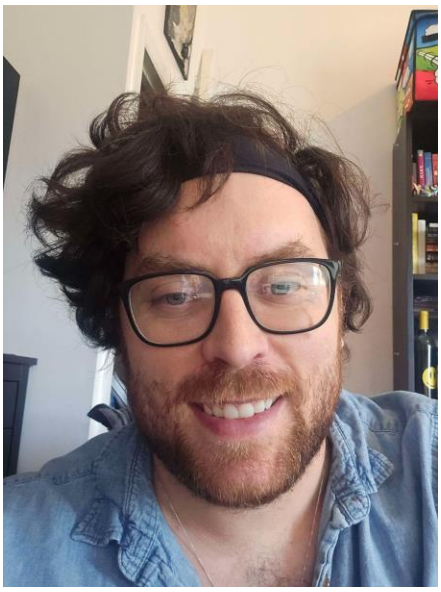
SQISIGN

Isogeny-Based Digital Signature Scheme

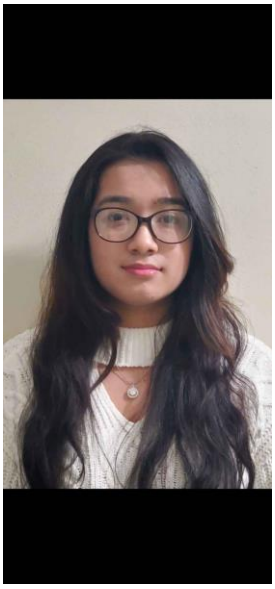
Osasere Imade
Abdul Mutallif
Anjiya Shrestha
Matthew Withee
Suresh Yhap

Midterm Research Paper
CS381/780 – Post Quantum Cