper:
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: *EUROCRYPT 2023, Part II*. LNCS. Springer, Heidelberg, June 2023, pp. 405–437. DOI: 10.1007/978-3-031-30617-4_14

I contributed to the high-level design of the protocol and the zero-knowledge proof implementation, and I experimentally validated the results of Sec. 3.

Abstract: Generating a supersingular elliptic curve such that nobody knows its endomorphism ring is a notoriously hard task, despite several isogeny-based protocols relying on such an object. A trusted setup is often proposed as a workaround, but several aspects remain unclear. In this work, we develop the tools necessary to practically run such a distributed trusted-setup ceremony.

Our key contribution is the first statistically zero-knowledge proof of isogeny knowl-

edge that is compatible with any base field. To prove statistical ZK, we introduce isogeny graphs with Borel level structure and prove they have the Ramanujan property. Then, we analyze the security of a distributed trusted-setup protocol based on our ZK proof in the simplified universal composability framework. Lastly, we develop an optimized implementation of the ZK proof, and we propose a strategy to concretely deploy the trusted-setup protocol.

6.1 — Introduction

Be it foundationally or for efficiency, most of isogeny based cryptography is built upon supersingular elliptic curves [CLG09; JD11; CLMPR18; DMPS19; GPS20; DKLPW20; DDFKL+21]. At the heart of it, lies the *supersingular isogeny graph*: a graph whose vertices represent supersingular elliptic curves (up to isomorphism) and whose edges represent isogenies (up to isomorphism) of some fixed small prime degree between them. A foundational hard problem for isogeny based cryptography is then: given two supersingular elliptic curves, find a path in the supersingular isogeny graph connecting them.

An endomorphism is an isogeny from a curve E to itself, and their collection forms the *endomorphism ring* $\mathrm{End}(E)$. In recent years, the connection between finding isogeny paths and computing endomorphism rings of supersingular curves has become increasingly important [GPST16; EHLMP18; Wes22b; Wes22a]. It is now established that, assuming the generalised Riemann hypothesis, there exists probabilistic polynomial time algorithms for these two problems:

1. Given supersingular elliptic curves E_0, E_1 along with descriptions of their endomorphism rings, compute an isogeny path $E_0 \to E_1$;

2. Given a supersingular elliptic curve E_0 along with a description of its endomorphism ring, and given an isogeny path $E_0 \to E_1$, compute a description of the endomorphism ring of E_1.

These algorithms—and variants—have successfully been used both constructively [GPS20; DKLPW20; DDFKL+21] and for cryptanalysis [GPST16; Pet17; QKLMP+21; EHLMP18; DMPS19; FKMT22].

Without the additional information above, computing the endomorphism ring of an arbitrary supersingular curve remains a hard problem, both for classical and quantum computers. Given the importance of this problem, it is natural to ask whether it is possible to sample supersingular curves such that computing their endomorphism ring is a hard problem, crucially, even for the party who does the sampling. We shall call these objects *Supersingular Elliptic Curves of Unknown Endomorphism Ring*, or SECUER[1] in short.

Applications. Generating a SECUER has turned out to be a delicate task, and no such curve has ever been generated. Yet, several isogeny based schemes can only be instantiated with a SECUER. This is the case, for example, of isogeny based verifiable delay functions [DMPS19] and delay encryption [BD21]. The so-called CGL hash function based on supersingular curves [CLG09] has been shown to be broken by the knowledge of the endomorphism ring [EHLMP18], and one possible fix is to instantiate it with a SECUER. Other protocols which require a SECUER include hash proof systems, dual mode PKE [ADMP20], oblivious transfer [LGD21], and commitment schemes [Ste22].

Contributions. We analyze and put into practice a protocol for distributed generation of SECUERS. Our main technical contribution is a key ingredient of the protocol: a new proof of isogeny knowledge (two curves E_0 and E_1 being public, a party wishes to prove that they know an isogeny $E_0 \to E_1$ without revealing it). Our proof is similar to the SIDH proof of knowledge [DJP14; DDGZ22], but extends it in a way that makes it compatible

[1]The British spelling is SECURE.

with any base field, any walk length, and has provable statistical zero-knowledge (unlike any previous proof of isogeny knowledge). In particular, its statistical security makes it fully immune to the recent attacks [CD23; MMPPW23; Rob23].

To prove statistical security, we analyze *supersingular ℓ-isogeny graphs with level structure*, a generalization of isogeny graphs that was recently considered in [DKLPW20; Arp22]. We prove that these graphs, like classic isogeny graphs, possesses the Ramanujan property, a fact that is of independent interest. Using the property, we analyze the mixing behavior of random walks, which lets us give very precise parameters to instantiate the proof of knowledge at any given security level.

To show that the resulting protocol is practical, we implement it on top of Microsoft's SIDH library[2] and benchmark it for each of the standard SIKE primes [JACCD+20]. We must stress that SIDH-style primes are possibly the most favorable to our protocol, in terms of practical efficiency. Finally, we sketch a roadmap to run the distributed generation protocol for the SIKE primes in a real world setting with hundreds of participants.

Limitations. We must point out that our new proof of knowledge is not well adapted to a secure distributed generation protocol in the case where one wants to generate a SECUER defined over a prime field $\mathbb{F}_p$, instead of $\mathbb{F}_{p^2}$, such as in [ADMP20; LGD21]. Different proofs of knowledge [DG19; BKV19] could be plugged in the distributed protocol for the $\mathbb{F}_p$ case, however their practical usability is dubious.

6.1.1 – Generating a SECUER

The cornerstone of isogeny based cryptography is the endomorphism ring problem: if it could be solved efficiently, then all of supersingular isogeny based cryptography would be broken [GPST16; EHLMP18; Wes22a], leaving only ordinary isogeny based cryptography [Cou06; Sto10; DKS18] standing.

[2] https://github.com/microsoft/PQCrypto-SIDH

Definition 6 (Endomorphism ring problem)**.** *Given a supersingular curve $E/\mathbb{F}_{p^2}$, compute its endomorphism ring $\mathrm{End}(E)$. That is, compute an integral basis for a maximal order $\mathcal{O}$ of the quaternion algebra ramified at p and ∞, as well as an explicit isomorphism $\mathcal{O} \simeq \mathrm{End}(E)$.*

For any p, there exists a polynomially sized subset of all supersingular curves for which the endomorphism ring can be computed in polynomial time [CPV20; LB20], but the problem is believed to be exponentially hard in general, even for quantum computers. A related problem, commonly encountered in isogeny protocols, is finding paths in supersingular isogeny graphs.

Definition 7 (Isogeny ℓ-walk problem)**.** *Given two supersingular curves $E, E'/\mathbb{F}_{p^2}$ of the same order, and a small prime ℓ, find a walk from E to E' in the ℓ-isogeny graph.*

Such walks are always guaranteed to exist, as soon as they are allowed to have length in $O(\log(p))$ [Mes86; Piz90; Koh96; CLG09]. The two problems are known to be polynomial time equivalent, assuming GRH [Wes22b]. Indeed, given $\mathrm{End}(E)$ and $\mathrm{End}(E')$, it is easy to compute a path $E \to E'$. Reciprocally, given $\mathrm{End}(E)$ and a path $E \to E'$, it is easy to compute $\mathrm{End}(E')$; and, by random self-reducibility, we can always assume that one of $\mathrm{End}(E)$ or $\mathrm{End}(E')$ is known.

Our goal is to generate a SECUER: a curve for which the endomorphism ring problem is hard, and consequently one for which it is hard to find a path to any other given curve.

What does not work. The supersingular elliptic curves over a finite field k of characteristic p are those such that $\#E(k) = 1 \bmod p$. Any supersingular curve is isomorphic to one defined over a field with p^2 elements, thus, without loss of generality, we are only interested in supersingular curves defined over $\mathbb{F}_{p^2}$. Among the p^2 isomorphism classes of elliptic curves over $\mathbb{F}_{p^2}$, only $\approx p/12$ of them correspond to supersingular curves. The standard way to construct supersingular curves is to start from a curve with complex multiplication over a number field, and then reduce modulo p. Complex multiplication elliptic curves have supersingular reduction modulo 50% of the primes, thus this technique quickly produces supersingular curves for almost all primes. For example, the curve

$y^2 = x^3 + x$, which has complex multiplication by the ring $\mathbb{Z}[i]$ of Gaussian integers, is supersingular modulo p if and only if $p = 3 \bmod 4$. Most isogeny based protocols are instantiated using precisely this curve as starting point. These curves are not SECUERS, though, because from the information on complex multiplication one can compute the endomorphism ring in polynomial time [CPV20; LB20].

As p grows, the curves with computable[3] complex multiplication form only a negligible fraction of all supersingular curves in characteristic p, so we may still hope to get a SECUER if we can sample a supersingular curve at random from the whole set. The natural way to do so is to start from a well known supersingular curve, e.g. $E_0 : y^2 = x^3 + x$, take a random walk $E_0 \to E_1$ in the isogeny graph, and then select the arrival curve E_1. But, by virtue of the reductions mentioned above, any E_1 constructed this way cannot be called a SECUER either.

Several other techniques have been considered for generating SECUERS, however all attempts have failed so far [BBDFG+22; MMP22].

Distributed generation of SECUERS. An obvious solution that has been proposed for schemes that need a SECUER is to rely on a trusted party to start from a special curve E_0 and to perform an isogeny walk to a random curve E_1. Although E_1 is not a SECUER, if the trusted party keeps the walk $E_0 \to E_1$ secret, no one else will be able to compute $\mathrm{End}(E_1)$.

Of course, relying on a trusted third party is undesirable. The natural next step is to turn this idea into a distributed protocol with t parties generating a sequence of walks $E_0 \to E_1 \to E_2 \to \cdots \to E_t$. First, suppose that the sequence was generated honestly: the i-th party indeed generated a random isogeny from the previous curve E_{i-1} to a new curve E_{i+1}. Then it is sufficient for a *single* party to honestly discard their isogeny, for no path to be known by *anyone* from E_0 to E_t. Then, E_t is a SECUER for all practical purposes.

[3] Deuring showed that any supersingular curve can be lifted in several ways to a curve with complex multiplication, but for almost all curves computing such lifts has complexity exponential in $\log(p)$.

To make this protocol secure against active adversaries, an additional ingredient is needed. As it is, the last party could cheat as follows: instead of generating an isogeny $E_{t-1} \to E_t$, they could reboot the chain and generate an isogeny $E_0 \to E_t$. They could then compute the endomorphism ring of E_t. If only the curves E_i along the path are revealed, it is impossible to detect such misbehavior. To prevent this, each party needs to prove that they know their component of the walk: an isogeny $E_{i-1} \to E_i$ (as first discussed in [BD21]). To this end, we develop a statistically zero-knowledge proof of isogeny knowledge.

6.1.2 – Proof of isogeny knowledge

State-of-the-art. Protocols to prove knowledge of an isogeny have been mostly studied for signatures. The first such protocol is the SIDH-based proof of knowledge of [DJP14]. Its security proof was found to be flawed and then fixed, either by changing the assumptions [GPV21] or by changing the protocol [DDGZ22]. However, these protocols are now fully broken by the recent polynomial time attacks on SIDH-like protocols [CD23; MMPPW23; Rob23].

CSIDH-based proofs of knowledge were first introduced in [DG19], and then improved in [BKV19] for the parameter set CSIDH-512. These are limited to isogeny walks between curves defined over a prime field $\mathbb{F}_p$, and tend to be prohibitively slow outside of the specially prepared parameter set CSIDH-512.

Finally, De Feo and Burdges propose an efficient proof of knowledge tailored to finite fields used in delay protocols [BD21]. However the soundness of this protocol is only conjectural, and, being based on pairing assumptions, is broken by quantum computers. In summary, no general purpose, quantum-safe, zero-knowledge proof of knowledge of an isogeny walk between supersingular curves defined over $\mathbb{F}_{p^2}$ exists in previous literature.

Overview of our method. Our main technical contribution is a new proof of knowledge that ticks all the boxes above: it is compatible with any base field, any walk length, it has

provable statistical zero-knowledge, and is practical—as illustrated by our implementation. The idea is the following. Two elliptic curves E_0 and E_1 being public, some party, the prover, wishes to convince the verifier that they know an isogeny $\phi : E_0 \to E_1$ (of degree, say, 2^m, large enough so it is guaranteed that such an isogeny exists). First, the prover secretly generates a random isogeny walk $\psi : E_0 \to E_2$ of degree, say, 3^n. Defining ϕ' with kernel $\psi(\ker(\phi))$, and ψ' with kernel $\phi(\ker(\psi))$, one obtains the following commutative diagram:

$$\begin{array}{ccc} E_0 & \xrightarrow{\phi} & E_1 \\ {\scriptstyle\psi}\downarrow & & \downarrow{\scriptstyle\psi'} \\ E_2 & \xrightarrow[\phi']{} & E_3 \end{array} \tag{6.1}$$

Now, the prover publishes a hiding and binding commitment to E_2 and E_3. The verifier may now ask the prover to reveal one of the three isogenies ψ, ϕ', or ψ', by drawing a random $\mathsf{chall} \in \{-1, 0, 1\}$ (and open the commitment(s) corresponding to the relevant endpoints). For the prover to succeed with overwhelming probability, they must know all three answers, so they must know an isogeny from E_0 to E_1: the composition $\psi' \circ \varphi' \circ \psi : E_0 \to E_1$. This is the idea behind the soundness of the protocol.

So far, this protocol is more or less folklore and superficially similar to [DDGZ22, §5.3]. But does it leak any information? Whereas previous protocols only achieved computational zero-knowledge, we provide a tweak that achieves statistical zero-knowledge: there is a simulator producing transcripts that are statistically indistinguishable from a valid run of the protocol. The simulator starts by choosing the challenge chall *first*, then it generates an isogeny that is statistically indistinguishable from either ψ, φ', or ψ', according to the value of chall. Simulating ψ (or ψ') is straightforward: generate a random isogeny walk $\tilde{\psi}$ (or $\tilde{\psi}'$) of degree 3^n from E_0 (or from E_1). The isogeny $\tilde{\psi}$ is a *perfect* simulation of ψ. Simulating ϕ' seems trickier. An obvious approach is to first generate a random E_2 (for instance, by simulating $\psi : E_0 \to E_2$), then generate a random walk isogeny $\tilde{\varphi}' : E_2 \to E_3$ of degree 2^m. While this may seem too naive, we in fact prove that when $\deg(\psi)$ is large enough, the distribution of $\tilde{\varphi}'$ is statistically close to a honestly generated φ'. The key is

a proof that the isogeny graph enriched with so-called *level structure* has rapid mixing properties.

Isogeny graphs with level structure The isogeny ϕ' is essentially characterised by its source, E_2, and its kernel $\ker(\phi')$, a (cyclic) subgroup of order $\deg(\phi')$. We are thus interested in random variables of the form (E, C), where E is an elliptic curve, and C a cyclic subgroup of E, of order some integer d (not divisible by p). We call such a pair (E, C) a *level d Borel structure*.

The simulator proposed above essentially generates $\tilde{\phi}'$ as a uniformly random level 2^m Borel structure $(E, C) = (E_2, \ker(\tilde{\phi}'))$. On the other hand, a honestly generated ϕ' corresponds to a pair $(\psi(E_0), \psi(\ker \phi))$, and ψ is a uniformly random isogeny walk of degree 3^n. This process corresponds to a random walk of length n in the 3-*isogeny graph with level 2^m structure*, with starting point $(E_0, \ker \phi)$. We prove the following result.

Theorem 5. *Let $G = G(p, d, \ell)$ the supersingular ℓ-isogeny graph with level d Borel structure. The adjacency matrix A of G is diagonalizable, with real eigenvalues, and has the Ramanujan property, i.e the integer $\ell + 1$ is an eigenvalue of A of multiplicity one, while all the other eigenvalues are contained in the Hasse interval $[-2\sqrt{\ell}, 2\sqrt{\ell}]$.*

As a consequence, we prove that random walks quickly converge to the stationary distribution, so $\tilde{\phi}'$ and ϕ' are statistically indistinguishable.

Paper outline. We start in Section 6.2 with a few technical preliminaries on elliptic curves, isogenies, and proofs of knowledge. Section 6.3 is dedicated to the proof of Theorem 5. This section can be read independently from the rest. The reader only interested in applications, and willing to accept Theorem 5 (and its consequence on non-backtracking random walks, Theorem 8, page 188), can safely skip to the following section. This theoretical tool at hand, we then describe and analyse the new proof of isogeny knowledge in Section 6.4. We describe the protocol to generate a SECUER in Section 6.5, and prove its security. Finally, we report on our implementation in Section 6.6.

6.2 — Preliminaries

6.2.1 – General Notations

We write $x \leftarrow \chi$ to represent that an element x is sampled at random from a set/distribution $\mathcal{X}$. The output x of a deterministic algorithm $\mathcal{A}$ is denoted by $x = \mathcal{A}$ and the output x' of a randomized algorithm $\mathcal{A}'$ is denoted by $x' \leftarrow \mathcal{A}'$. For $a, b \in \mathbb{N}$ such that $a, b \geq 1$, we denote by $[a, b]$ (resp. $[a]$) the set of integers lying between a and b, both inclusive (the set of integers lying between 1 and a, both inclusive). We refer to $\lambda \in \mathbb{N}$ as the security parameter, and denote by $\mathsf{poly}(\lambda)$, $\mathsf{polylog}(\lambda)$ and $\mathsf{negl}(\lambda)$ any generic (unspecified) polynomial, poly-logarithmic or negligible function in λ, respectively.[4] For probability distributions $\mathcal{X}$ and $\mathcal{Y}$, we write $\mathcal{X} \approx \mathcal{Y}$ if the statistical distance between $\mathcal{X}$ and $\mathcal{Y}$ is negligible.

6.2.2 – Elliptic curves, isogenies and "SIDH squares"

We assume the reader has some familiarity with elliptic curves and isogenies. Throughout the text, p shall be a prime number, $\mathbb{F}_p$ and $\mathbb{F}_{p^2}$ the finite fields with p and p^2 elements respectively. Unless specified otherwise, all elliptic curves will be supersingular and defined over $\mathbb{F}_{p^2}$. We write $E[d]$ for the subgroup of d-torsion points of E over the algebraic closure.

Unless specified otherwise, all isogenies shall be separable. If G is a finite subgroup of E, we write $\phi : E \to E/G$ for the unique (up to post-composition with an isomorphism of E/G) separable isogeny with kernel G. If G is cyclic, we say the isogeny is cyclic. We denote by $\hat{\phi}$ the dual isogeny to ϕ. Separable isogenies and their duals can be computed and/or evaluated in time $\mathsf{poly}(\#G)$ using any of the algorithms in [Vél71; BDLS20], however in some cases, e.g. when $\#G$ only contains small factors, this cost may be lowered to as little as $\mathsf{polylog}(\#G)$.

[4] A function $f : \mathbb{N} \to \mathbb{N}$ is said to be negligible in λ if for every positive polynomial p, $f(\lambda) < 1/p(\lambda)$ when λ is sufficiently large.

Given separable isogenies $\phi : E_0 \to E_1$ and $\psi : E_0 \to E_2$ of coprime degrees, we obtain the commutative diagram in (6.1) by defining $\phi' : E_2 \to E_2/\psi(\ker(\phi))$ and $\psi' : E_1 \to E_1/\phi(\ker(\psi))$. Again, E_3 is only defined up to isomorphism. In categorical parlance, this is the *pushout* of ϕ and ψ, but cryptographers may know it better through its use in the SIDH key exchange. We refer to these commutative diagrams as *SIDH squares* or *SIDH ladders* (see Section 6.4.2 for more details).

6.2.3 – Proofs of Knowledge

Our main technical contribution is a Σ-protocol to prove knowledge of an isogeny of given degree between two supersingular elliptic curves. Recall a Σ-protocol for an NP-language $\mathcal{L}$ is a public-coin three-move interactive proof system consisting of two parties: a verifier and a prover. The prover is given a witness w for an element $x \in \mathcal{L}$, his goal is to convince the verifier that he knows w.

Definition 8 (Σ-protocol)**.** *A Σ-protocol Π_Σ for a family of relations $\{\mathcal{R}\}_\lambda$ parameterized by security parameter λ consists of PPT algorithms $(\mathsf{P}_1, \mathsf{P}_2, \mathsf{V})$ where V is deterministic and we assume $\mathsf{P}_1, \mathsf{P}_2$ share states. The protocol proceeds as follows:*

1. *The prover, on input $(x, w) \in \mathcal{R}$, returns a commitment com $\leftarrow \mathsf{P}_1(x, w)$ which is sent to the verifier.*
2. *The verifier flips λ coins and sends the result to the prover.*
3. *Call chall the message received from the verifier, the prover runs resp $\leftarrow \mathsf{P}_2(\mathit{chall})$ and returns resp to the verifier.*
4. *The verifier runs $V(x, \mathit{com}, \mathit{chall}, \mathit{resp})$ and outputs a bit.*

A transcript $(\mathsf{com}, \mathsf{chall}, \mathsf{resp})$ is said to be valid, or accepting, if $V(x, \mathsf{com}, \mathsf{chall}, \mathsf{resp})$ outputs 1. The main requirements of a Σ-protocol are:

Correctness: If the prover knows $(x, w) \in \mathcal{R}$ and behaves honestly, then the verifier outputs 1.

***n*-special soundness:** There exists a polynomial-time extraction algorithm that, given a statement x and n valid transcripts

$$(\mathsf{com}, \mathsf{chall}_1, \mathsf{resp}_1), \ldots, (\mathsf{com}, \mathsf{chall}_n, \mathsf{resp}_n)$$

where $\mathsf{chall}_i \neq \mathsf{chall}_j$ for all $1 \leq i < j \leq n$, outputs a witness w such that $(x, w) \in \mathcal{R}$ with probability at least $1 - \varepsilon$ for soundness error ε.

A special sound Σ-protocol for $\mathcal{R}$ is also called a *Proof of Knowledge (PoK)* for $\mathcal{R}$. Our Σ-protocol will have the peculiar property that the relation used to prove correctness turns out to be a subset of the one used to prove soundness. This will require extra care when proving security in Section 6.5.

Special Honest Verifier Zero-knowledge (SHVZK): There exists a polynomial-time simulator that, given a statement x and a challenge chall, outputs a valid transcript $(\mathsf{com}, \mathsf{chall}, \mathsf{resp})$ that is indistinguishable from a real transcript.

Definition 9. *A Σ-protocol $(\mathsf{P}_1, \mathsf{P}_2, \mathsf{V})$ is computationally special honest verifier zero-knowledge if there exists a probabilistic polynomial time simulator Sim such that for all probabilistic polynomial time stateful adversaries $\mathcal{A}$*

$$\Pr\left[\mathcal{A}(\mathit{com}, \mathit{chall}, \mathit{resp}) = 1 \;\middle|\; \begin{array}{l} (x, w, \mathit{chall}) \leftarrow \mathcal{A}(1^\lambda); \\ \mathit{com} \leftarrow \mathsf{P}_1(x, w); \\ \mathit{resp} = \mathsf{P}_2(\mathit{chall}) \end{array}\right]$$
$$\approx \Pr\left[\mathcal{A}(\mathit{com}, \mathit{chall}, \mathit{resp}) = 1 \;\middle|\; \begin{array}{l} (x, w, \mathit{chall}) \leftarrow \mathcal{A}(1^\lambda); \\ (\mathit{com}, \mathit{resp}) \leftarrow \mathsf{Sim}(x, \mathit{chall}) \end{array}\right].$$

If the adversary is unbounded, the protocol is said to be statistically SHVZK.

6.2.4 – Non-Interactive Zero-Knowledge Proofs

In this paper, we consider non-interactive zero-knowledge (NIZK) proofs in the random oracle model that satisfy correctness, computational extractability and statistical zero-knowledge.

Definition 10. ***(NIZK proofs.)*** *Let $\mathcal{R}$ be a relation and let the language $\mathcal{L}$ be a set of statements $\{\mathsf{st} \in \{0,1\}^n\}$ such that for each statement $\mathsf{st} \in \mathcal{L}$, there exists a corresponding witness wit such that $(\mathsf{st}, \mathsf{wit}) \in \mathcal{R}$. A non-interactive zero-knowledge (NIZK) proof system for $\mathcal{R}$ is a tuple of probabilistic polynomial-time (PPT) algorithms $\mathsf{NIZK} = (\mathsf{P}_{\mathsf{NIZK}}, \mathsf{V}_{\mathsf{NIZK}})$ defined as follows (we assume that all algorithms in the description below have access to a common random oracle; we omit specifying it explicitly for ease of exposition):*

- $\mathsf{P}_{\mathsf{NIZK}}(\mathsf{st}, \mathsf{wit})$*: A PPT algorithm that, given a statement $\mathsf{st} \in \{0,1\}^n$ and a witness wit such that $(\mathsf{st}, \mathsf{wit}) \in \mathcal{R}$, outputs a proof Π.*

- $\mathsf{V}_{\mathsf{NIZK}}(\mathsf{st}, \Pi)$*: A deterministic algorithm that, given a statement $\mathsf{st} \in \{0,1\}^n$ and a proof Π, either outputs 1 (accept) or 0 (reject).*

The following correctness and security properties should be satisfied:

Correctness. For any $(\mathsf{st}, \mathsf{wit}) \in \mathcal{R}$, letting $\Pi = \mathsf{P}_{\mathsf{NIZK}}(\mathsf{st}, \mathsf{wit})$, we must have $\mathsf{V}_{\mathsf{NIZK}}(\mathsf{st}, \Pi) = 1$.

Computational Extractability. There exists an efficient PPT extractor $\mathsf{Ext}_{\mathsf{NIZK}}$ such that for any security parameter $\lambda \in \mathbb{N}$ and for any *polynomially bounded* cheating prover P^* where: (i) $\mathsf{Ext}_{\mathsf{NIZK}}$ has rewinding access to P^*, and (ii) $\mathsf{P}_{\mathsf{NIZK}}$, $\mathsf{Ext}_{\mathsf{NIZK}}$ and P^* all have access to a common random oracle, letting $(\mathsf{st}, \Pi) \leftarrow P^*(1^\lambda)$ and $\mathsf{wit} = \mathsf{Ext}_{\mathsf{NIZK}}(\mathsf{st}, \Pi)$, if $\mathsf{V}_{\mathsf{NIZK}}(\mathsf{st}, \Pi) = 1$, then we must have $\Pr[(\mathsf{st}, \mathsf{wit}) \in \mathcal{R}] > 1 - \mathsf{negl}(\lambda)$.

Statistical Zero Knowledge. There exists an efficient PPT simulator $\mathsf{Sim}_{\mathsf{NIZK}}$ such that for any security parameter $\lambda \in \mathbb{N}$ and for any non-uniform *unbounded* "cheating" verifier $V^* = (V_1^*, V_2^*)$ where $\mathsf{P}_{\mathsf{NIZK}}$, V_1^* and V_2^* all have access to a common random oracle, and such that $\mathsf{Sim}_{\mathsf{NIZK}}$ is allowed programming access to the same random oracle, we have

$$\left|\Pr\left[V_2^*(\mathsf{st}, \Pi, \xi) = 1 \wedge (\mathsf{st} \in \mathcal{L})\right] - \Pr\left[V_2^*(\mathsf{st}, \widehat{\Pi}, \xi) = 1 \wedge (\mathsf{st} \in \mathcal{L})\right]\right| \leq \mathsf{negl}(\lambda),$$

where $(\mathsf{st}, \mathsf{wit}, \xi) \leftarrow V_1^*(1^\lambda)$, $\Pi \leftarrow \mathsf{P}_{\mathsf{NIZK}}(\mathsf{st}, \mathsf{wit})$, and $\widehat{\Pi} \leftarrow \mathsf{Sim}_{\mathsf{NIZK}}(\mathsf{st})$.

6.3 — Isogeny graphs and expansion

Let p be a prime and d an integer not divisible by p. An elliptic curve with level d Borel structure is a pair (E, C), where E is an elliptic curve defined over a field of characteristic p and C is an order d cyclic subgroup of $E[d]$. We say that two such pairs (E_1, C_1) and (E_2, C_2) are isomorphic if there exists an isomorphism $\phi : E_1 \to E_2$ such that $\phi(C_1) = C_2$. An automorphism of (E, C) is an isomorphism $(E, C) \to (E, C)$. They form the group $\mathrm{Aut}(E, C)$.

Let ℓ be a prime not dividing pd. The supersingular ℓ-isogeny graph with level d structure $G = G(p, d, \ell)$ is defined as follows. The set of vertices of G is a complete set $V = V(p, d) = \{(E_i, C_i)\}$ of representatives of the set of isomorphism classes of supersingular elliptic curves with a level d Borel structure defined over $\mathbb{F}_{p^2}$. We note that each such class over $\overline{\mathbb{F}_{p^2}}$ admits a model defined over $\mathbb{F}_{p^2}$: Each isomorphism class of supersingular elliptic curves has a representative E such that $\#E(\mathbb{F}_{p^2}) = (p+1)^2$ and thus the p^2-Frobenius acts as a scalar multiplication $[-p]$, so the kernel of any ℓ-isogeny is $\mathrm{Gal}(\overline{\mathbb{F}_{p^2}})$-invariant.

Now, the set of edges from (E, C) to (E', C') in G is the set of degree ℓ isogenies from E to E' which map C to C', modulo the action of $\mathrm{Aut}(E', C')$ (by postcomposition). The number of edges is independent of the representative of the isomorphism classes. When $d = 1$, we recover the usual definition of the supersingular ℓ-isogeny graph.

This graph is directed. The out-degree of each vertex is $\ell + 1$, however the in-degree is not always $\ell + 1$, hence the adjacency matrix of the graph is not always symmetric.

6.3.1 – Generalities on the graph and its adjacency matrix

Let $V = \{(E_i, C_i)\}$ for $i = 1, \dots, n$ be the vertex set of $G = G(p, d, \ell)$. On the complex vector the space $\mathbb{C}^V$, we introduce the Hermitian form $Q((E_i, C_i), (E_j, C_j)) = w_i \delta_{ij}$, where δ_{ij} is the Kronecker symbol and $w_i := \frac{1}{2} |\operatorname{Aut}(E_i, C_i)|$. Denote by $|\cdot|_Q$ the associated norm. We compare will compare $|\cdot|_Q$ with the L^1 and L^2 norms on $\mathbb{C}^V$. The set Ω of probability distributions on V is the set of vectors with real positive entries and L^1 norm equal to 1. Consider also the vector $\mathcal{E} = \sum_{i=1}^n \frac{1}{w_i}(E_i, C_i)$, and s the probability distribution obtained normalizing $\mathcal{E}$. The following result contains a number of general facts about the adjacency matrix of G, which will be used later on.

Theorem 6. *1. The adjacency matrix A of G is self-adjoint with respect to Q; in particular it is diagonalizable with real eigenvalues and eigenvectors;*

2. The vector $\mathcal{E}$ is a left-eigenvector of eigenvalue $\ell + 1$ of A;

3. The vector u with all entries equal to 1 is a right-eigenvector of A; in particular its orthogonal complement S with respect to the L^2 scalar product is preserved by right multiplication by A;

4. $K := \inf\{ |v|_Q \,:\, v \in \mathbb{C}^V \text{ and } |v|_{L^1} = 1\} = \left(\frac{(p-1)d}{12} \prod_q (1 + \frac{1}{q})\right)^{-1/2}$, where, in the product, q runs over the prime divisors of d;

5. $M := \sup\{ |\pi - s|_Q \,:\, \pi \in \Omega\} \leq \sqrt{3}$.

Proof. See Appendix 6.A. □

6.3.2 – Proof of Theorem 5

We now prove that $G = G(p, d, \ell)$ has the Ramanujan property. This follows from the first three items of Theorem 6 combined with the following result, whose proof heavily relies

on the theory of modular forms.

Theorem 7. *Let $S \subset \mathbb{C}^V$ be the subspace of vectors $\sum_i v_i(E_i, C_i)$ such that $\sum_i v_i = 0$, as in Theorem 6. The eigenvalues of the action of A on S are all contained in the Hasse interval $[-2\sqrt{\ell}, 2\sqrt{\ell}]$.*

To prove Theorem 7, we assume standard notations and results about quadratic forms and modular forms, such as the ones from [DS05; Sch74; HPS89]. Given two elliptic curves with level structure (E_i, C_i) and (E_j, C_j), we denote by Λ_{ij} the lattice of isogenies $\phi\colon E_i \to E_j$ such that $\phi(C_i) \subset C_j$. The degree defines a quadratic form $\deg$ on Λ_{ij}. This quadratic module has rank four, level dp and determinant d^2p^2. We can thus define the theta series

$$\Theta_{ij}(\tau) = \frac{1}{|\operatorname{Aut}(E_j, C_j)|} \sum_{\phi \in \Lambda_{ij}} q^{\deg(\phi)}, \quad \text{with } q = e^{2\pi i \tau}.$$

This function is in $M_2(\Gamma_0(dp))$, the space of modular forms of weight two for the modular group $\Gamma_0(dp)$, by [HPS89, Theorem 4.2] (observe that in *loc. cit.* the exponential is one because $Q(h)$ is an integer; moreover, we choose $P = 1$) or [Sch74, Chapter IX, Theorem 5, page 218]. The above construction extends to an Hermitian pairing

$$\Theta\colon \mathbb{C}^V \otimes \mathbb{C}^V \to M_2(\Gamma_0(dp)) : ((\alpha_i)_i \otimes (\beta_j)_j) \longmapsto \sum_{i,j} \alpha_i \beta_j \Theta_{ij}.$$

We call this pairing the Brandt pairing, even though there is a little ambiguity[5] in this set-up. The Brandt pairing is non-degenerate: let $v = \sum c_i(E_i, C_i)$, then the coefficient of q of $\Theta(v, v)$ is the Hermitian norm of the vector of coefficients $(\dots, c_i, \dots)$. We will prove the following two key propositions.

Proposition 1. *The Brandt pairing intertwines the adjacency matrix A of G and the Hecke operator T_ℓ; in symbols $T_\ell \Theta(w, v) = \Theta(wA, v)$ for all $w, v \in \mathbb{C}^V$.*

[5] Rather than using the condition $\phi(C_i) \subset C_j$, we could have defined Λ_{ij} using $\phi(C_i) = C_j$. The second definition does not give a lattice but still permits to define a pairing. This second pairing generalizes to all level structures, so it might deserve better the name of Brandt pairing. However, the second pairing gives a more complicated proof in the Borel case, so we have opted for the first one.

Proposition 2. *For every three elliptic curves with level structure* (E_1, C_1)*,* (E_2, C_2) *and* (E_3, C_3)*, the modular form* $\Theta((E_1, C_1), (E_3, C_3)) - \Theta((E_2, C_2), (E_3, C_3))$ *is a cusp form.*

The combination of these two results tells that the spectrum of the action of A restricted to S is contained into the spectrum of the action of the Hecke operator T_ℓ on the space of cusp modular forms of weight two for $\Gamma_0(dp)$. The Ramanujan Conjecture, proved by Eichler, predicts that this second spectrum is contained in the Hasse interval, and hence proves Theorem 7.

We refer to [Del80, Theorem 8.2] for a proof of the Ramanujan Conjecture. In *loc. cit.* this result is proven only for eigenvectors of T_ℓ which are new-forms. An eigenvector which is an old form will come from an embedding $\iota\colon S_2(\Gamma_0(m)) \to S_2(\Gamma_0(dp))$ with m that divides dp. Since ℓ is coprime with dp, the map ι is T_ℓ-equivariant (cf. [DS05, proof of Proposition 5.6.2]), so we can still deduce our result from [Del80, Theorem 8.2]. It is worth recalling that [Del80, Theorem 8.2] is way more general that what we need, as it applies to modular forms of every weight.

Proof of Proposition 1 We prove that both sides have the same q-expansions. For a power series $F \in \mathbb{C}[[q]]$, denote $a_n(F)$ the coefficient of q^n. By definition

$$a_n(\Theta((E_i, C_i), (E_j, C_i))) = |\operatorname{Aut}(E_j, C_j)|^{-1} \cdot |\operatorname{Hom}^n((E_i, C_i), (E_j, C_j))|\,,$$

where $\operatorname{Hom}^n((E_i, C_i), (E_j, C_j))$ is the set of degree n isogenies in Λ_{ij}. For $f \in M_2(\Gamma_0(dp))$, we have $a_n(T_\ell f) = a_{n\ell}(f) + \ell a_{n/\ell}(f)$ (see e.g. [DS05, Proposition 5.2.2]), where $a_{n/\ell}(f)$ is set to zero in the case $n/\ell \notin \mathbb{Z}$. In particular,

$$\begin{aligned} &a_n(T_\ell\, \Theta((E_i, C_i), (E_j, C_j))) = \\ &= |\operatorname{Aut}(E_j, C_j)|^{-1} \Big(|\operatorname{Hom}^{n\ell}((E_i, C_i), (E_j, C_j))| + \ell |\operatorname{Hom}^{n/\ell}((E_i, C_i), (E_j, C_j)| \Big) \end{aligned} \tag{6.2}$$

On the other side,

$$\begin{aligned} a_n(\Theta((E_i,C_i)A,(E_j,C_j))) &= \sum_C a_n(\Theta((E_i/C,\pi_C(C_i)),(E_j,C_j))) = \\ &= |\operatorname{Aut}(E_j,C_j)|^{-1} \sum_C |\operatorname{Hom}^n((E_i/C,\pi_C(C_i)),(E_j,C_j))| \end{aligned} \tag{6.3}$$

where C varies among the cyclic non-trivial subgroups of $E_i[\ell]$ of cardinality ℓ, and π_C is the projection $E_i \to E_i/C$. For each C let

$$\begin{aligned} F_C\colon \operatorname{Hom}^n((E_i/C,\pi_C(C_i)),(E_j,C_j)) &\longrightarrow \operatorname{Hom}^{n\ell}((E_i,C_i),(E_j,C_j)) \\ f &\longmapsto f\circ\pi_C\,, \end{aligned}$$

and let F be the disjoint union of the above maps. The map F is surjective: if $\alpha\colon (E_i,C_i)\to(E_j,C_j)$ has degree $n\ell$, then $\ker(\alpha)\cap E_i[\ell]\neq\{0\}$, hence there exists a cyclic non-trivial $C\subset\ker(\alpha)\cap E_i[\ell]$, and we can write $\alpha = f\circ\pi_C$. In particular, let us compute the cardinality of the fiber $F^{-1}(\alpha)$ for α in the codomain. Each F_C is injective, hence $|F^{-1}(\alpha)|$ is equal to the number of subgroups C such that $F_C^{-1}(\alpha)$ is not empty, that is the number of subgroups C contained in $\ker(\alpha)\cap E_i[\ell]$. Hence

$$|F^{-1}(\alpha)| = \begin{cases} \ell+1 & \text{if } \alpha=\ell\beta \text{ for some } \beta\in\operatorname{Hom}^{n/\ell}((E_i,C_i),(E_j,C_j)), \\ 1 & \text{otherwise} \end{cases}$$

By (6.3), the domain of F has cardinality exactly $|\operatorname{Aut}(E_j,C_j)|\cdot a_n(\Theta(A(E_i,C_i),(E_j,C_j)))$, hence the proposition follows from (6.2) together with the above formula summed over α in the codomain.

Proof of Proposition 2 We have to show that, for any two pairs (E,C) and (E',C') and any cusp of $X_0(dp)$, the residue r of $\Theta((E,C),(E',C'))d\tau$ does not depend on (E,C) and (E',C') at the cusp but only on p, d and the cusp.

By the discussion in [DS05, Section 3.8, page 103] each cusp can be represented as

$\binom{a}{c}$ with c dividing dp, and r is equal to $a_0(\Theta((E,C),(E',C'))|_M)$ for M any matrix in $\mathrm{SL}_2(\mathbb{Z})$ of the form $\left(\begin{smallmatrix} a & b \\ c & \delta \end{smallmatrix}\right)$.

By [Sch74, Chapter IX, Equation (21), page 213], we have

$$r = \frac{1}{c^2 pd} \sum_{\nu,\lambda \in \Lambda/c\Lambda} e\left(\frac{(a-1)\deg(\lambda) + \deg(\lambda+\nu) + (\delta-1)\deg(\nu)}{c}\right)$$

where $e(z) = e^{2\pi i z}$, and Λ is the lattice of isogenies from (E,C) to (E',C') which map C into C'. The above formula tells us that r only depends on M and on the quadratic form $\deg\colon \Lambda/c\Lambda \to \mathbb{Z}/c\mathbb{Z}$. Writing $c = c_0 p^\epsilon$ with c_0 dividing N and $\epsilon = 0, 1$ and using the Chinese remainder theorem we can split the quadratic form in two parts

$$\Lambda/c\Lambda = \Lambda/c_0\Lambda \times \Lambda/p^\epsilon\Lambda \xrightarrow{\deg \times \deg} \mathbb{Z}/c_0\mathbb{Z} \times \mathbb{Z}/p^\epsilon\mathbb{Z} \cong \mathbb{Z}/c\mathbb{Z}\,.$$

The quadratic module $(\Lambda/c_0\Lambda, \deg)$ is (non-canonically) isomorphic to a Borel subalgebra of $(\mathrm{End}((\mathbb{Z}/c_0\mathbb{Z})^{\oplus 2}), \det)$. An isomorphism can be obtained mapping it to $\mathrm{Hom}(E[c_0], E'[c_0])$, and then choosing a symplectic basis.

If $\epsilon = 0$ we are done, otherwise $\epsilon = 1$. Since $[\mathrm{Hom}(E,E') : \Lambda] = d$ is prime to p, we have $\Lambda/p = \mathrm{Hom}(E,E')/p = (\mathrm{Hom}(E,E') \otimes \mathbb{Z}_p)/p$, and the quadratic $\mathbb{Z}_p$-module $\mathrm{Hom}(E,E') \otimes \mathbb{Z}_p$ does not depend on the pair because, by the Deuring correspondence (see e.g. [Voi21, Theorem 42.3.2.]), together with [Voi21, Lemma 19.6.6], it is isomorphic to $\lambda\mathcal{O}_p$ with the reduced norm, where $\mathcal{O}_p$ is the maximal order in the non-ramified quaternions over $\mathbb{Q}_p$, and λ is an element of norm prime to p.

6.3.3 – Mixing time of non-backtracking walks

We finally analyze the behavior of random walks in $G = G(p,d,\ell)$, which we will ultimately use to prove statistical indistinguishability of distribution arising from our proof of knowledge. First, observe that Theorem 6 item 2 shows that the probability distribution s introduced in Subsection 6.3.1 is the stationary distribution on G. This is nearly the

uniform distribution: all curves are equally likely, with the possible exception of the two curves with extra automorphisms, $j = 1728$ and $j = 0$, which are respectively twice and thrice less likely.

We are going to determine the speed at which random walks converge to the stationary distribution. We focus on non-backtracking walks, which are the most useful for cryptographic protocols, but, because the graph is directed, we need some care to define them. Edges of G are equivalence classes of isogenies, so we choose a representative for each class. For an edge α we define its dual edge as the chosen representative β for the class $\mathrm{Aut}(E, C)\hat{\alpha}$, so that $\beta\alpha = u\ell$ for $u \in \mathrm{Aut}(E, C)$. Notice that the dual of β (as an edge) might be different from α, but this is not relevant for us. We say that a random walk on G is non-backtracking walk if an edge is never followed by its dual.

With this "duality", we have that isogenies of degree a power of ℓ and with cyclic kernel (up to the equivalence $\alpha \sim \beta$ iff $\ker \alpha = \ker \beta$) correspond to non-backtracking walks.

Theorem 8 (Mixing time)**.** *Let π be a probability distribution on G, and $\pi^{(k)}$ the distribution obtained after a non-backtracking random walk of length k. Then we have*

$$d_{TV}(\pi^{(k)}, s) \leq \frac{1}{2} K^{-1} M \frac{(\ell+1)(k+1) - 2}{(\ell+1)\sqrt{\ell^k}},$$

where K and M are as in Theorem 6.

Proof. This follows from [ABLS07] for the case of undirected graphs. In Appendix 6.A we adapt the proof to the graph $G(p, d, \ell)$. □

6.4 — Proof of Knowledge

Our goal is to provide a PoK of an isogeny walk $\phi : E_0 \to E_1$ between two supersingular curves defined over $\mathbb{F}_{p^2}$ that can be seamlessly plugged in a distributed SECUER generation protocol. For this, we need the following properties:

1. Compatible with any pair of curves (E_0, E_1); this rules out [GPS17; GPS20], which is restricted to a special starting curve E_0, and [DG19] and derivatives, which are restricted to curves defined over $\mathbb{F}_p$.

2. Statistically ZK, so that the security of the final SECUER does not hinge on computational assumptions brought in by the PoK; this rules out all other isogeny based PoKs in the literature.

3. Post-quantum secure, possibly relying on as few additional assumptions as possible; this rules out many generic ZK proof systems.

4. Possibly compatible with any walk length and any base field $\mathbb{F}_{p^2}$.

5. Usable in practice for cryptographically sized finite fields.

The only attempt at using generic proof systems to prove knowledge of isogeny walks has been made in [CRT22], and is based on a SNARG derived from a Sumcheck protocol carefully optimized for isogenies. However this work does not consider ZK, and does not evaluate the concrete efficiency of the SNARG. Even if it could be made efficient, adding post-quantum ZK would likely come at a considerable cost, thus we do not investigate this path further.

Our new PoK inherits from the SIDH-based Σ-protocol of De Feo, Jao and Plût [DJP14], and from the recent developments of De Feo, Dobson, Galbraith and Zobernig [DDGZ22]. The common theme to all of them is to construct random SIDH squares on top of the secret isogeny $\phi : E_0 \to E_1$

$$\begin{array}{ccc} E_0 & \xrightarrow{\phi} & E_1 \\ {\scriptstyle\psi}\downarrow & & \downarrow{\scriptstyle\psi'} \\ E_2 & \xrightarrow{\phi'} & E_3 \end{array}$$

and to reveal some, but not all of the edges ψ, ψ', ϕ' in response to a challenge. The reason these protocols are not statistically ZK is that the side ϕ' is strongly correlated to the parallel side ϕ (often unique given E_2) and can thus easily be distinguished by an

unbounded adversary. Our first idea is to *make the walk ψ long enough* that the distribution of (E_2, ϕ') becomes statistically close to the uniform distribution on supersingular curves with isogenies of degree $\deg(\phi)$. To prove it, we will use the properties of isogeny graphs with level structure analyzed in Section 6.3.

But making ψ longer is easier said than done. SIDH-based protocols are constrained in the lengths of ϕ and ψ by the form of the prime p: typically, $p + 1 = 2^a 3^b$ and then $\deg(\phi) = 2^a$ and $\deg(\psi) = 3^b$. Our second idea is to *glue several SIDH squares together* to make longer walks (see Fig. 6.2). We call these larger diagrams *SIDH ladders*.

A valuable side-effect of gluing SIDH squares together is that we can free ourselves from the constraints on p. All we need is that isogenies of a small prime degree ℓ coprime to $\deg(\phi)$ can be computed efficiently, then we stack vertically sufficiently many SIDH squares to make $\deg(\psi) = \ell^n$ as large as we need. In practice, we will take $\deg(\phi) = 2^m$, $\deg(\psi) = 3^n$, and the protocol will be most efficient for SIDH primes, but in full generality our protocol works for any base field and any isogeny degree.

6.4.1 – Protocol description and analysis

Let E_0, E_1 be supersingular curves defined over a finite field $\mathbb{F}_{p^2}$, and let $\phi : E_0 \to E_1$ be a cyclic separable isogeny of smooth degree d. Let ℓ be a small prime not dividing pd. Let $\mathsf{C}(m; r)$ be a statistically hiding and computationally binding commitment scheme. Our Σ-protocol is described in Fig. 6.1; it depends on a parameter n, controlling the length of the ℓ-isogeny walks, that we will determine in Definition 12. The prover consists of two stateful algorithms $(\mathsf{P}_1, \mathsf{P}_2)$: the former is randomized and produces a commitment $(\mathsf{com}_2, \mathsf{com}_3)$, the latter receives a ternary challenge $\mathsf{chall} \in \{-1, 0, 1\}$ and produces a deterministic response resp. The verifier is a deterministic algorithm that receives $\big((\mathsf{com}_2, \mathsf{com}_3), \mathsf{chall}, \mathsf{resp}\big)$ and outputs a bit indicating whether or not the proof is accepted.

$\mathsf{P}_1(E_0, E_1, \phi, n)$:

1: Sample a random cyclic isogeny $\psi : E_0 \to E_2$ of degree ℓ^n;
2: Construct the SIDH ladder $(E_0, E_1, E_2, E_3, \phi', \psi')$ on (ϕ, ψ);
3: Sample random strings r_2, r_3;
4: **return** $\big(\mathsf{C}(E_2; r_2), \mathsf{C}(E_3; r_3)\big)$.

$\mathsf{P}_2(\mathsf{chall})$:

1: **if** $\mathsf{chall} == -1$ **then**
2: **return** (ψ, E_2, r_2);
3: **else if** $\mathsf{chall} == 1$ **then**
4: **return** (ψ', E_3, r_3);
5: **else if** $\mathsf{chall} == 0$ **then**
6: **return** $(\phi', E_2, r_2, E_3, r_3)$.

$\mathsf{V}(E_0, E_1, d, n, (\mathsf{com}_2, \mathsf{com}_3), \mathsf{chall}, \mathsf{resp})$:

1: **if** $\mathsf{chall} == -1$ **then**
2: $(\psi, E_2, r_2) = \mathsf{resp}$;
3: Check $\mathsf{com}_2 = \mathsf{C}(E_2; r_2)$;
4: Check ψ is an ℓ^n-isogeny $E_0 \to E_2$;
5: **else if** $\mathsf{chall} == 1$ **then**
6: $(\psi', E_3, r_3) = \mathsf{resp}$;
7: Check $\mathsf{com}_3 = \mathsf{C}(E_3; r_3)$;
8: Check ψ' is a cyclic d-isogeny $E_1 \to E_3$;
9: **else if** $\mathsf{chall} == 0$ **then**
10: $(\phi', E_2, r_2, E_3, r_3) = \mathsf{resp}$;
11: Check $\mathsf{com}_2 = \mathsf{C}(E_2; r_2)$;
12: Check $\mathsf{com}_3 = \mathsf{C}(E_3; r_3)$;
13: Check ϕ' is an ℓ^n-isogeny $E_2 \to E_3$.

Figure 6.1: Interactive proof of knowledge of a cyclic isogeny $\phi : E_0 \to E_1$ of degree d.

Proposition 3. *The Σ-protocol in Fig. 6.1 is correct for the relation*

$$\mathcal{R}_d = \{((E_0, E_1), \phi) \mid \phi : E_0 \to E_1 \text{ is a cyclic } d\text{-isogeny}\}.$$

Assuming the commitment C is computationally binding, it is 3-special sound for the relation

$$\mathcal{R}^\star = \{((E_0, E_1), \chi) \mid \chi : E_0 \to E_1 \text{ is a cyclic } \ell^{2i}d\text{-isogeny for some } 0 \leq i \leq n\}.$$

More precisely, there is a probabilistic polynomial time algorithm that, given three successful transcripts of the protocol with same commitments and distinct challenges, either recovers a witness $\chi : E_0 \to E_1$, or opens one of the commitments $\mathsf{C}(E_i; r_i)$ to two distinct values (breaking the binding property).

Proof. Correctness. Suppose that the prover $\mathsf{P} = (\mathsf{P}_1, \mathsf{P}_2)$ and the verifier V follow the protocol. First note that, since the degree d of ϕ is smooth, the SIDH ladder in P_1 can be constructed as described in Section 6.4.2. Then it is clear that the commitments open successfully, and the verifier accepts the transcript for any challenge.

3-special soundness. Given three accepting transcripts $(\mathsf{com}, -1, \mathsf{resp}_{-1})$, $(\mathsf{com}, 0, \mathsf{resp}_0)$

and $(\mathsf{com}, 1, \mathsf{resp}_1)$, recover $(\phi', E_2, r_2, E_3, r_3) = \mathsf{resp}_0$ where $\phi' : E_2 \to E_3$ is an isogeny. If the curves in resp_{-1} and resp_1 are not equal to E_2 and E_3 respectively, then we can open one of the commitments $\mathsf{C}(E_2; r_2)$ or $\mathsf{C}(E_3; r_3)$ to two distinct outputs. Otherwise, we have $\mathsf{resp}_{-1} = (\psi, E_2, r_2)$ and $\mathsf{resp}_1 = (\psi', E_3, r_3)$ where $\psi : E_0 \to E_2$ and $\psi' : E_1 \to E_3$ are isogenies. Therefore $\chi' = \widehat{\psi'} \circ \phi' \circ \psi$ is an isogeny from E_0 to E_1 of degree $\ell^{2n}d$. Factoring out the non-cyclic part of χ', we extract a cyclic isogeny $\chi : E_0 \to E_1$ of degree $\ell^{2i}d$ such that $\chi' = [\ell^{2(n-i)}] \circ \chi$ for some $0 \leq i \leq n$; however, like in the original SIDH PoK [DDGZ22; GPV21], we cannot guarantee that $i = 0$. □

We are now going to define the simulator for proving ZK. Simulating $\mathsf{chall} = \pm 1$ is easy, however how well we can simulate the case $\mathsf{chall} = 0$ depends on the parameter n given to P_1. The opening $(E_2, \phi' : E_2 \to E_3)$ can be equivalently viewed as the curve with level d Borel structure $(E_2, \ker(\phi'))$. Our goal is to have this opening distributed like a "random" vertex in the graph $G = G(p, d, \ell)$. To this effect, we define two sequences $D_1(k)$ and $D_2(k)$ of probability distributions on G, and we show that they converge as k grows.

Definition 11. *Let $\phi\colon E_0 \to E_1$ be a cyclic separable isogeny of degree d. Define*

$$\begin{aligned} \mathcal{D}_1(k) &= \big\{(E_0/K, \tau(\ker(\phi)) \quad \big| \; K \leftarrow \mathcal{C}_E(\ell^k),\, \tau : E_0 \to E_0/K\big\}, \\ \mathcal{D}_2(k) &= \big\{(E_0/K, C) \qquad\quad \big| \; K \leftarrow \mathcal{C}_E(\ell^k),\, C \leftarrow \mathcal{C}_{E_0/K}(d)\big\}, \end{aligned} \tag{6.4}$$

where $\mathcal{C}_E(f)$ is the uniform distribution on the cyclic subgroups of order f of E, up to $\mathrm{Aut}(E)$.

Lemma 18. *Keep notations as above, fix a positive real number ε, and let k be a positive integer such that*

$$\tau(p, d, \ell, k) = \tfrac{1}{4}(p-1)^{1/2}\Big(1 + \sqrt{d} \prod_{\substack{q \mid d \\ q \text{ prime}}} (1+\tfrac{1}{q})^{1/2}\Big) \cdot \Big(k + \tfrac{\ell-1}{\ell+1}\Big) \cdot \ell^{-k/2} \leq \varepsilon\,,$$

then $d_{TV}\left(\mathcal{D}_1(k), \mathcal{D}_2(k)\right) \leq \varepsilon$, where d_{TV} is the total variation distance between the two distributions, also known as statistical distance.

Proof. We bound the statistical distance of each of $\mathcal{D}_1(k)$ and $\mathcal{D}_2(k)$ from the stationary distribution of $G(p, d, \ell)$, as determined in Theorem 6, then we conclude with the triangle inequality. For $\mathcal{D}_1(k)$, we can directly apply Theorem 8. The argument for $\mathcal{D}_2(k)$ is slightly more involved and is presented in Appendix 6.A. □

Definition 12. *Given p, d, ℓ and m, define*

$$n(p, d, \ell, m) = \min\left\{k \in \mathbb{Z} \mid \tau(p, d, \ell, k) \leq 2^{-m}\right\}.$$

Proposition 4. *Let λ be a security parameter and let $n = n(p, d, \ell, \lambda)$. The Σ-protocol of Fig. 6.1 is statistically SHVZK for the relation $\mathcal{R}_d$ defined in Proposition 3, assuming the commitment C is statistically hiding.*

Proof. We simulate the honest prover for each of the three challenges as follows.

chall $= -1$. Sample a random isogeny $\psi : E_0 \to E_2$ of degree ℓ^n, and random strings r_2, r_3. Set $\mathsf{com}_2 = \mathsf{C}(E_2; r_2)$ and set $\mathsf{com}_3 = \mathsf{C}(\perp; r_3)$. Return $(\mathsf{com}_2, \mathsf{com}_3), \mathsf{chall}, (\psi, E_2, r_2)$.

The isogeny ψ is distributed exactly like in the real protocol, thus this transcript is valid. Because C is statistically hiding, an adversary cannot distinguish com_3 from a real commitment.

chall $= 1$. This is nearly identical to the above. The simulator samples $\psi' : E_1 \to E_3$ of degree ℓ^n and random strings r_2, r_3. It sets $\mathsf{com}_2 = \mathsf{C}(\perp; r_2)$ and $\mathsf{com}_3 = \mathsf{C}(E_3; r_3)$, and returns $(\mathsf{com}_2, \mathsf{com}_3), \mathsf{chall}, (\psi', E_3, r_3)$.

Because ℓ is coprime to d, if ψ is uniformly distributed so is ψ'. Then, the transcript is indistinguishable from a real one as before.

chall $= 0$. Sample a random isogeny $\psi : E_0 \to E_2$ of degree ℓ^n, and then a random isogeny $\rho : E_2 \to E_3$ of degree d. Sample random strings r_2, r_3 and set $\mathsf{com}_2 = \mathsf{C}(E_2; r_2)$ and $\mathsf{com}_3 = \mathsf{C}(E_3; r_3)$. Return $(\mathsf{com}_2, \mathsf{com}_3), \mathsf{chall}, (\rho, E_2, r_2, E_3, r_3)$.

Thanks to Lemma 18, the statistical distance between the simulated $(E_2, \ker(\rho))$ and $(E_2, \psi(\ker(\phi)))$ is negligible. Because ρ is uniquely determined from $\ker(\rho)$, and the real response ϕ' by $\psi(\ker(\phi))$, an adversary has negligible probability of distinguishing the transcript output by the simulator. □

6.4.2 – Executing the protocol

The protocol we just described crucially depends on the ability to construct a commutative square with sides of degrees d and ℓ^n. The SIDH setting has $p + 1 = d \cdot \ell^n$ so that the square can be constructed by simply pushing a single kernel point for ψ through ϕ and vice versa. We refer to such a square as an *SIDH square*. For more general choices of ℓ^n and d, the kernels are typically generated by points defined over very large extension fields, requiring superpolynomial space. We efficiently construct such "larger" squares by gluing together several SIDH squares in what we call *SIDH ladders*, as depicted in Fig. 6.2.

For simplicity, we shall present the case $d = (2^a)^w$ and $\ell^n = (3^b)^h$, where 2^a and 3^b are the side lengths of an SIDH square, and w and h are positive integers defining the **w**idth and **h**eight of the ladders in units of SIDH squares. However, the technique generalizes easily to any coprime d and ℓ^n, as long as isogenies of degrees d and ℓ can be efficiently computed.

First, notice that there always exist some choice of a and b such that points (and hence kernel subgroups) of orders 2^a and 3^b can be represented efficiently. This is clear if the prime p is a SIDH prime where $2^a 3^b \mid (p + 1)$, but for a generic prime p, one can set $a = b = 1$: Points of order 2 and 3 are defined over a small extension field and can thus be efficiently represented. Moreover, any isogeny of degree $(3^b)^h$ is the composition of h isogenies of degree 3^b each, which can be stored as a sequence of h kernel generators which are efficiently representable.

This means that the prover can generate the isogeny $\psi : E_0 \to E_2$ in step 2 of P_1 by generating a random kernel $K_{1,0}$ on E_0, computing the isogeny $\psi_{1,0} : E_0 \to$

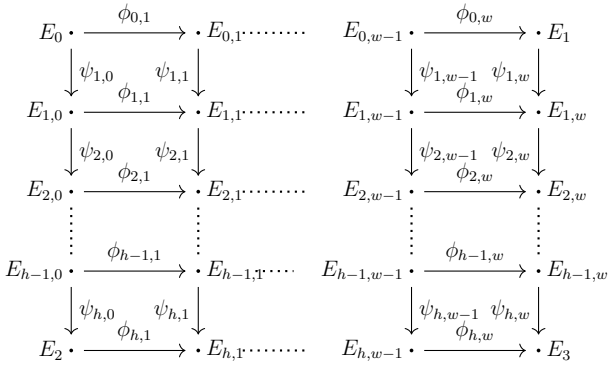

Figure 6.2: An SIDH ladder.

$E_0/K_{1,0} =: E_{1,0}$, generating a random kernel $K_{2,0}$ on $E_{1,0}$ such that $K_{2,0} \cap \ker \widehat{\psi_{1,0}} = \{0\}$ to prevent backtracking, and repeating the process h times to obtain a chain of h isogenies $\psi_{i,0} : E_{i-1,0} \to E_{i,0}$. The curve E_2 is the codomain of the last isogeny $\psi_{h,0}$, i.e., $E_2 = E_{h,0}$.

If the width w of the ladder is one, the prover can now recursively push the kernel G of the isogeny $\phi = \phi_{0,1}$ through the isogenies $\psi_{i,0}$ to obtain its image G_i on each curve $E_{i,0}$. Each horizontal isogeny $\phi_{0,i}$ has kernel G_i, and the prover can compute the kernel of the right-side vertical isogeny $\psi'_{i,0}$ as the image of the kernel of $\psi_{i,0}$ under the isogeny $\phi_{i-1,1}$. Since each square composed of $(E_{i,0}, E_{i+1,0}, E'_{i,0}, E'_{i+1,0})$ is a commutative diagram, so is the larger square (E_0, E_1, E_2, E_3). In the general case where $w > 1$, the prover can use a similar approach for the horizontal isogeny ϕ as used for the vertical isogeny ψ: The isogeny ϕ can be written as the composition of w isogenies $\phi_{0,w} \circ \ldots \circ \phi_{0,1}$ of degree 2^a and their kernels can be mapped through the vertical isogenies. In other words, the prover can glue horizontally w compatible ladders, one for each factor $\phi_{0,i}$ of ϕ. The right descending isogenies of each ladder are used as the left descending isogenies of the next one. This allows the prover to compute $w \times h$ SIDH squares in such a way that the curves (E_0, E_1, E_2, E_3) and the isogenies between them form a commutative diagram. This is illustrated in Fig. 6.2. For the challenges $\mathsf{chall} = \pm 1$, the prover reveals the isogenies $\psi_{i,0}$ of the leftmost squares, or the isogenies $\psi_{i,w}$ of the rightmost squares. For the challenge $\mathsf{chall} = 0$, the prover responds with the isogenies $\phi_{h,i}$ of the bottom squares.

Verification consists of evaluating (depending on the challenge) either w or h isogenies

of degree 2^a or 3^b, which can be done efficiently. Generating the proof is slower, as the prover needs to fill in all the $w \times h$ SIDH squares that make up the ladder. The proving complexity is thus quadratic in w and h, while the verification complexity is linear in w and h. However, the complexity of computing an SIDH square with degrees 2^a or 3^b is only quasilinear in a and b using sparse strategies [DJP14]; thus, maximizing the size of SIDH squares improves performance, which explains why SIDH primes are the most efficient scenario for this proof. If the degree of the isogenies and the size of the underlying field are kept constant, in the SIDH setting we have that $2^a 3^b \mid (p+1)$ for large values of a and b (in the order of several hundreds), and thus w and h can be small. For a generic prime, the prover might need to set $a = b = 1$ and work with large values of w and h, incurring a quadratic cost, besides possibly having to compute points over an extension field of degree bounded by a small constant.

Remark 7. *Above, we assumed that the degree of the witness ϕ was $d = (2^a)^w$. As mentioned before, this can be generalized to any witness ϕ of smooth degree $d = d_1 \dots d_w$ as far as the d_i-torsion groups are accessible (ideally, one should have $E_0[d_i] \subseteq E_0(\mathbb{F}_{p^2})$). In this case, one factors ϕ as $\phi = \phi_{0,w} \circ \dots \circ \phi_{0,1}$ where each isogeny $\phi_{0,i}$ has degree d_i, and constructs compatible ladders for each $\phi_{0,i}$.*

6.5 — Distributed SECUER Setup and its Security

In this section, we formally describe the distributed SECUER setup protocol and prove its security under a security definition using the simplified universal composability (SUC) framework due to Canetti, Cohen, and Lindell [CCL15] in the real/ideal world paradigm. Our security definitions consider a *dishonest majority* corruption model, wherein the adversary can corrupt up to $t-1$ of the t participating parties in the distributed SECUER setup protocol. The protocol uses a non-interactive version of the Σ-protocol described in Section 6.4. We begin by formally describing this non-interactive zero-knowledge (NIZK) PoK protocol.

$\mathsf{P}_{\mathsf{NIZK}}(E_0, E_1, \phi, n, N)$:

1: For each $i \in [N]$, sample $(\mathsf{com}_{2,i}, \mathsf{com}_{3,i}) \leftarrow \mathsf{P}_1(E_0, E_1, \phi, n)$.
2: Set $(\mathsf{chall}_1, \ldots, \mathsf{chall}_N) = H\big((\mathsf{com}_{2,1}, \mathsf{com}_{3,1}), \ldots, (\mathsf{com}_{2,N}, \mathsf{com}_{3,N})\big)$.
3: For each $i \in [N]$, set $\mathsf{resp}_i = \mathsf{P}_2(\mathsf{chall}_i)$.
4: **return** $\Pi = \big(\{(\mathsf{com}_{2,i}, \mathsf{com}_{3,i}, \mathsf{resp}_i)\}_{i\in[N]}\big)$.

$\mathsf{V}_{\mathsf{NIZK}}(E_0, E_1, \Pi, N)$:

1: Parse Π as $\big(\{(\mathsf{com}_{2,i}, \mathsf{com}_{3,i}, \mathsf{resp}_i)\}_{i\in[N]}\big)$.
2: Compute $(\mathsf{chall}_1, \ldots, \mathsf{chall}_N) = H\big((\mathsf{com}_{2,1}, \mathsf{com}_{3,1}), \ldots, (\mathsf{com}_{2,N}, \mathsf{com}_{3,N})\big)$.
3: For each $i \in [N]$, compute $b_i = \mathsf{V}(E_0, E_1, (\mathsf{com}_{2,i}, \mathsf{com}_{3,i}), \mathsf{chall}_i, \mathsf{resp}_i)$.
4: Output $b = \wedge_{i\in[N]} b_i$.

Figure 6.3: The NIZK.

6.5.1 – The NIZK protocol

We transform the Σ-protocol of Section 6.4 into a NIZK using the standard Fiat-Shamir heuristic [FS87] for transforming interactive PoK protocols into NIZK proofs, albeit with the difference that soundness and zero-knowledge hold for slightly different languages.

The NIZK construction. Let E_0, E_1 be supersingular curves defined over a finite field $\mathbb{F}_{p^2}$, let $\phi : E_0 \to E_1$ be a separable isogeny of smooth degree d and let $\mathsf{C}(m; r)$ be a statistically hiding and computationally binding commitment scheme. Additionally, let $\Sigma = (\mathsf{P}_1, \mathsf{P}_2, \mathsf{V})$ be the interactive PoK protocol described in Section 6.4, let $\lambda \in \mathbb{N}$ be the security parameter, let ℓ be a small prime not dividing dp, let $n = n(p, d, \ell, \lambda)$, and let $N = \mathsf{poly}(\lambda)$ be a fixed polynomial. Finally, let $H : \{0,1\}^* \to \{-1, 0, 1\}^N$ be a random oracle. The NIZK proof system consists of a pair of algorithms $\mathsf{NIZK} = (\mathsf{P}_{\mathsf{NIZK}}, \mathsf{V}_{\mathsf{NIZK}})$ as described in Fig. 6.3. The prover algorithm $\mathsf{P}_{\mathsf{NIZK}}$ is randomized and produces a proof Π. The verifier algorithm $\mathsf{V}_{\mathsf{NIZK}}$ is deterministic; it receives the proof Π and outputs a bit $b \in \{0, 1\}$ indicating whether or not the proof is accepted.

Correctness, Extractability and ZK. Correctness follows immediately from the correctness of the underlying Σ-protocol. We state and prove the following propositions for extractability and ZK.

Proposition 5. *Assuming that* $\Sigma = (\mathsf{P}_1, \mathsf{P}_2, \mathsf{V})$ *satisfies* 3*-special soundness with respect to the relation* $\mathcal{R}^\star$ *(described in Proposition 3) and that* H *is a random oracle, the NIZK* $\mathsf{NIZK} = (\mathsf{P}_{\mathsf{NIZK}}, \mathsf{V}_{\mathsf{NIZK}})$ *satisfies extractability (and hence soundness) with respect to the relation* $\mathcal{R}^\star$*.*

Proof. We provide an informal proof overview. We begin by noting that Σ is a public-coin protocol, and that there exists a probabilistic polynomial-time algorithm that extracts a witness from 3 accepting transcripts corresponding to N parallel executions of Σ w.r.t. the same statement. Consequently, we can invoke the generalized forking lemma of [BCCGP16] to argue the existence of a probabilistic polynomial-time witness-extraction algorithm for NIZK. This completes the proof of extractability (and hence, soundness) for NIZK.

Proposition 6. *Assuming that* $\Sigma = (\mathsf{P}_1, \mathsf{P}_2, \mathsf{V})$ *is statistically SHVZK for the relation* R_d *(described in Proposition 4) and that* H *is a random oracle, the NIZK* $\mathsf{NIZK} = (\mathsf{P}_{\mathsf{NIZK}}, \mathsf{V}_{\mathsf{NIZK}})$ *is statistically ZK for the relation* R_d*.*

Proof. We again provide an informal proof overview. Let Sim_Σ be a ZK simulator that simulates an accepting transcript for the underlying Σ-protocol (as described in the proof of ZK for Σ). We construct a ZK simulator $\mathsf{Sim}_{\mathsf{NIZK}}$ that simulates an accepting proof as follows:

1. $\mathsf{Sim}_{\mathsf{NIZK}}$ simulates the random oracle H as follows: it maintains a local table consisting of tuples of the form $(x, y) \in \{0,1\}^* \times \{-1, 0, 1\}^N$. On receiving a query $x \in \{0,1\}^*$ from the adversary $\mathcal{A}$, it looks up this table to check if an entry of the from (x, y) exists. If yes, it responds with y. Otherwise, it responds with a uniformly sampled $y \leftarrow \{-1, 0, 1\}^N$, and programs the random oracle as $H(x) := y$ by adding the entry (x, y) to the table.

2. For each $i \in [N]$, $\mathsf{Sim}_{\mathsf{NIZK}}$ internally invokes the simulator Sim_Σ for the underlying

Σ-protocol to obtain the i-th accepting transcript of the form

$$((\mathsf{com}_{2,i}, \mathsf{com}_{3,i}), \mathsf{chall}_i, \mathsf{resp}_i)\,.$$

3. At this point, $\mathsf{Sim}_{\mathsf{NIZK}}$ aborts if the adversary $\mathcal{A}$ has already issued a random oracle query on the input $x = \big((\mathsf{com}_{2,1}, \mathsf{com}_{3,1}), \ldots, (\mathsf{com}_{2,N}, \mathsf{com}_{3,N})\big)$.

4. Otherwise, $\mathsf{Sim}_{\mathsf{NIZK}}$ programs the random oracle as

$$H\big((\mathsf{com}_{2,1}, \mathsf{com}_{3,1}), \ldots, (\mathsf{com}_{2,N}, \mathsf{com}_{3,N})\big) := (\mathsf{chall}_1, \ldots, \mathsf{chall}_N),$$

and outputs the simulated proof as $\Pi = \big(\{(\mathsf{com}_{2,i}, \mathsf{com}_{3,i}, \mathsf{resp}_i)\}_{i\in[N]}\big)$.

We note that $\mathsf{Sim}_{\mathsf{NIZK}}$ runs in polynomial time as long as Sim_{Σ} runs in polynomial time. Additionally, if $\mathsf{Sim}_{\mathsf{NIZK}}$ does not abort, it outputs a simulated proof that is distributed in a statistically indistinguishable manner from the distribution of a real proof, assuming that Sim_{Σ} outputs a simulated accepting transcript with distribution statistically indistinguishable from a real accepting transcript for Σ. Finally, $\mathsf{Sim}_{\mathsf{NIZK}}$ aborts with only negligible probability, since the adversary $\mathcal{A}$ guesses $((\mathsf{com}_{2,i}, \mathsf{com}_{3,i}), \mathsf{chall}_i, \mathsf{resp}_i)$ for each $i \in [n]$ with at most negligible probability. This completes the proof of statistical ZK for NIZK.

6.5.2 – Our distributed SECUER setup protocol

We now move to the distributed SECUER setup protocol. Let $P_1, \ldots, P_t$ be a set of t participating parties and let E_0 be some fixed starting curve. In a nutshell, the idea is to have the parties act sequentially: each P_i at its own turn performs a secret random walk $E_{i-1} \to E_i$ and broadcasts E_i and a NIZK PoK of the secret walk. We claim that, as long as one party is honest, the final curve E_t is a SECUER.

To get any security guarantee, we need to carefully set the parameters of the random walk $E_{i-1} \to E_i$. The natural choice is to fix some small prime q, not dividing ℓp, and to take a random walk long enough that the distribution of E_i is negligibly far from the

stationary distribution on the q-isogeny graph $G(p, 1, q)$. For example we may set $q = 2$ and $\ell = 3$, then Theorem 8 provides a precise bound to set the length $\delta = n(p, 1, q, \lambda)$ of the q-walk as a function of the security parameter, and ultimately the parameter $n(p, q^\delta, \ell, \lambda)$ of the PoK.

Remark 8. *For increased efficiency, we may choose to perform shorter q-walks $E_{i-1} \to E_i$ of length $\log_q(p)$. This length approximates the diameter of the supersingular q-isogeny graph; hence, it ensures that the secret isogeny can reach almost any curve in the graph.*

Under mild assumptions, this choice would still yield a secure protocol, but it would also make the security proof somewhat more involved. For this reason, we shall stick here to the more conservative choice of walking long enough to ensure nearly stationary distribution of E_i.

We formally describe the protocol (referred to as Γ_{SECUER} henceforth). Assume that E_0 is known to all the parties at the start. Let $\mathsf{NIZK} = (\mathsf{P}_{\mathsf{NIZK}}, \mathsf{V}_{\mathsf{NIZK}})$ be the non-interactive proof as described above. The protocol Γ_{SECUER} proceeds in t rounds while only using broadcast channels of communication, where round-i for each $i \in [t]$ is as follows:

- Party P_i performs a q-isogeny walk starting at curve E_{i-1} and ending at curve E_i (where E_{i-1} and E_i are both supersingular curves defined over $\mathbb{F}_{p^2}$), such that party P_i knows a separable isogeny $\phi_i : E_{i-1} \to E_i$ of degree q^δ, where $\delta = n(p, 1, q, \lambda)$.
- Party P_i generates $\Pi_i \leftarrow \mathsf{P}_{\mathsf{NIZK}}(E_{i-1}, E_i, \phi_i, n, N)$, where $n = n(p, q^\delta, \ell, \lambda)$, and broadcasts (E_i, Π_i) to all other parties.
- Each party P_j for $j \in [t] \setminus \{i\}$ verifies the NIZK proof Π_i by computing $b_i = \mathsf{V}_{\mathsf{NIZK}}(E_{i-1}, E_i, \Pi_i, N)$. If $b_i = 0$ (i.e., the proof is invalid), P_j aborts.

At the end of round-t, all parties output E_t to be the final output curve.

Correctness. Correctness of Γ_{SECUER} follows immediately from the correctness guarantees of the NIZK.

6.5.3 – Proof of security for Γ_{SECUER}

We now present the proof of security for Γ_{SECUER} using the simplified universal composability (SUC) framework [CCL15] in the real/ideal world paradigm. We consider a *dishonest majority* corruption model, wherein the adversary can corrupt up to $(t-1)$ of the t participating parties.

The ideal functionality. Intuitively, the ideal functionality for distributed SECUER setup should simply take as input the initial curve E_0 and output a SECUER E_t. It is however not obvious how to model the property of being a SECUER in the plain SUC model: a game based definition, stating that an adversary who can compute $\mathrm{End}(E_t)$ can be used to break some other assumption, appears to be more appropriate.

Thus, we prove security in two steps. First, we prove that Γ_{SECUER} securely emulates a *less-than-ideal* functionality $\mathcal{F}^*_{\text{SECUER}}$ (described in Fig. 6.4) that enforces that: (a) for each $i \in [t]$, if a corrupt party P_i outputs a curve E_i, it must know a valid isogeny $\phi_i : E_{i-1} \to E_i$, and (b) for each $i \in [t]$, if an honest party P_i outputs a curve E_i, then the corresponding isogeny $\phi_i : E_{i-1} \to E_i$ is hidden from the adversary. This step relies on the extractability and ZK properties of the NIZK protocol described above. Next, we prove that, assuming the hardness of the endomorphism ring problem in the $\mathcal{F}^*_{\text{SECUER}}$-hybrid model, the output curve E_t is a SECUER, i.e. that the (malicious) adversary cannot compute $\mathrm{End}(E_t)$.

Theorem 9. *Assuming that* $\mathsf{NIZK} = (\mathsf{P}_{\mathsf{NIZK}}, \mathsf{V}_{\mathsf{NIZK}})$ *satisfies extractability and zero-knowledge, and assuming the hardness of the endomorphism ring problem (Definition 6) and GRH, the output* E_t *of the protocol* Γ_{SECUER} *is a* SECUER *if at least one party* P_{i^*} *for some* $i^* \in [t]$ *is honest.*

Secure emulation of $\mathcal{F}^*_{\text{SECUER}}$. We now prove that Γ_{SECUER} securely emulates the less-than-ideal functionality $\mathcal{F}^*_{\text{SECUER}}$. Our proof is in the real/ideal world paradigm defined formally as follows.

$\mathcal{F}^*_{\text{SECUER}}(E_0, i \in [t])$

- Let $\mathcal{H}_i \subseteq [i-1]$ be the set of honest parties, and let $\mathcal{C}_i \subseteq [i-1]$ be the set of corrupt parties among the first $(i-1)$ parties $P_1, \ldots, P_{(i-1)}$.
- For each $j \in \mathcal{H}_i$, $\mathcal{F}^*_{\text{SECUER}}$ receives as input from P_j a tuple of the form (E_j, ϕ_j).
- For each $j' \in \mathcal{C}_i$, $\mathcal{F}^*_{\text{SECUER}}$ receives as input from the simulator Sim a tuple of the form $(E_{j'}, \phi_{j'})$.
- If for any $j \in [i-1]$, ϕ_j is not an isogeny from the curve E_{j-1} to the curve E_j, $\mathcal{F}^*_{\text{SECUER}}$ outputs $\perp$ and aborts.
- Otherwise, $\mathcal{F}^*_{\text{SECUER}}$ takes a random walk starting from the $(i-1)$-th curve E_{i-1} and ending in a curve E_i such that $\mathcal{F}^*_{\text{SECUER}}$ knows $\phi_i : E_{i-1} \to E_i$, where ϕ_i is a separable isogeny of degree d.
- Finally, $\mathcal{F}^*_{\text{SECUER}}$ outputs (E_i, ϕ_i) to the party P_i, and outputs E_i to the simulator Sim and to all parties P_j for $j \neq i$.

Figure 6.4: The Ideal functionality $\mathcal{F}^*_{\text{SECUER}}$

The real world. The following entities engage in the real protocol Γ_{SECUER}: (i) a set $\mathcal{H} \subseteq [t]$ of honest parties, (ii) a real-world adversary $\mathcal{A}$ controlling a set $\mathcal{C} \subset [t]$ of corrupt parties, and (iii) the environment $\mathcal{E}$ that provides E_0 to each party, interacts with the real-world adversary $\mathcal{A}$, receives the final output curve E_t from the honest parties, and eventually outputs a bit $b \in \{0, 1\}$.

The ideal world. The following entities interact with the functionality $\mathcal{F}^*_{\text{SECUER}}$: (i) A set $\mathcal{H} \subseteq [t]$ of honest parties, where for each $i \in \mathcal{H}$, party P_i directly forwards its secret isogeny to $\mathcal{F}^*_{\text{SECUER}}$, (ii) an ideal-world simulator Sim that sends inputs to $\mathcal{F}^*_{\text{SECUER}}$ on behalf of a set $\mathcal{C} \subset [t]$ of corrupt parties, and (iii) the environment $\mathcal{E}$ that provides each party with the starting curve E_0, interacts with the simulator Sim, receives the final output curve E_t from the functionality, and eventually outputs a bit $b \in \{0, 1\}$.

For any t-party SECUER setup protocol Γ_{SECUER}, any adversary $\mathcal{A}$, any simulator Sim, and any environment $\mathcal{E}$, we define the following random variables:

- $\mathsf{real}_{\Gamma_{\text{SECUER}}, \mathcal{A}, \mathcal{E}}$: denotes the output of the environment $\mathcal{E}$ after interacting with the adversary $\mathcal{A}$ during a real-world execution of Γ_{SECUER}.

- $\mathsf{ideal}_{\mathcal{F}^*_{\text{SECUER}}, \mathsf{Sim}, \mathcal{E}}$: denotes the output of the environment $\mathcal{E}$ after interacting with the

simulator Sim in the ideal world.

Theorem 10. *Assuming that* $\mathsf{NIZK} = (\mathsf{P}_{\mathsf{NIZK}}, \mathsf{V}_{\mathsf{NIZK}})$ *satisfies extractability and zero-knowledge, for any security parameter* $\lambda \in \mathbb{N}$ *and any probabilistic polynomial time (PPT) adversary* $\mathcal{A}$*, there exists a PPT simulator* Sim *such that, for any PPT environment* $\mathcal{E}$*, we have*

$$\left|\Pr\left[\mathsf{real}_{\Gamma_{\textsc{Secuer}},\mathcal{A},\mathcal{E}} = 1\right] - \Pr\left[\mathsf{ideal}_{\mathcal{F}^*_{\textsc{Secuer}},\mathsf{Sim},\mathcal{E}} = 1\right]\right| \leq \mathsf{negl}(\lambda).$$

Proof. We prove this theorem by constructing a PPT simulator Sim that simulates the view of the environment $\mathcal{E}$ in the ideal world. Details are given in Appendix 6.A. □

Analyzing E_t in $\mathcal{F}^*_{\textsc{Secuer}}$-hybrid model. Based on the above secure emulation guarantee, we now analyze the output E_t of $\Gamma_{\textsc{Secuer}}$ in the $\mathcal{F}^*_{\textsc{Secuer}}$-hybrid model. Concretely, we state and prove the following theorem.

Theorem 11. *Assuming the hardness of the endomorphism ring problem and GRH, the output* E_t *of* $\mathcal{F}^*_{\textsc{Secuer}}(E_0, t)$ *is a* Secuer *if at least one party is honest.*

To prove this theorem, we first prove the following lemma.

Lemma 19. *Assuming the hardness of the endomorphism ring problem, the output* E_i *of* $\mathcal{F}^*_{\textsc{Secuer}}(E_0, i)$ *for* $i \in [t]$ *is a* Secuer *whenever* P_i *is honest.*

Proof. Suppose that there exists an adversary $\mathcal{A}$ corrupting a dishonest majority of the parties that efficiently computes the endomorphism ring of E_i with non-negligible probability. Also assume that $\mathcal{A}$ corrupts all of $P_1, \ldots, P_{i-1}$. We can use $\mathcal{A}$ to construct an algorithm $\mathcal{B}$ that solves the endomorphism ring problem. The algorithm $\mathcal{B}$ receives as input a uniformly random curve $E^*/\mathbb{F}_{p^2}$, internally runs the adversary $\mathcal{A}$ to emulate the outputs of the corrupt parties $P_1, \ldots, P_{i-1}$, and finally feeds $\mathcal{A}$ with $E_i := E^*$. The view of the adversary $\mathcal{A}$ is properly simulated by $\mathcal{B}$, since E_i output by $\mathcal{F}^*_{\textsc{Secuer}}$ and E^* provisioned by $\mathcal{B}$ are statistically indistinguishable (here we use Theorem 8, which crucially follows from the honest party taking a q-walk of length $n(p, 1, q, \lambda)$). Finally, $\mathcal{B}$ uses $\mathcal{A}$ to recover

the endomorphism ring of E^* with non-negligible probability. This concludes the proof of Lemma 19. □

We now prove Theorem 11. We break the proof into two cases: (i) when P_t is honest, and (ii) when P_t is corrupt. The proof for case (i) is immediate from Lemma 19. Hence, we focus on case (ii). Let $\mathcal{H} \subseteq [t]$ be the set of honest parties, and let $i^* = \max(\{i : P_i \in \mathcal{H}\})$. By Lemma 19, E_{i^*} must be a SECUER. Now, suppose that E_t is not a SECUER, i.e., there exists an adversary $\mathcal{A}$ corrupting dishonest majority of the parties that efficiently computes the endomorphism ring of E_t with non-negligible probability. Since all of $P_{i^*+1}, \ldots, P_t$ are corrupt, $\mathcal{A}$ knows a walk from E_{i^*} to E_t in the ℓ-isogeny graph. However, since E_t is not a SECUER, $\mathcal{A}$ can use the reduction [Wes22b] (assuming GRH) to recover $\mathrm{End}(E_{i^*})$, thereby violating Lemma 19. This completes the proof of Theorem 11. Finally, the proof of Theorem 9 follows immediately from the proofs of Theorem 10 and Theorem 11, which completes the proof of security for our distributed SECUER setup protocol Γ_{SECUER}.

6.6 — Implementation and Results

In this section, we report on our proof-of-concept implementation of our proof of knowledge (Section 6.4), including a discussion of proof sizes and running times. Moreover, we lay out concretely how one may deploy the trusted setup protocol from Section 6.5 in the real world.

Parameter selection. The base-field primes p in our proof-of-knowledge implementation are taken from the four SIKE parameter sets `p434`, `p503`, `p610`, and `p751`. As discussed in Section 6.4.2, our proof of knowledge achieves its optimal efficiency for SIDH-style primes. Moreover, those primes have been featured extensively in the literature, and thus appear to be the obvious choice to demonstrate our proof of knowledge. That said, we stress once more that our techniques are generic and can be applied in any choice of characteristic.

Table 6.1: Parameters and corresponding secret/proof size for each of the four SIKE finite fields.

		Degree		SIDH Squares		Size (kB)	
$\log(p)$	Reps	2-isog.	3-isog.	Columns	Rows	Secret	Proof
434	219	705	890	4	7	0.99	191.19
503	219	774	977	4	7	1.13	215.75
610	329	1010	1275	4	7	1.39	404.32
751	438	1280	1616	4	7	1.69	662.63

We use the degree $q = 2$ for the random walks $E_i \to E_{i-1}$, and $\ell = 3$ for the random walks of the Σ-protocol of Fig. 6.1. Like Section 6.5, we set $\delta = n(p, 1, 2, \lambda)$ for the length of the 2-walks, and $n = n(p, 2^\delta, 3, \lambda)$ for the 3-walks. Lastly, the Σ-protocol needs to be repeated several times to achieve a negligible soundness error. Since one repetition has soundness error $2/3$, the protocol needs to be repeated $-\lambda/\log(2/3)$ times to achieve $2^{-\lambda}$ soundness error. We target the same security levels as the corresponding SIKE parameter sets, i.e., $\lambda = 128$ for `p434` and `p503`, $\lambda = 192$ for `p610`, and $\lambda = 256$ for `p751`. The resulting conservative parameters are summarized in Table 6.1.

Implementation. We developed an optimized implementation[6] of our proof of knowledge (Section 6.4.1) for the trusted-setup application (Section 6.5) based on version 3.5.1 of Microsoft's SIDH library[7]. Our implementation inherits and benefits from all lower-level optimizations contained in that library, and it supports a wide range of platforms with optimized code for a variety of Intel and ARM processors. Compiling our software produces two command-line tools `prove` and `verify`, which use a simple ASCII-based interface to communicate the data contributed to the trusted setup and its associated proof of isogeny knowledge.

The implementation closely follows the strategy outlined in Section 6.4.2. This includes the choices $d = (2^a)^w$ and $\ell^n = (3^b)^h$; thus, both the witness and the commitment isogenies are uniformly random cyclic isogenies of degree d and ℓ^n respectively. To

[6]The source code is available at https://github.com/trusted-isogenies/SECUER-pok.
[7]https://github.com/microsoft/PQCrypto-SIDH

reduce latency, we additionally exploit parallelism: Recall that the proof of knowledge is repeated many times to achieve a low soundness error; indeed most of the computations are independent between those repetitions and can thus easily be performed at the same time on a multi-core system. This is confirmed by experimental results, where our implementation is observed to parallelize almost perfectly when run on an eight-core processor.

Sampling purely random large-degree isogenies with code from SIDH comes with two caveats: First, the sampling of "small" squares must avoid backtracking between the individual squares being glued to ensure that the composition is cyclic in the end; in both cases this is done by keeping track of the kernel of the dual of the last prime-degree step of the previous square and avoiding points lying above this "forbidden" kernel when choosing the next square. Besides that, the specific isogeny formulas used in SIDH fail for the 2-torsion point $(0, 0)$, which can be resolved by changing to a different Montgomery model each time this kernel point is encountered. For curves revealed in the proof, the choice of Montgomery model should be randomized to avoid leakage. Similarly, the kernel generators of the horizontal isogeny ϕ' also need to be randomized, as Lemma 18 only distinguishes cyclic subgroups and revealing specific generators may leak.

Our software sacrifices some performance for simplicity, which aids auditability and hence helps increase trust in the results of a trusted-setup ceremony. Some unused optimizations: Two-isogenies are faster to compute than three-isogenies, and since the SIDH ladder is taller than wider, swapping the role of two- and three-isogenies in the trusted-setup application could somewhat improve the resulting performance. For simplicity, our implementation also only uses full SIDH squares, and thus all isogeny degrees are rounded up to the closest multiple of an SIDH square; shortening the sides of some of the squares can save time. We also did not apply all optimizations to reduce the proof size. This includes applying SIDH-style compression techniques [CJLNR+17] to the points contained in the proof, cutting their size approximately in half. Moreover, applying a slight bias when sampling the challenges chall_i means smaller responses can

Table 6.2: Benchmarks for instance generation, proving, and verification of our proof of isogeny knowledge for each of the four SIKE finite fields.

	Single-core Time (s)			Eight-core Time (s)		
$\log(p)$	Instance	Prove	Verify	Instance	Prove	Verify
434	0.01	18.15	1.93	0.01	2.96	0.32
503	0.01	25.70	2.71	0.01	4.17	0.44
610	0.02	74.82	7.69	0.02	12.12	1.24
751	0.04	162.47	17.01	0.04	26.07	2.89

appear more often, at the expense of requiring slightly more repetitions; we investigated this tradeoff and determined that the potential improvement is essentially void.

Results. We benchmarked the three algorithms (instance generation, proving, and verification) that make up the zero-knowledge proof of knowledge. We run our tests on an ARM Apple M1 Pro with eight cores, and we averaged the running times of 100 iterations for the parallel implementation and the running times of 50 iterations of the single-core version. The resulting timings are shown in Table 6.2. They demonstrate that the algorithm is highly practical and can realistically be used within a trusted setup protocol: Generating proofs of knowledge for all four base fields takes less than five core-minutes on a modern CPU. Note that these algorithms need to be run only once per contributor.

Real-world deployment. We briefly discuss how we intend to deploy the trusted setup protocol proposed in Section 6.5. The goals of such a deployment include include a transparent setup that allows participants to trust the process, a low bar of entry to participate in the protocol, and a secure system that can withstand Sybil and Denial-of-Service (DoS) attacks.

Firstly, we will release a set of tools that participants can download and run to generate a valid addition to the trusted setup, and for ceremony orchestrators to validate protocol submissions on the server-side. To increase user trust, we also provide higher-level

versions (e.g., in SageMath) of some components. Moreover, the proof format is made public, so that any participant can—if they choose to—re-implement the proof algorithm and generate a compatible proof.

Then, we propose leveraging the existing infrastructure of git and GitHub to host our distributed protocol. Thus, participants can generate a random walk from the latest curve to a new curve, generate a proof of knowledge of their secret isogeny walk, and submit the new curve and the proof of knowledge to the server as a pull request (PR). The server is a separate git repository and execution environment maintaining the series of curves and the proofs, with checks that are run automatically against submissions from participants. The repository automation verifies that the submitted proof of knowledge of the isogeny between the current 'tip' curve and the new proposed curve is valid, and that all the prior proofs are valid in the context of the proposed changes. The server also verifies that the number of proofs / curves is strictly increasing (that the participant is not replacing any prior hops with their own). If the checks pass, the PR is rebased on top of the main branch, adding the new proof of knowledge of the latest hop, and updating the current latest curve to the new one. New participants in the protocol will generate isogeny walks starting from the new tip curve.

If any existing or 'in-flight' isogeny walk submissions exist, they will have been rendered moot after the tip curve has changed, and would no longer be accepted as they no longer extend from the new tip curve. In our tests, walk generation, proving, and verification per-submission is quite fast, but if the protocol runs long or has many participants, many proofs will be verified and the chances of a valid in-flight submission failing to complete full chain verification before the tip curve is updated again increase. We can parallelize our verification of the multiple proofs to lower these chances, and do a quick validation abort if any proof or any checks of the validity of the chaining of curves fails.

The configuration for the continuous integration (CI) checks is maintained in a separate repository to prevent modification from protocol participants. Hosting the protocol on

GitHub raises the bar to Sybil attacks, as it requires all participants to have a GitHub account with a verified email address. Using our tool requires generation of a GitHub personal access token to authenticate when generating the submission, which further complicates automation / collusion of adversarial participants.

The end result of the protocol is a public git repository whose final commit contains a series of curves and valid proofs of knowledge of isogenies between them, the last of which is the final SECUER, a curve with unknown endomorphism ring, in a parsable hex encoding. Anyone can pull down this artifact and verify the series of curves and proofs independently if they wish.

6.A — Proofs of the theorems

Proof of Theorem 6

Proof. First we show 1. Let L_{ij} be the set of degree ℓ isogenies from (E_i, C_i) to (E_j, C_j). If f is in L_{ij}, then the dual isogeny $\hat{f}$ is a degree ℓ isogeny from (E_j, C_j) to $(E_i, \ell C_i)$. Since ℓ is coprime with d, ℓC_i is equal to C_i, and the duality gives a bijection between L_{ij} and L_{ji}. The entry a_{ij} of A is the cardinality of the quotient $L_{ij}/\operatorname{Aut}(E_j, C_j)$, hence $|\operatorname{Aut}(E_i, C_i)|a_{ji} = |\operatorname{Aut}(E_j, C_j)|a_{ij}$. Dividing this equality by two we get $w_i a_{ji} = w_j a_{ij}$. The claim now follows from the definition of Q.

We now prove 2. We have

$$\begin{aligned}\mathcal{E}A &= \sum_{i=1}^{n} \frac{1}{w_i}(E_i, C_i)A = \sum_{i,j=1}^{n} \frac{1}{w_i}\frac{|L_{ij}|}{w_i}(E_j, C_j) = \sum_{j=1}^{n} \frac{1}{w_j}(E_j, C_j)\sum_{i=1}^{n} \frac{|L_{ji}|}{w_i} \\ &= \sum_{j=1}^{n} \frac{1}{w_j}(E_j, C_j)(\ell+1) = (\ell+1)\mathcal{E}\,.\end{aligned}$$

To see part 3, observe that the out-degree of each vertex of G is $\ell + 1$, hence the sum of the elements of the rows of A is $\ell + 1$, so the claim.

We now prove 4. Let $\langle \cdot, \cdot \rangle$ be the Hermitian product on $\mathbb{C}^V$ such that the basis (E_i, C_i) is unitary. Let $w = \sum w_i^{-1/2}(E_i, C_i)$ and, for each $v = \sum v_i(E_i, C_i)$, let $\tilde{v} = \sum w_i^{1/2}|v_i|(E_i, C_i)$. Then, the Cauchy-Schwarz inequality gives

$$|v|_{L^1} = \langle \tilde{v}, w \rangle \leq \sqrt{\langle \tilde{v}, \tilde{v} \rangle}\sqrt{\langle w, w \rangle} = |v|_Q \sqrt{\langle w, w \rangle} = |v|_Q \sqrt{\sum \frac{1}{w_i}}$$

and moreover we get the equality when $\tilde{v} = w/|w|_{L^1}$. We now compute $K^{-1} = \sqrt{\sum \frac{1}{w_i}}$. Eichler's formula [Hus04, Section 13.5, Theorem 4.1] gives

$$\sum_{\substack{E/\overline{\mathbb{F}_p} \text{ supersingular,} \\ \text{up to } \overline{F_p}\text{-isomorphism}}} \frac{1}{|\operatorname{Aut}(E)|} = \frac{p-1}{24}\,.$$

We are going to show that, for H the group of upper triangular matrices

$$\sum_{i \text{ such that } E_i \simeq E} \frac{|\operatorname{Aut}(E)|}{|\operatorname{Aut}(E_i, C_i)|} = [\mathrm{GL}_2(\mathbb{Z}/d\mathbb{Z}) : H]\,. \tag{6.5}$$

Given this equation for granted, K can be computed by writing $d = \prod_q q^{e_q}$ and checking that $|\mathrm{GL}_2(\mathbb{Z}/d\mathbb{Z})| = \prod_q (q^{2e_q} - q^{2e_q-2})(q^{2e_q} - q^{2e_q-1})$ and $|H| = \prod_q q^{e_q}(q^{e_q} - q^{e_q-1})^2$.

Equation (6.5) is the equation of the orbits for a group action. Fix an elliptic curve E, let X be the set of order d cyclic subgroups of $E[d]$. This set has a natural transitive action by $\operatorname{Aut}(E[d]) \cong \mathrm{GL}_2(\mathbb{Z}/d\mathbb{Z})$, which gives a bijection $X \leftrightarrow \mathrm{GL}_2(\mathbb{Z}/d\mathbb{Z})/H$, so the right hand side of Equation (6.5) is the cardinality of X. Level d Borel structures on E are the orbits of the action of $\operatorname{Aut}(E)$ on X. The left hand side of Equation (6.5) is again the cardinality of X, obtained summing the cardinalities of each orbit.

Finally we prove 5. Let $\pi = \sum_{i=1}^n \pi_i(E_i, C_i)$ be a probability distribution and let $\lambda = \sum_{i=1}^n \frac{1}{w_i}$, so that $s = \sum_{i=1}^n \frac{1}{\lambda w_i}(E_i, C_i)$. Then, using $\sum \pi_i = 1$, we have

$$\begin{aligned} |\pi - s|_Q^2 &= \sum_{i=1}^n w_i \left(\pi_i - \frac{1}{\lambda w_i}\right)^2 = \sum_{i=1}^n \left(w_i \pi_i^2 - \frac{2\pi_i}{\lambda} + \frac{1}{\lambda^2 w_i}\right) \\ &= \sum_{i=1}^n w_i \pi_i^2 - \frac{2}{\lambda} + \frac{1}{\lambda} \leq \sum_{i=1}^n w_i \pi_i^2 \leq (\max w_i) \sum_{i=1}^n \pi_i = \max w_i\,. \end{aligned}$$

We conclude recalling that $w_i \leq 3$ for every i. Notice that for $\pi = (E_i, C_i)$ we get $|\pi - s|_Q^2 = w_i - 1/\lambda$, hence the above estimate is not too loose. □

Proof of Theorem 8

Proof. Denote by $A^{(k)}$ the matrix whose (i, j) entry is the number of non-backtracking walks from i to j. Since each edge has a unique dual, we get the same recurrence formula as in the non-oriented case, namely

$$A^{(1)} = A, \quad A^{(2)} = A^2 - (\ell + 1), \quad A^{(k+1)} = AA^{(k)} - \ell A^{(k-1)}\,.$$

Observe that the sum of all the entries in a fixed row of $A^{(k)}$ is $(\ell+1)\ell^{k-1}$. We denote by $P^{(k)}$ its normalization

$$P^{(k)} := \frac{1}{(\ell+1)\ell^{k-1}} A^{(k)} .$$

Hence, $P^{(k)}$ is a polynomial in A, see e.g. [ABLS07, Section 2]. Let us call this polynomial $\mu_k(x)$ (here, the use of the symbol μ_i is slightly different from the one of [ABLS07]). The matrix $P^{(k)}$ is diagonalizable, it has the same eigenvectors as A, and has eigenvalues $\mu_k(\ell+1) = 1$ and $\mu_k(\lambda_i)$, where λ_i is any eigenvalue of A different from $\ell+1$.

Combining the proof of [ABLS07, Lemma 2.3] and Theorem 5, we get

$$\mu_k(\lambda_i) = \frac{1}{\sqrt{(\ell+1)\ell^{k-1}}} \left(\sqrt{\frac{\ell}{\ell+1}} \frac{\sin((k+1)\theta)}{\sin(\theta)} - \frac{1}{\sqrt{(\ell+1)\ell}} \frac{\sin((k-1)\theta)}{\sin(\theta)} \right) \tag{6.6}$$

where $\cos(\theta) = \lambda_i/(2\sqrt{\ell})$. Recall that $|\sin(x+y)| \leq |\sin(x)| + |\sin(y)|$, hence $|\sin(m\theta)| \leq m|\sin(\theta)|$ and we can achieve the bound:

$$|\mu_k(\lambda_i)| \leq \frac{1}{\sqrt{(\ell+1)\ell^{k-1}}} \left(\sqrt{\frac{\ell}{\ell+1}}(k+1) + \frac{1}{\sqrt{(\ell+1)\ell}}(k-1) \right) = \frac{(\ell+1)(k+1)-2}{(\ell+1)\sqrt{\ell^k}} . \tag{6.7}$$

Now observe that $\pi^{(k)} = \pi P^{(k)}$, and hence $\pi^{(k)} - s = (\pi - s)\, P^{(k)}$. The difference of two probability distributions is orthogonal for the standard L^2 scalar product to the vector u from Theorem 6 item 3. Since $\mathcal{E}$ is not orthogonal to u, by Theorem 6 item 3 we conclude that $\pi - s$ is in the linear span of the eigenvectors of A corresponding to eigenvalues different from $\ell+1$. Since A is self-adjoint with respect to Q, using Equation (6.7) we have

$$|(\pi - s)P^{(k)}|_Q \leq \frac{(\ell+1)(k+1)-2}{(\ell+1)\sqrt{\ell^k}} |\pi - s|_Q \tag{6.8}$$

The definition of K and M from Theorem 6 tells that $K|\pi^{(k)} - s|_{L^1} \leq |\pi^{(k)} - s|_Q$, and $|\pi - s|_Q \leq M$. We obtain the result recalling that the total variation distance between two probability distributions is half of the L^1 distance, see e.g. [LP17, Proposition 4.2].

□

Remark 9 (Improvement of Theorem 8 and Lemma 18)**.** *Under the assumption that the eigenvalues of the adjacency matrix of G are strictly contained in the Hasse interval (so there are no eigenvalues equal to $\pm 2\sqrt{\ell}$), Theorem 8 can be improved: the linear factor $(k+1)$ can be replaced by a constant which does not depend on k. Indeed, as $\pm 2\sqrt{\ell}$ is not an eigenvalue, $\sin(\theta)$ in Equation 6.6 never vanishes. If we write $|\sin(\theta)| \geq \varepsilon$ for some $\varepsilon > 0$, we obtain*

$$|\mu_k(\lambda_i)| \leq \left(\varepsilon\sqrt{\ell^k}\right)^{-1}$$

which can be used in place of Equation 6.7. Observe that, even with this improvement, the bound will not be sharp, because in the bound of Equation 6.8 we consider only the eigenvalues with greatest modulus, but the other eigenvalues of A have smaller modulus.

This argument in turn improves Lemma 18, where the linear factor k can be replaced by a constant independent of k.

Proof of Lemma 18

Proof. We bound the statistical distance of each of $\mathcal{D}_1(k)$ and $\mathcal{D}_2(k)$ from the stationary distribution of $G(p, d, \ell)$, as determined in Theorem 6, then we conclude with the triangle inequality. For $\mathcal{D}_1(k)$, we can directly apply Theorem 8, but $\mathcal{D}_2(k)$ needs more care.

Let G_0 be the classical isogeny graph. This can be thought of as the graph with $d = 1$ Borel level structure. Let s_0 be the stationary distribution on G_0. Consider the projection map $P\colon G \to G_0$ which forgets the level structure. The push-forward distribution $P_*\mathcal{D}_2(k)$ is the distribution of the length k non-backtracking walks starting at E_0, so we can bound its total variation distance from s_0 using Theorem 8. For any probability distribution π on G_0 let us denote $\widetilde{\pi}$ the distribution on G obtained by first choosing E with distribution π and then choosing C uniformly inside the set of cyclic subgroups of order d. Notice that for each two subgroups C, C', the pair (E, C) defines the same vertex as (E, C') if and only if there exists an automorphism of E sending C to C'. This, together with the

fact that the set of C's for a single E has cardinality $[\mathrm{GL}_2(\mathbb{Z}/N\mathbb{Z}) : H]$, implies

$$\widetilde{\pi}((E,C)) = \frac{|\operatorname{Aut}(E)/\operatorname{Aut}(E,C)|}{[\mathrm{GL}_2(\mathbb{Z}/N\mathbb{Z}) : H]}\pi(E)\,,$$

where H is the subgroup of upper triangular matrices. The above formula, together with (6.5), implies that for every probability distribution π on G_0 and every subset A of G_0, one has $\widetilde{\pi}(P^{-1}(A)) = \pi(A)$. In turn, this means that for π_1, π_2 probability measures on G_0, we have $d_{TV}(\pi_1, \pi_2) = d_{TV}(\widetilde{\pi}_1, \widetilde{\pi}_2)$. One can then check by direct computation that $s = \widetilde{s_0}$. We conclude that $d_{TV}(\mathcal{D}_2(k), s) = d_{TV}(P_*\mathcal{D}_2(k), s_0)$, and the right hand side can be bound using Theorem 8. □

Proof of Theorem 10

Proof. We prove this theorem by constructing a PPT simulator Sim that simulates the view of the environment $\mathcal{E}$ in the ideal world. The simulator Sim receives E_0 from the environment $\mathcal{E}$, internally runs the real-world adversary $\mathcal{A}$ and the NIZK simulator $\mathsf{Sim}_{\mathsf{NIZK}}$, and proceeds in round-$i$ for $i \in [t]$ as described next. Note that we implicitly assume that Sim has rewinding access to the adversary $\mathcal{A}$ and programming access to the random oracle in the analysis below.

Case-1: Party P_i is corrupt. In this case, Sim internally runs the real-world adversary $\mathcal{A}$ to obtain the broadcast message (E_i, Π_i) corresponding to the corrupt party P_i. It then uses the extraction algorithm of NIZK to extract the corresponding witness ϕ_i. If extraction fails, Sim outputs $\perp$ and aborts. Otherwise, Sim stores (E_i, Π_i, ϕ_i) internally, and broadcasts (E_i, Π_i) as the message corresponding to the corrupt party P_i.

Case-2: Party P_i is honest. In this case, Sim invokes the ideal functionality to obtain E_i. Concretely, let $\mathcal{C}_i \subseteq [i-1]$ be the set of corrupt parties among the first $(i-1)$ parties $P_1, \ldots, P_{(i-1)}$. Sim invokes the ideal functionality $\mathcal{F}^*_{\text{SECUER}}(E_0, i)$ with the set $\{(E_{j'}, \phi_{j'})\}_{j' \in [C_i]}$. If $\mathcal{F}^*_{\text{SECUER}}$ outputs $\perp$, Sim outputs $\perp$ and aborts. Otherwise, Sim re-

ceives from $\mathcal{F}^*_{\text{SECUER}}$ the corresponding curve E_i. At this point, it invokes the simulator $\mathsf{Sim}_{\mathsf{NIZK}}$ of the NIZK protocol to obtain a simulated proof as $\overline{\Pi}_i \leftarrow \mathsf{Sim}_{\mathsf{NIZK}}(E_{i-1}, E_i, N)$, and broadcasts $(E_i, \overline{\Pi}_i)$ as the message corresponding to the honest party P_i.

Indistinguishability of views. We now prove that for the above construction of Sim, the view of $\mathcal{E}$ in the ideal world is indistinguishable from that in the real world. We prove this by a sequence of hybrids as described below (recall that $\mathcal{H} \subseteq [t]$ and $\mathcal{C} \subset [t]$ denote the set of honest and corrupt parties, respectively).

- **Hybrid-0:** In this hybrid, the distribution of messages broadcast by each party is identical to the real-world protocol γ_{SECUER}.
- **Hybrid-1:** In this hybrid, for each corrupt party P_j such that $j \in \mathcal{C}$, instead of verifying the NIZK proof Π_j using $\mathsf{V}_{\mathsf{NIZK}}$ (as in the real protocol), extract the witness ϕ_j using the the extraction algorithm of NIZK. If extraction fails, output $\perp$.
- **Hybrid-2:** In this hybrid, for each honest party P_i such that $i \in \mathcal{H}$, instead of generating the NIZK proof $\Pi_i \leftarrow \mathsf{P}_{\mathsf{NIZK}}(E_{i-1}, E_i, \phi_i, n, N)$ (as in the real protocol), generate a simulated proof as $\overline{\Pi}_i \leftarrow \mathsf{Sim}_{\mathsf{NIZK}}(E_{i-1}, E_i, N)$.
- **Hybrid-3:** In this hybrid, the distribution of messages broadcast by each party is identical to the ideal-world messages broadcast by Sim.

Lemma 20. *Assuming that* $\mathsf{NIZK} = (\mathsf{P}_{\mathsf{NIZK}}, \mathsf{V}_{\mathsf{NIZK}})$ *satisfies extractability, hybrid-0 and hybrid-1 are indistinguishable.*

Note that for $\mathcal{E}$ to distinguish between hybrid-0 and hybrid-1 with non-negligible probability, the adversary $\mathcal{A}$ must be able to produce with non-negligible probability a proof Π_j corresponding to a corrupt party P_j for $j \in \mathcal{C}$ such that $\mathsf{V}_{\mathsf{NIZK}}(E_{j-1}, E_j, \Pi_j, N) = 1$ but extraction fails. This immediately violates extractability of NIZK, thus completing the proof of the lemma.

Lemma 21. *Assuming that* $\mathsf{NIZK} = (\mathsf{P}_{\mathsf{NIZK}}, \mathsf{V}_{\mathsf{NIZK}})$ *satisfies ZK, hybrid-1 and hybrid-2 are indistinguishable.*

Note that for $\mathcal{E}$ to distinguish between hybrid-1 and hybrid-2 with non-negligible probability, there must exist an honest party P_i for $i \in \mathcal{H}$ and a distinguisher $\mathcal{D}$ such that $\left|\Pr[\mathcal{D}(E_0, E_1, \Pi_i) = 1] - \Pr[\mathcal{D}(E_0, E_1, \overline{\Pi}_i) = 1]\right| > \mathsf{negl}(\lambda)$, where λ is the security parameter, and where $\Pi_i \leftarrow \mathsf{P}_{\mathsf{NIZK}}(E_{i-1}, E_i, \phi_i, n, N)$ and $\overline{\Pi}_i \leftarrow \mathsf{Sim}_{\mathsf{NIZK}}(E_{i-1}, E_i, N)$. This immediately violates the ZK property of NIZK, thus completing the proof of the lemma.

Finally, hybrid-2 and hybrid-3 are identical by inspection, thus completing the proof of Theorem 10.

Chapter 7

Conclusion

It's a magical world, Hobbes, ol' buddy… let's go exploring!

— Bill Watterson, Last words of Calvin and Hobbes

In this thesis, we analyzed how SIDH-based protocols can be used to develop static-key constructions. In particular, we assessed the security of three protocols: the Jao-Urbanik variant of k-SIDH, the HealSIDH validation method, and the OPRF by Boneh et al. This showed the limitations and the difficulties in protecting long-term static secrets with SIDH. By building upon these results, we also proposed a new OPRF construction, together with a trusted-setup protocol to securely generate the parameters required within the OPRF protocol. Altogether, this work brings us closer to a post-quantum internet with quantum-resistant versions of commonly used protocols such as NIKEs and OPRFs; nonetheless, much more work is still needed.

The design of efficient post-quantum protocols with long-term static keys can take different paths. On one hand, it may be ideal to develop alternative primitives that achieve the same functionality with different structures. For instance, Brendel et al. [BFGJS22] proposed a post-quantum alternative to the Signal protocol that avoids NIKEs altogether by relying on designated-verifier signatures. However, it is also hard to develop efficient and post-quantum versions of such signatures. Indeed, the SIDH-based Signal protocol

proposed by Dobson and Galbraith [DG22] was far more efficient than the version with designated-verifier signatures. Thus, reinventing new protocols and primitives may not solve the issue. Moreover, post-quantum drop-in replacements allow cryptographers to re-use and build upon decade-long analyses of classical protocols; switching to new ones would require significantly more work to establish their correctness and security.

On the other hand, it is likely that we will need efficient post-quantum drop-in replacements for all the primitives we currently use, including static-key protocols such as non-interactive key exchanges and oblivious pseudorandom functions. For this, it may be possible that developments come from non-isogeny-based approaches. Lattice-based techniques may eventually lead to interesting results. After all, one (inefficient) OPRF protocol based on lattices already exists, and the recent developments on lattice-based blind signatures [PK22; AKSY22; BLNS23], a primitive deeply linked with OPRFs, bring hope for a future efficient OPRF based on lattices. Nonetheless, some limitations of lattices with regards to static-key exchanges seem hard to overcome [GKRS22]. Similarly, potential breakthroughs in code-based or multivariate cryptography may bring useful results.

While such developments are theoretically possible, the only post-quantum constructions of non-interactive key exchanges and vaguely practical oblivious pseudorandom functions have been built on isogenies. This is not a coincidence, given the inherent ability of isogeny-based techniques to work with static keys. The efficiency of the proposed protocols is still far from practical, but the history of cryptography counts many examples of protocols whose initial performance was lackluster. Several years of innovation and improvements have made those protocols efficient and widely used. The same trend can be seen within isogeny-based protocols, although the attacks on SIDH significantly slowed it down. However, the new techniques that were developed to break SIDH may have further applications, and their study may eventually lead to new and more efficient static-key constructions.

A path forward

The SIDH attacks deeply changed our understanding of several aspects of isogeny-based protocols, but they are not the end of isogeny-based cryptography and, in some sense, might not even be the end of SIDH.

While many isogeny-based protocols, such as CSIDH and SQISign, are unaffected by the attacks on SIDH, the original SIDH protocol is fully broken by the recent attacks. Nonetheless, some countermeasures are possible. They are very recent, and their security needs to withstand the test of time; yet, they currently appear to offer valid and secure SIDH-like key exchange protocols. Such schemes are much more inefficient than the original SIDH, and as pure key-exchange protocols, they may be far from competitive. Nonetheless, they can still productively be used for complex SIDH-based protocols. One such example is the construction of the oblivious pseudorandom function (OPRF) described in Chapter 5. Despite being based on SIDH and the novel countermeasures, the resulting protocol is still the most efficient post-quantum OPRF reported in the literature. This protocol may be the first SIDH-based protocol to be proposed after the SIDH protocols, and it provides an example of research directions based on SIDH in a after the SIDH attacks. Furthermore, the attacks and the corresponding countermeasures are very recent. It remains an open problem to develop new and more efficient countermeasures, which will widen the range of SIDH-based protocols that may be competitive.

Moreover, the SIDH attacks made many computational problems that were assumed to be hard suddenly easy. This renewed understanding of the field has been used cryptanalytically, but it may be possible to constructively employ the tools used in the SIDH attacks to develop new protocols. This may be possible since the attacks on SIDH are fully classical (no quantum computer is needed) and extremely efficient. This would also not be the first instance of a constructive application of attacks on isogeny-based protocols: the attacks on unbalanced SIDH [Pet17] were used to build a one-way function and the encryption protocol SÉTA [DDFKL+21]. A similar approach might turn the SIDH attacks

into an encryption protocol, or even more advanced protocols for which no post-quantum solutions exist yet. Developing such protocols remains, at the moment, an open problem.

Regardless of the specific developments in isogeny-based constructions, it seems likely—at this stage—that a post-quantum world will require the development of practical post-quantum static-key protocols, and isogeny-based cryptography appears to be the most promising technique to do so.

Bibliography

[AKSY22] Shweta Agrawal, Elena Kirshanova, Damien Stehlé, and Anshu Yadav. "Practical, Round-Optimal Lattice-Based Blind Signatures". In: ACM Press, 2022, pp. 39–53. DOI: 10.1145/3548606.3560650 (cit. on p. 218).

[ABCMR23] Aikata Aikata, Andrea Basso, Gaetan Cassiers, Ahmet Can Mert, and Sujoy Sinha Roy. "Kavach: Lightweight masking techniques for polynomial arithmetic in lattice-based cryptography". In: *IACR Transactions on Cryptographic Hardware and Embedded Systems* 2023, Issue 3 (2023), pp. 366–390. DOI: 10.46586/tches.v2023.i3.366-390 (cit. on p. vi).

[ADMP20] Navid Alamati, Luca De Feo, Hart Montgomery, and Sikhar Patranabis. "Cryptographic Group Actions and Applications". In: *ASIACRYPT 2020, Part II*. Vol. 12492. Springer, Heidelberg, 2020, pp. 411–439. DOI: 10.1007/978-3-030-64834-3_14 (cit. on pp. 25, 171, 172).

[ADDS21] Martin R. Albrecht, Alex Davidson, Amit Deo, and Nigel P. Smart. "Round-Optimal Verifiable Oblivious Pseudorandom Functions from Ideal Lattices". In: *PKC 2021, Part II*. Vol. 12711. Springer, Heidelberg, 2021, pp. 261–289. DOI: 10.1007/978-3-030-75248-4_10 (cit. on pp. 7, 93, 129, 165).

[ABLS07] Noga Alon, Itai Benjamini, Eyal Lubetzky, and Sasha Sodin. "Non-backtracking random walks mix faster". In: *Commun. Contemp. Math.* 9.4 (2007), pp. 585–603. DOI: 10.1142/S0219199707002551 (cit. on pp. 188, 212).

[Arp22] Sarah Arpin. *Adding Level Structure to Supersingular Elliptic Curve Isogeny Graphs*. Preprint. arXiv:2203.03531. 2022. DOI: 10.48550/ARXIV.2203.03531 (cit. on p. 172).

[AJKKL16] Reza Azarderakhsh, David Jao, Kassem Kalach, Brian Koziel, and Christopher Leonardi. "Key compression for isogeny-based cryptosystems". In: *Proceedings of the 3rd ACM International Workshop on ASIA Public-Key Cryptography*. ACM. 2016, pp. 1–10 (cit. on p. 155).

[AJL17] Reza Azarderakhsh, David Jao, and Christopher Leonardi. "Post-Quantum Static-Static Key Agreement Using Multiple Protocol Instances". In: *SAC 2017*. Vol. 10719. Springer, Heidelberg, 2017, pp. 45–63. DOI: 10.1007/978-3-319-72565-9_3 (cit. on pp. 13, 19, 20, 30, 34, 35, 39, 57, 62).

[Bas22] Andrea Basso. "Poster: A Post-Quantum Oblivious PRF from Isogenies". In: ACM Press, 2022, pp. 3327–3329. DOI: 10.1145/3548606.3563542 (cit. on p. vi).

[Bas23] Andrea Basso. *A Post-Quantum Round-Optimal Oblivious PRF from Isogenies*. Cryptology ePrint Archive, Paper 2023/225. 2023. URL: https://eprint.iacr.org/2023/225 (cit. on pp. v, 127).

[BADFV+21] Andrea Basso, Furkan Aydin, Daniel Dinu, Joseph Friel, Avinash Varna, Manoj Sastry, and Santosh Ghosh. *Where Star Wars Meets Star Trek: SABER and Dilithium on the Same Polynomial Multiplier*. Cryptology ePrint Archive, Report 2021/1697. https://eprint.iacr.org/2021/1697. 2021 (cit. on p. vi).

[BCCDF+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: *EUROCRYPT 2023, Part II*. Springer, Heidelberg, 2023, pp. 405–437. DOI: 10.1007/978-3-031-30617-4_14 (cit. on pp. v, 13, 16, 151, 164, 169).

[BFPW22] Andrea Basso, Tako Boris Fouotsa, Christophe Petit, and Charlotte Weitkämper. *Another Look at Adaptive Attacks on SIDH: Breaking HealSIDH*. Unpublished. 2022 (cit. on pp. v, 55).

[BKMPS21] Andrea Basso, Péter Kutas, Simon-Philipp Merz, Christophe Petit, and Antonio Sanso. "Cryptanalysis of an Oblivious PRF from Supersingular Isogenies". In: *ASIACRYPT 2021, Part I*. Vol. 13090. Springer, Heidelberg, 2021, pp. 160–184. DOI: 10.1007/978-3-030-92062-3_6 (cit. on pp. v, 7, 91, 129–131, 134, 135, 138, 139, 144, 164).

[BKMPW20] Andrea Basso, Péter Kutas, Simon-Philipp Merz, Christophe Petit, and Charlotte Weitkämper. "On Adaptive Attacks Against Jao-Urbanik's Isogeny-Based Protocol". In: *AFRICACRYPT 20*. Vol. 12174. Springer, Heidelberg, 2020, pp. 195–213. DOI: 10.1007/978-3-030-51938-4_10 (cit. on pp. v, 13, 29, 57, 63).

[BR21] Andrea Basso and Sujoy Sinha Roy. "Optimized Polynomial Multiplier Architectures for Post-Quantum KEM Saber". In: *2021 58th ACM/IEEE Design Automation Conference (DAC)*. 2021, pp. 1285–1290. DOI: 10.1109/DAC18074.2021.9586219 (cit. on p. vi).

[BDLS20] Daniel J Bernstein, Luca De Feo, Antonin Leroux, and Benjamin Smith. "Faster computation of isogenies of large prime degree". In: *Open Book Series* 4.1 (2020), pp. 39–55. DOI: 10.2140/obs.2020.4.39 (cit. on p. 178).

[BKV19] Ward Beullens, Thorsten Kleinjung, and Frederik Vercauteren. "CSI-FiSh: Efficient Isogeny Based Signatures Through Class Group Computations". In: *ASIACRYPT 2019, Part I*. Vol. 11921. Springer, Heidelberg, 2019, pp. 227–247. DOI: 10.1007/978-3-030-34578-5_9 (cit. on pp. 172, 175).

[BLNS23] Ward Beullens, Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Gregor Seiler. *Lattice-Based Blind Signatures: Short, Efficient, and Round-Optimal*. Cryptology ePrint Archive, Report 2023/077. https://eprint.iacr.org/2023/077. 2023 (cit. on p. 218).

[BKW20] Dan Boneh, Dmitry Kogan, and Katharine Woo. “Oblivious Pseudorandom Functions from Isogenies”. In: *ASIACRYPT 2020, Part II*. Vol. 12492. Springer, Heidelberg, 2020, pp. 520–550. DOI: 10.1007/978-3-030-64834-3_18 (cit. on pp. 7, 18, 19, 22–24, 93, 94, 96, 100–103, 112, 120, 129–134, 137, 138, 144, 149, 151, 153, 162, 165).

[BS20] Xavier Bonnetain and André Schrottenloher. “Quantum Security Analysis of CSIDH”. In: *EUROCRYPT 2020, Part II*. Vol. 12106. Springer, Heidelberg, 2020, pp. 493–522. DOI: 10.1007/978-3-030-45724-2_17 (cit. on p. 14).

[BBDFG+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. https://eprint.iacr.org/2022/518. 2022 (cit. on pp. 25, 174).

[BCCGP16] Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Jens Groth, and Christophe Petit. “Efficient Zero-Knowledge Arguments for Arithmetic Circuits in the Discrete Log Setting”. In: *EUROCRYPT 2016, Part II*. Vol. 9666. Springer, Heidelberg, 2016, pp. 327–357. DOI: 10.1007/978-3-662-49896-5_12 (cit. on p. 198).

[BFGJS22] Jacqueline Brendel, Rune Fiedler, Felix Günther, Christian Janson, and Douglas Stebila. “Post-quantum Asynchronous Deniable Key Exchange and the Signal Handshake”. In: *PKC 2022, Part II*. Springer, Heidelberg, 2022, pp. 3–34. DOI: 10.1007/978-3-030-97131-1_1 (cit. on p. 217).

[BD21] Jeffrey Burdges and Luca De Feo. “Delay Encryption”. In: *EUROCRYPT 2021, Part I*. Vol. 12696. Springer, Heidelberg, 2021, pp. 302–326. DOI: 10.1007/978-3-030-77870-5_11 (cit. on pp. 121, 171, 175).

[Can01] Ran Canetti. “Universally Composable Security: A New Paradigm for Cryptographic Protocols”. In: *42nd FOCS*. IEEE Computer Society Press, 2001, pp. 136–145. DOI: 10.1109/SFCS.2001.959888 (cit. on pp. 25, 130, 136).

[CCL15] Ran Canetti, Asaf Cohen, and Yehuda Lindell. “A Simpler Variant of Universally Composable Security for Standard Multiparty Computation”. In: *CRYPTO 2015, Part II*. Vol. 9216. Springer, Heidelberg, 2015, pp. 3–22. DOI: 10.1007/978-3-662-48000-7_1 (cit. on pp. 196, 201).

[CHL22] S. Casacuberta, J. Hesse, and A. Lehmann. “SoK: Oblivious Pseudorandom Functions”. In: *2022 IEEE 7th European Symposium on Security and Privacy (EuroS&P)*. IEEE Computer Society, 2022, pp. 625–646. DOI: 10.1109/EuroSP53844.2022.00045 (cit. on p. 7).

[CD23] Wouter Castryck and Thomas Decru. “An Efficient Key Recovery Attack on SIDH”. In: Springer, Heidelberg, 2023, pp. 423–447. DOI: 10.1007/978-3-031-30589-4_15 (cit. on pp. 14, 130, 133, 145, 156, 172, 175).

[CDS20] Wouter Castryck, Thomas Decru, and Benjamin Smith. “Hash functions from superspecial genus-2 curves using Richelot isogenies”. In: *J. Math. Cryptol.* 14.1 (2020), pp. 268–292. DOI: 10.1515/jmc-2019-0021 (cit. on p. 86).

[CLMPR18] Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. “CSIDH: An Efficient Post-Quantum Commutative Group Action”. In: *ASIACRYPT 2018, Part III*. Vol. 11274. Springer, Heidelberg, 2018, pp. 395–427. DOI: 10.1007/978-3-030-03332-3_15 (cit. on pp. 14, 94, 170).

[CPV20] Wouter Castryck, Lorenz Panny, and Frederik Vercauteren. “Rational Isogenies from Irrational Endomorphisms”. In: *EUROCRYPT 2020, Part II*. Vol. 12106. Springer, Heidelberg, 2020, pp. 523–548. DOI: 10.1007/978-3-030-45724-2_18 (cit. on pp. 120, 173, 174).

[CLG09] Denis Xavier Charles, Kristin E. Lauter, and Eyal Z. Goren. “Cryptographic Hash Functions from Expander Graphs”. In: *Journal of Cryptology* 22.1 (2009), pp. 93–113. DOI: 10.1007/s00145-007-9002-x (cit. on pp. 10, 25, 94, 139, 170, 171, 173).

[Cha82] David Chaum. “Blind Signatures for Untraceable Payments”. In: *CRYPTO’82*. Plenum Press, New York, USA, 1982, pp. 199–203 (cit. on pp. 93, 129).

[CRT22] Jorge Chavez-Saab, Francisco Rodríguez-Henríquez, and Mehdi Tibouchi. “Verifiable Isogeny Walks: Towards an Isogeny-Based Postquantum VDF”. In: *Selected Areas in Cryptography*. Springer International Publishing, 2022, pp. 441–460. DOI: 10.1007/978-3-030-99277-4_21 (cit. on p. 189).

[CCJR22] 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”. In: *Journal of Cryptographic Engineering* 12.3 (2022), pp. 349–368. DOI: 10.1007/s13389-021-00271-w (cit. on p. 14).

[CMP22] Jesús-Javier Chi-Domínguez, Víctor Mateu, and Lucas Pandolfo Perin. *SIDH-sign: an efficient SIDH PoK-based signature*. Cryptology ePrint Archive, Report 2022/475. https://eprint.iacr.org/2022/475. 2022 (cit. on p. 13).

[CJS14] Andrew Childs, David Jao, and Vladimir Soukharev. “Constructing elliptic curve isogenies in quantum subexponential time”. In: *Journal of Mathematical Cryptology* 8.1 (2014), pp. 1–29. DOI: doi:10.1515/jmc-2012-0016 (cit. on pp. 9, 11).

[CJLNR+17] Craig Costello, David Jao, Patrick Longa, Michael Naehrig, Joost Renes, and David Urbanik. “Efficient Compression of SIDH Public Keys”. In: *EUROCRYPT 2017, Part I*. Vol. 10210. Springer, Heidelberg, 2017, pp. 679–706. DOI: 10.1007/978-3-319-56620-7_24 (cit. on p. 206).

[CLN16] Craig Costello, Patrick Longa, and Michael Naehrig. “Efficient Algorithms for Supersingular Isogeny Diffie-Hellman”. In: *CRYPTO 2016, Part I*. Vol. 9814. Springer, Heidelberg, 2016, pp. 572–601. DOI: 10.1007/978-3-662-53018-4_21 (cit. on pp. 56, 61, 67, 132).

[Cou06] Jean-Marc Couveignes. *Hard Homogeneous Spaces*. Cryptology ePrint Archive, Report 2006/291. https://eprint.iacr.org/2006/291. 2006 (cit. on pp. 9, 14, 93, 172).

[DKRVM+20] Jan-Pieter D’Anvers, Angshuman Karmakar, Sujoy Sinha Roy, Frederik Vercauteren, Jose Maria Bermudo Mera, Michiel Van Beirendonck, and Andrea Basso. *SABER*. Tech. rep. available at https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions. National Institute of Standards and Technology, 2020 (cit. on p. vi).

[DFSW23] Alex Davidson, Armando Faz-Hernandez, Nick Sullivan, and Christopher A. Wood. *Oblivious Pseudorandom Functions (OPRFs) using Prime-Order Groups*. Internet-Draft draft-irtf-cfrg-voprf-17. Work in Progress. Internet Research Task Force, 2023 (cit. on p. 129).

[DGSTV18] Alex Davidson, Ian Goldberg, Nick Sullivan, George Tankersley, and Filippo Valsorda. “Privacy Pass: Bypassing Internet Challenges Anonymously”. In: *Proc. Priv. Enhancing Technol.* 2018.3 (2018), pp. 164–180. DOI: 10.1515/popets-2018-0026 (cit. on pp. 93, 128).

[DSW19] Alex Davidson, Nick Sullivan, and Christopher A. Wood. *Oblivious Pseudorandom Functions (OPRFs) Using Prime-Order Groups*. Internet-Draft draft-sullivan-cfrg-voprf-03. Internet Engineering Task Force / Internet Engineering Task Force, 2019 (cit. on p. 93).

[DDFKL+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: *ASIACRYPT 2021, Part IV*. Vol. 13093. Springer, Heidelberg, 2021, pp. 249–278. DOI: 10.1007/978-3-030-92068-5_9 (cit. on pp. 170, 171, 219).

[DDGZ22] Luca De Feo, Samuel Dobson, Steven D. Galbraith, and Lukas Zobernig. “SIDH Proof of Knowledge”. In: *ASIACRYPT 2022, Part II*. Springer, Heidelberg, 2022, pp. 310–339. DOI: 10.1007/978-3-031-22966-4_11 (cit. on pp. 13, 132, 149, 171, 175, 176, 189, 192).

[DG19] Luca De Feo and Steven D. Galbraith. “SeaSign: Compact Isogeny Signatures from Class Group Actions”. In: *EUROCRYPT 2019, Part III*. Vol. 11478. Springer, Heidelberg, 2019, pp. 759–789. DOI: 10.1007/978-3-030-17659-4_26 (cit. on pp. 172, 175, 189).

[DJP14] Luca De Feo, David Jao, and Jérôme Plût. “Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies”. In: *Journal of Mathematical Cryptology* 8.3 (2014), pp. 209–247. DOI: 10.1515/jmc-2012-0015 (cit. on pp. 171, 175, 189, 196).

[DKS18] Luca De Feo, Jean Kieffer, and Benjamin Smith. “Towards Practical Key Exchange from Ordinary Isogeny Graphs”. In: *ASIACRYPT 2018, Part III*. Vol. 11274. Springer, Heidelberg, 2018, pp. 365–394. DOI: 10.1007/978-3-030-03332-3_14 (cit. on pp. 9, 172).

[DKLPW20] Luca De Feo, David Kohel, Antonin Leroux, Christophe Petit, and Benjamin Wesolowski. “SQISign: Compact Post-quantum Signatures from Quaternions and Isogenies”. In: *ASIACRYPT 2020, Part I*. Vol. 12491. Springer, Heidelberg, 2020, pp. 64–93. DOI: 10.1007/978-3-030-64837-4_3 (cit. on pp. 170–172).

[DMPS19] Luca De Feo, Simon Masson, Christophe Petit, and Antonio Sanso. “Verifiable Delay Functions from Supersingular Isogenies and Pairings”. In: *ASIACRYPT 2019, Part I*. Vol. 11921. Springer, Heidelberg, 2019, pp. 248–277. DOI: 10.1007/978-3-030-34578-5_10 (cit. on pp. 170, 171).

[Del80] Pierre Deligne. “La conjecture de Weil. II”. In: *Inst. Hautes Études Sci. Publ. Math.* 52 (1980), pp. 137–252. URL: http://www.numdam.org/item?id=PMIHES_1980__52__137_0 (cit. on p. 185).

[DRRT18] Daniel Demmler, Peter Rindal, Mike Rosulek, and Ni Trieu. “PIR-PSI: Scaling Private Contact Discovery”. In: *Proc. Priv. Enhancing Technol.* 2018.4 (2018), pp. 159–178. DOI: 10.1515/popets-2018-0037 (cit. on pp. 6, 92, 128).

[Den03] Alexander W. Dent. “A Designer’s Guide to KEMs”. In: *Cryptography and Coding, 9th IMA International Conference, Cirencester, UK, December 16-18, 2003, Proceedings*. Vol. 2898. Springer, 2003, pp. 133–151. DOI: 10.1007/978-3-540-40974-8_12 (cit. on p. 57).

[DS05] Fred Diamond and Jerry Shurman. *A first course in modular forms*. Vol. 228. Springer-Verlag, New York, 2005, pp. xvi+436 (cit. on pp. 184–186).

[DH76] W. Diffie and M. Hellman. “New directions in cryptography”. In: *IEEE Transactions on Information Theory* 22.6 (1976), pp. 644–654. DOI: 10.1109/TIT.1976.1055638 (cit. on pp. 2, 5).

[DG22] Samuel Dobson and Steven D. Galbraith. “Post-Quantum Signal Key Agreement from SIDH”. In: *Post-Quantum Cryptography*. Springer International Publishing, 2022, pp. 422–450 (cit. on pp. 14, 218).

[DGLTZ20] Samuel Dobson, Steven D. Galbraith, Jason T. LeGrow, Yan Bo Ti, and Lukas Zobernig. “An adaptive attack on 2-SIDH”. In: *Int. J. Comput. Math. Comput. Syst. Theory* 5.4 (2020), pp. 282–299. DOI: 10.1080/23799927.2020.1822446 (cit. on pp. 13, 20, 30, 31, 37, 39, 41, 44–46, 50, 57, 62).

[EJKM22] Edward Eaton, David Jao, Chelsea Komlo, and Youcef Mokrani. “Towards Post-Quantum Key-Updatable Public-Key Encryption via Supersingular Isogenies”. In: *SAC 2021*. Vol. 13203. Springer, Heidelberg, 2022, pp. 461–482. DOI: 10.1007/978-3-030-99277-4_22 (cit. on p. 14).

[EHLMP18] Kirsten Eisenträger, Sean Hallgren, Kristin E. Lauter, Travis Morrison, and Christophe Petit. "Supersingular Isogeny Graphs and Endomorphism Rings: Reductions and Solutions". In: *EUROCRYPT 2018, Part III*. Vol. 10822. Springer, Heidelberg, 2018, pp. 329–368. DOI: 10.1007/978-3-319-78372-7_11 (cit. on pp. 10, 120, 139, 170–172).

[ECSJR15] Adam Everspaugh, Rahul Chatterjee, Samuel Scott, Ari Juels, and Thomas Ristenpart. "The Pythia PRF Service". In: *USENIX Security 2015*. USENIX Association, 2015, pp. 547–562 (cit. on pp. 7, 92, 128).

[Feo17] Luca De Feo. "Mathematics of Isogeny Based Cryptography". In: *CoRR* abs/1711.04062 (2017). URL: http://arxiv.org/abs/1711.04062 (cit. on pp. 59, 97).

[FS87] Amos Fiat and Adi Shamir. "How to Prove Yourself: Practical Solutions to Identification and Signature Problems". In: *CRYPTO'86*. Vol. 263. Springer, Heidelberg, 1987, pp. 186–194. DOI: 10.1007/3-540-47721-7_12 (cit. on pp. 156, 160, 197).

[FT19] E. Victor Flynn and Yan Bo Ti. "Genus Two Isogeny Cryptography". In: *Post-Quantum Cryptography - 10th International Conference, PQCrypto 2019*. Springer, Heidelberg, 2019, pp. 286–306. DOI: 10.1007/978-3-030-25510-7_16 (cit. on pp. 58, 86).

[Fou22] Tako Boris Fouotsa. "On the design and cryptanalysis of Isogeny-Based Public Key Encryption schemes". PhD thesis. Università degli Studi Roma Tre, 2022. URL: http://www.matfis.uniroma3.it/Allegati/Dottorato/TESI/fboris/Fouotsa_thesis_final_.pdf (cit. on p. 59).

[FKMT22] Tako Boris Fouotsa, Péter Kutas, Simon-Philipp Merz, and Yan Bo Ti. "On the Isogeny Problem with Torsion Point Information". In: *PKC 2022, Part I*. Springer, Heidelberg, 2022, pp. 142–161. DOI: 10.1007/978-3-030-97121-2_6 (cit. on p. 171).

[FMP23a] Tako Boris Fouotsa, Tomoki Moriya, and Christophe Petit. "M-SIDH and MD-SIDH: Countering SIDH Attacks by Masking Information". In: Springer, Heidelberg, 2023, pp. 282–309. DOI: 10.1007/978-3-031-30589-4_10 (cit. on pp. 130, 133, 146, 147, 163).

[FMP23b] Tako Boris Fouotsa, Tomoki Moriya, and Christophe Petit. *M-SIDH and MD-SIDH: countering SIDH attacks by masking information*. Cryptology ePrint Archive, Report 2023/013. https://eprint.iacr.org/2023/013. 2023 (cit. on p. 17).

[FP21] Tako Boris Fouotsa and Christophe Petit. "SHealS and HealS: Isogeny-Based PKEs from a Key Validation Method for SIDH". In: *ASIACRYPT 2021, Part IV*. Vol. 13093. Springer, Heidelberg, 2021, pp. 279–307. DOI: 10.1007/978-3-030-92068-5_10 (cit. on pp. 13, 19, 21, 57, 71, 72, 76).

[FP22] Tako Boris Fouotsa and Christophe Petit. "A New Adaptive Attack on SIDH". In: *CT-RSA 2022*. Springer, Heidelberg, 2022, pp. 322–344. DOI: 10.1007/978-3-030-95312-6_14 (cit. on pp. 12, 56, 61).

[FIPR05] Michael J. Freedman, Yuval Ishai, Benny Pinkas, and Omer Reingold. "Keyword Search and Oblivious Pseudorandom Functions". In: *TCC 2005*. Vol. 3378. Springer, Heidelberg, 2005, pp. 303–324. DOI: 10.1007/978-3-540-30576-7_17 (cit. on pp. 98, 135).

[FHKP13] Eduarda S. V. Freire, Dennis Hofheinz, Eike Kiltz, and Kenneth G. Paterson. "Non-Interactive Key Exchange". In: *PKC 2013*. Vol. 7778. Springer, Heidelberg, 2013, pp. 254–271. DOI: 10.1007/978-3-642-36362-7_17 (cit. on p. 5).

[FO99] Eiichiro Fujisaki and Tatsuaki Okamoto. "Secure Integration of Asymmetric and Symmetric Encryption Schemes". In: *CRYPTO'99*. Vol. 1666. Springer, Heidelberg, 1999, pp. 537–554. DOI: 10.1007/3-540-48405-1_34 (cit. on p. 12).

[FO13] Eiichiro Fujisaki and Tatsuaki Okamoto. "Secure Integration of Asymmetric and Symmetric Encryption Schemes". In: *Journal of Cryptology* 26.1 (2013), pp. 80–101. DOI: 10.1007/s00145-011-9114-1 (cit. on pp. 12, 56, 57).

[Gal18] Steven D. Galbraith. *Authenticated key exchange for SIDH*. Cryptology ePrint Archive, Report 2018/266. https://eprint.iacr.org/2018/266. 2018 (cit. on pp. 12, 19, 149).

[GL22] Steven D. Galbraith and Yi-Fu Lai. "Attack on SHealS and HealS: The Second Wave of GPST". In: *Post-Quantum Cryptography - 13th International Workshop, PQCrypto 2022, Virtual Event, September 28-30, 2022, Proceedings*. Vol. 13512. Springer, 2022, pp. 399–421. DOI: 10.1007/978-3-031-17234-2_19 (cit. on pp. 13, 21, 57, 58, 80, 82).

[GPST16] Steven D. Galbraith, Christophe Petit, Barak Shani, and Yan Bo Ti. "On the Security of Supersingular Isogeny Cryptosystems". In: *ASIACRYPT 2016, Part I*. Vol. 10031. Springer, Heidelberg, 2016, pp. 63–91. DOI: 10.1007/978-3-662-53887-6_3 (cit. on pp. 12, 21, 30, 34–36, 41, 44, 56, 58, 61, 67, 69, 100, 132, 156, 170–172).

[GPS17] Steven D. Galbraith, Christophe Petit, and Javier Silva. "Identification Protocols and Signature Schemes Based on Supersingular Isogeny Problems". In: *ASIACRYPT 2017, Part I*. Vol. 10624. Springer, Heidelberg, 2017, pp. 3–33. DOI: 10.1007/978-3-319-70694-8_1 (cit. on p. 189).

[GPS20] Steven D. Galbraith, Christophe Petit, and Javier Silva. "Identification Protocols and Signature Schemes Based on Supersingular Isogeny Problems". In: *Journal of Cryptology* 33.1 (2020), pp. 130–175. DOI: 10.1007/s00145-019-09316-0 (cit. on pp. 170, 171, 189).

[GV18] Steven D. Galbraith and Frederik Vercauteren. "Computational problems in supersingular elliptic curve isogenies". In: *Quantum Information Processing* 17.10 (2018), p. 265. DOI: 10.1007/s11128-018-2023-6 (cit. on p. 16).

[GPV21] Wissam Ghantous, Federico Pintore, and Mattia Veroni. *Collisions in Supersingular Isogeny Graphs and the SIDH-based Identification Protocol*. Cryptology ePrint Archive, Report 2021/1051. https://eprint.iacr.org/2021/1051. 2021 (cit. on pp. 175, 192).

[GKRS22] Siyao Guo, Pritish Kamath, Alon Rosen, and Katerina Sotiraki. "Limits on the Efficiency of (Ring) LWE-Based Non-interactive Key Exchange". In: *Journal of Cryptology* 35.1 (2022), p. 1. DOI: 10.1007/s00145-021-09406-y (cit. on pp. 6, 218).

[HPS89] Hiroaki Hijikata, Arnold K. Pizer, and Thomas R. Shemanske. "The basis problem for modular forms on $\Gamma_0(N)$". In: *Mem. Amer. Math. Soc.* 82.418 (1989), pp. vi+159. DOI: 10.1090/memo/0418 (cit. on p. 184).

[HHK17] Dennis Hofheinz, Kathrin Hövelmanns, and Eike Kiltz. "A Modular Analysis of the Fujisaki-Okamoto Transformation". In: *TCC 2017, Part I*. Vol. 10677. Springer, Heidelberg, 2017, pp. 341–371. DOI: 10.1007/978-3-319-70500-2_12 (cit. on pp. 34, 57).

[HBDEF+22] Andreas Hülsing, Daniel J. Bernstein, Christoph Dobraunig, Maria Eichlseder, Scott Fluhrer, Stefan-Lukas Gazdag, Panos Kampanakis, Stefan Kölbl, Tanja Lange, Martin M. Lauridsen, Florian Mendel, Ruben Niederhagen, Christian Rechberger, Joost Rijneveld, Peter Schwabe, Jean-Philippe Aumasson, Bas Westerbaan, and Ward Beullens. *SPHINCS$^+$*. Tech. rep. available at https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022. National Institute of Standards and Technology, 2022 (cit. on p. 4).

[Hus04] Dale Husemöller. *Elliptic curves*. Second. Vol. 111. Springer-Verlag, New York, 2004, pp. xxii+487 (cit. on p. 210).

[IABSP23] Malik Imran, Felipe Almeida, Andrea Basso, Sujoy Sinha Roy, and Samuel Pagliarini. "High-speed SABER key encapsulation mechanism in 65nm CMOS". In: *Journal of Cryptographic Engineering* (2023). DOI: 10.1007/s13389-023-00316-2 (cit. on p. vi).

[IARBR+21] Malik Imran, Felipe Almeida, Jaan Raik, Andrea Basso, Sujoy Sinha Roy, and Samuel Pagliarini. "Design Space Exploration of SABER in 65nm ASIC". In: *Proceedings of the 5th Workshop on Attacks and Solutions in Hardware Security*. Association for Computing Machinery, 2021, pp. 85–90. DOI: 10.1145/3474376.3487278 (cit. on p. vi).

[JACCD+17] David Jao, Reza Azarderakhsh, Matthew Campagna, Craig Costello, Luca De Feo, Basil Hess, Amir Jalali, Brian Koziel, Brian LaMacchia, Patrick Longa, Michael Naehrig, Joost Renes, Vladimir Soukharev, and David Urbanik. *SIKE*. Tech. rep. available at https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-1-submissions. National Institute of Standards and Technology, 2017 (cit. on p. 12).

[JACCD+20] David Jao, Reza Azarderakhsh, Matthew Campagna, Craig Costello, Luca De Feo, Basil Hess, Amir Jalali, Brian Koziel, Brian LaMacchia, Patrick Longa, Michael Naehrig, Joost Renes, Vladimir Soukharev, David Urbanik, Geovandro Pereira, Koray Karabina, and Aaron Hutchinson. *SIKE*. Tech. rep. available at https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions. National Institute of Standards and Technology, 2020 (cit. on pp. 30, 57, 66, 94, 172).

[JD11] David Jao and Luca De Feo. "Towards Quantum-Resistant Cryptosystems from Supersingular Elliptic Curve Isogenies". In: *Post-Quantum Cryptography - 4th International Workshop, PQCrypto 2011*. Springer, Heidelberg, 2011, pp. 19–34. DOI: 10.1007/978-3-642-25405-5_2 (cit. on pp. 10, 13, 30, 34, 39, 56, 60, 94, 97, 131, 137, 149, 153, 170).

[JKKX16] S. Jarecki, A. Kiayias, H. Krawczyk, and J. Xu. "Highly-Efficient and Composable Password-Protected Secret Sharing (Or: How to Protect Your Bitcoin Wallet Online)". In: *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE Computer Society, 2016, pp. 276–291. DOI: 10.1109/EuroSP.2016.30 (cit. on p. 136).

[JKK14] Stanislaw Jarecki, Aggelos Kiayias, and Hugo Krawczyk. "Round-Optimal Password-Protected Secret Sharing and T-PAKE in the Password-Only Model". In: *ASIACRYPT 2014, Part II*. Vol. 8874. Springer, Heidelberg, 2014, pp. 233–253. DOI: 10.1007/978-3-662-45608-8_13 (cit. on pp. 7, 93, 128, 129, 136).

[JKKX17] Stanislaw Jarecki, Aggelos Kiayias, Hugo Krawczyk, and Jiayu Xu. "TOPPSS: Cost-Minimal Password-Protected Secret Sharing Based on Threshold OPRF". In: *ACNS 17*. Vol. 10355. Springer, Heidelberg, 2017, pp. 39–58. DOI: 10.1007/978-3-319-61204-1_3 (cit. on p. 136).

[JKX18] Stanislaw Jarecki, Hugo Krawczyk, and Jiayu Xu. "OPAQUE: An Asymmetric PAKE Protocol Secure Against Pre-computation Attacks". In: *EUROCRYPT 2018, Part III*. Vol. 10822. Springer, Heidelberg, 2018, pp. 456–486. DOI: 10.1007/978-3-319-78372-7_15 (cit. on pp. 6, 92, 93, 128, 136).

[JL09] Stanislaw Jarecki and Xiaomin Liu. "Efficient Oblivious Pseudorandom Function with Applications to Adaptive OT and Secure Computation of Set Intersection". In: *TCC 2009*. Vol. 5444. Springer, Heidelberg, 2009, pp. 577–594. DOI: 10.1007/978-3-642-00457-5_34 (cit. on pp. 6, 7, 92, 128, 135).

[Kan97] Ernst Kani. "The number of curves of genus two with elliptic differentials". In: *Journal für die reine und angewandte Mathematik* 1997.485 (1997), pp. 93–122. DOI: doi:10.1515/crll.1997.485.93 (cit. on p. 15).

[KO04] Jonathan Katz and Rafail Ostrovsky. "Round-Optimal Secure Two-Party Computation". In: *CRYPTO 2004*. Vol. 3152. Springer, Heidelberg, 2004, pp. 335–354. DOI: 10.1007/978-3-540-28628-8_21 (cit. on p. 7).

[Koh96] David Kohel. “Endomorphism rings of elliptic curves over finite fields”. PhD thesis. University of California at Berkley, 1996. URL: http://iml.univ-mrs.fr/~kohel/pub/thesis.pdf (cit. on p. 173).

[KTW22] Sabrina Kunzweiler, Yan Bo Ti, and Charlotte Weitkämper. “Secret Keys in Genus-2 SIDH”. In: *SAC 2021*. Vol. 13203. Springer, Heidelberg, 2022, pp. 483–507. DOI: 10.1007/978-3-030-99277-4_23 (cit. on pp. 66, 86–88).

[Kup05] Greg Kuperberg. “A Subexponential-Time Quantum Algorithm for the Dihedral Hidden Subgroup Problem”. In: *SIAM J. Comput.* 35.1 (2005), pp. 170–188. DOI: 10.1137/S0097539703436345 (cit. on p. 9).

[LGD21] Yi-Fu Lai, Steven D. Galbraith, and Cyprien Delpech de Saint Guilhem. “Compact, Efficient and UC-Secure Isogeny-Based Oblivious Transfer”. In: *EUROCRYPT 2021, Part I*. Vol. 12696. Springer, Heidelberg, 2021, pp. 213–241. DOI: 10.1007/978-3-030-77870-5_8 (cit. on pp. 25, 129, 171, 172).

[LKLM21] Kristin Lauter, Sreekanth Kannepalli, Kim Laine, and Radames Cruz Moreno. *Password Monitor: Safeguarding passwords in Microsoft Edge*. 2021. URL: https://www.microsoft.com/en-us/research/blog/password-monitor-safeguarding-passwords-in-microsoft-edge/ (cit. on p. 6).

[Leo20] Christopher Leonardi. *A Note on the Ending Elliptic Curve in SIDH*. Cryptology ePrint Archive, Report 2020/262. https://eprint.iacr.org/2020/262. 2020 (cit. on p. 11).

[Ler22] Antonin Leroux. “A New Isogeny Representation and Applications to Cryptography”. In: *ASIACRYPT 2022, Part II*. Springer, Heidelberg, 2022, pp. 3–35. DOI: 10.1007/978-3-031-22966-4_1 (cit. on p. 14).

[LP17] David A. Levin and Yuval Peres. *Markov chains and mixing times*. American Mathematical Society, Providence, RI, 2017, pp. xvi+447. DOI: 10.1090/mbk/107 (cit. on p. 212).

[LPASC+19] Lucy Li, Bijeeta Pal, Junade Ali, Nick Sullivan, Rahul Chatterjee, and Thomas Ristenpart. “Protocols for Checking Compromised Credentials”. In: *ACM CCS 2019*. ACM Press, 2019, pp. 1387–1403. DOI: 10.1145/3319535.3354229 (cit. on pp. 6, 92, 128).

[LB20] Jonhatan Love and Dan Boneh. “Supersingular curves with small non-integer endomorphisms”. In: *The Open Book Series: Proceedings of the Fourteenth Algorithmic Number Theory Symposium (ANTS XIV)* 4.1 (2020), pp. 7–22. DOI: 10.2140/obs.2020.4.7 (cit. on pp. 120, 173, 174).

[LDKLS+22] Vadim Lyubashevsky, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Peter Schwabe, Gregor Seiler, Damien Stehlé, and Shi Bai. *CRYSTALS-DILITHIUM*. Tech. rep. available at https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022. National Institute of Standards and Technology, 2022 (cit. on p. 4).

[MMPPW23] Luciano Maino, Chloe Martindale, Lorenz Panny, Giacomo Pope, and Benjamin Wesolowski. “A Direct Key Recovery Attack on SIDH”. In: Springer, Heidelberg, 2023, pp. 448–471. DOI: 10.1007/978-3-031-30589-4_16 (cit. on pp. 14, 130, 133, 145, 156, 172, 175).

[MP16] Moxie Marlinspike and Trevor Perrin. *The X3DH Key Agreement Protocol*. 2016. URL: https://signal.org/docs/specifications/x3dh/x3dh.pdf (cit. on pp. 5, 14).

[MP19] Chloe Martindale and Lorenz Panny. *How to not break SIDH*. Cryptology ePrint Archive, Report 2019/558. https://eprint.iacr.org/2019/558. 2019 (cit. on p. 17).

[MMP20] Simon-Philipp Merz, Romy Minko, and Christophe Petit. “Another Look at Some Isogeny Hardness Assumptions”. In: *CT-RSA 2020*. Vol. 12006. Springer, Heidelberg, 2020, pp. 496–511. DOI: 10.1007/978-3-030-40186-3_21 (cit. on p. 99).

[Mes86] Jean-François Mestre. “La méthode des graphes. Exemples et applications”. In: *Proceedings of the international conference on class numbers and fundamental units of algebraic number fields (Katata, 1986)*. Nagoya University, 1986. URL: https://wstein.org/msri06/refs/mestre-method-of-graphs/mestre-fr.pdf (cit. on p. 173).

[MP21] Michele Mosca and Marco Piani. *2021 Quantum Threat Timeline Report: Global Risk Institute*. Tech. rep. available at https://globalriskinstitute.org/publication/2021-quantum-threat-timeline-report-global-risk-institute-global-risk-institute/. evolutionQ, 2021 (cit. on p. 3).

[MMP22] Marzio Mula, Nadir Murru, and Federico Pintore. *Random sampling of supersingular elliptic curves*. Cryptology ePrint Archive, Report 2022/528. https://eprint.iacr.org/2022/528. 2022 (cit. on pp. 25, 174).

[NR97] Moni Naor and Omer Reingold. “Number-theoretic Constructions of Efficient Pseudo-random Functions”. In: *38th FOCS*. IEEE Computer Society Press, 1997, pp. 458–467. DOI: 10.1109/SFCS.1997.646134 (cit. on pp. 129, 135).

[Pan21] Lorenz Panny. “Cryptography on Isogeny Graphs”. PhD thesis. Technische Universiteit Eindhoven, 2021. URL: https://yx7.cc/docs/phd/thesis.pdf (cit. on p. 59).

[Pei20] Chris Peikert. “He Gives C-Sieves on the CSIDH”. In: *EUROCRYPT 2020, Part II*. Vol. 12106. Springer, Heidelberg, 2020, pp. 463–492. DOI: 10.1007/978-3-030-45724-2_16 (cit. on p. 14).

[Pet17] Christophe Petit. “Faster Algorithms for Isogeny Problems Using Torsion Point Images”. In: *ASIACRYPT 2017, Part II*. Vol. 10625. Springer, Heidelberg, 2017, pp. 330–353. DOI: 10.1007/978-3-319-70697-9_12 (cit. on pp. 12, 61, 171, 219).

[PL17] Christophe Petit and Kristin Lauter. *Hard and easy problems for supersingular isogeny graphs*. Cryptology ePrint Archive, Report 2017/962. https://eprint.iacr.org/2017/962. 2017 (cit. on pp. 10, 120, 139).

[PK22] Rafael del Pino and Shuichi Katsumata. “A New Framework for More Efficient Round-Optimal Lattice-Based (Partially) Blind Signature via Trapdoor Sampling”. In: *Advances in Cryptology – CRYPTO 2022*. Springer Nature Switzerland, 2022, pp. 306–336 (cit. on p. 218).

[Piz90] Arnold K. Pizer. “Ramanujan graphs and Hecke operators”. In: *Bulletin of the American Mathematical Society (N.S.)* 23.1 (1990). DOI: 10.1090/S0273-0979-1990-15918-X (cit. on p. 173).

[PFHKL+22] Thomas Prest, Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Ricosset, Gregor Seiler, William Whyte, and Zhenfei Zhang. *FALCON*. Tech. rep. available at https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022. National Institute of Standards and Technology, 2022 (cit. on p. 4).

[QKLMP+21] Victoria de Quehen, Péter Kutas, Chris Leonardi, Chloe Martindale, Lorenz Panny, Christophe Petit, and Katherine E. Stange. “Improved Torsion-Point Attacks on SIDH Variants”. In: *CRYPTO 2021, Part III*. Vol. 12827. Springer, Heidelberg, 2021, pp. 432–470. DOI: 10.1007/978-3-030-84252-9_15 (cit. on pp. 12, 17, 61, 120, 121, 171).

[RSA78] R. L. Rivest, A. Shamir, and L. Adleman. “A Method for Obtaining Digital Signatures and Public-Key Cryptosystems”. In: *Commun. ACM* 21.2 (1978), pp. 120–126. DOI: 10.1145/359340.359342 (cit. on p. 2).

[Rob22] Damien Robert. *Evaluating isogenies in polylogarithmic time*. Cryptology ePrint Archive, Report 2022/1068. https://eprint.iacr.org/2022/1068. 2022 (cit. on p. 16).

[Rob23] Damien Robert. “Breaking SIDH in Polynomial Time”. In: Springer, Heidelberg, 2023, pp. 472–503. DOI: 10.1007/978-3-031-30589-4_17 (cit. on pp. 14, 16, 17, 130, 133, 145, 156, 172, 175).

[RS06] Alexander Rostovtsev and Anton Stolbunov. *Public-Key Cryptosystem Based On Isogenies*. Cryptology ePrint Archive, Report 2006/145. https://eprint.iacr.org/2006/145. 2006 (cit. on pp. 9, 14, 93).

[RB20] Sujoy Sinha Roy and Andrea Basso. “High-speed Instruction-set Coprocessor for Lattice-based Key Encapsulation Mechanism: Saber in Hardware”. In: *IACR TCHES* 2020.4 (2020). https://tches.iacr.org/index.php/TCHES/article/view/8690, pp. 443–466. DOI: 10.13154/tches.v2020.i4.443-466 (cit. on p. v).

[SGP19] Rajeev Anand Sahu, Agnese Gini, and Ankan Pal. *Supersingular Isogeny-Based Designated Verifier Blind Signature*. Cryptology ePrint Archive, Report 2019/1498. https://eprint.iacr.org/2019/1498. 2019 (cit. on p. 13).

[Sch74] Bruno Schoeneberg. *Elliptic modular functions: an introduction*. Springer-Verlag, New York-Heidelberg, 1974, pp. viii+233 (cit. on pp. 184, 187).

[SABDK+22] Peter Schwabe, Roberto Avanzi, Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Gregor Seiler, Damien Stehlé, and Jintai Ding. *CRYSTALS-KYBER*. Tech. rep. available at https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022. National Institute of Standards and Technology, 2022 (cit. on p. 4).

[SSW20] Peter Schwabe, Douglas Stebila, and Thom Wiggers. "Post-Quantum TLS Without Handshake Signatures". In: *ACM CCS 2020*. ACM Press, 2020, pp. 1461–1480. DOI: 10.1145/3372297.3423350 (cit. on p. 5).

[SHB21] István András Seres, Máté Horváth, and Péter Burcsi. *The Legendre Pseudorandom Function as a Multivariate Quadratic Cryptosystem: Security and Applications*. Cryptology ePrint Archive, Report 2021/182. https://eprint.iacr.org/2021/182. 2021 (cit. on p. 93).

[SC18] M. Seshadri Srinath and V. Chandrasekaran. "Isogeny-based Quantum-resistant Undeniable Blind Signature Scheme". In: *International Journal of Network Security* 20.1 (2018). DOI: 10.6633/IJNS.201801.20(1).02 (cit. on p. 13).

[Sho94] P. W. Shor. "Algorithms for Quantum Computation: Discrete Logarithms and Factoring". In: *Proceedings of the 35th Annual Symposium on Foundations of Computer Science*. IEEE Computer Society, 1994, pp. 124–134. DOI: 10.1109/SFCS.1994.365700 (cit. on pp. 3, 9).

[Sil86] Joseph H. Silverman. *The arithmetic of elliptic curves*. Vol. 106. Springer, 1986 (cit. on pp. 32, 33, 49, 59, 97, 131).

[Ste22] Bruno Sterner. "Commitment Schemes from Supersingular Elliptic Curve Isogeny Graphs". In: *Mathematical Cryptology* 1.2 (2022), pp. 40–51. URL: https://journals.flvc.org/mathcryptology/article/view/130656 (cit. on pp. 25, 171).

[Sto10] Anton Stolbunov. "Constructing public-key cryptographic schemes based on class group action on a set of isogenous elliptic curves". In: *Advances in Mathematics of Communications* 4.2 (2010), pp. 215–235. DOI: 10.3934/amc.2010.4.215 (cit. on pp. 9, 172).

[SKFB21] Nick Sullivan, Dr. Hugo Krawczyk, Owen Friel, and Richard Barnes. *OPAQUE with TLS 1.3*. Internet-Draft draft-sullivan-tls-opaque-01. Internet Engineering Task Force / Internet Engineering Task Force, 2021. 13 pp. (cit. on p. 93).

[Unr15] Dominique Unruh. "Non-Interactive Zero-Knowledge Proofs in the Quantum Random Oracle Model". In: *EUROCRYPT 2015, Part II*. Vol. 9057. Springer, Heidelberg, 2015, pp. 755–784. DOI: 10.1007/978-3-662-46803-6_25 (cit. on pp. 156, 160).

[UJ18] David Urbanik and David Jao. "SoK: The Problem Landscape of SIDH". In: *Proceedings of the 5th ACM on ASIA Public-Key Cryptography Workshop, APKC@AsiaCCS, Incheon, Republic of Korea, June 4, 2018*. ACM, 2018, pp. 53–60. DOI: 10.1145/3197507.3197516 (cit. on pp. 13, 85).

[UJ20] David Urbanik and David Jao. “New Techniques for SIDH-based NIKE”. In: *Journal of Mathematical Cryptology* 14.1 (2020), pp. 120–128. DOI: doi:10.1515/jmc-2015-0056 (cit. on pp. 13, 19, 20, 31, 39–42, 51, 57, 62).

[vW99] Paul C. van Oorschot and Michael J. Wiener. “Parallel Collision Search with Cryptanalytic Applications”. In: *Journal of Cryptology* 12.1 (1999), pp. 1–28. DOI: 10.1007/PL00003816 (cit. on p. 112).

[Vél71] Jacques Vélu. “Isogénies entre courbes elliptiques”. In: *CR Acad. Sci. Paris, Séries A* 273 (1971), pp. 305–347 (cit. on pp. 33, 59, 178).

[Voi21] John Voight. *Quaternion algebras*. Vol. 288. Springer, Cham, 2021, pp. xxiii+885. DOI: 10.1007/978-3-030-56694-4 (cit. on p. 187).

[Wes22a] Benjamin Wesolowski. “Orientations and the Supersingular Endomorphism Ring Problem”. In: *EUROCRYPT 2022, Part III*. Vol. 13277. Springer, Heidelberg, 2022, pp. 345–371. DOI: 10.1007/978-3-031-07082-2_13 (cit. on pp. 170, 172).

[Wes22b] Benjamin Wesolowski. “The supersingular isogeny path and endomorphism ring problems are equivalent”. In: *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*. 2022, pp. 1100–1111. DOI: 10.1109/FOCS52979.2021.00109 (cit. on pp. 170, 173, 204).

[YAJJS17] Youngho Yoo, Reza Azarderakhsh, Amir Jalali, David Jao, and Vladimir Soukharev. “A Post-quantum Digital Signature Scheme Based on Supersingular Isogenies”. In: *Financial Cryptography 2017*. Vol. 10322. Springer, Heidelberg, 2017, pp. 163–181 (cit. on p. 13).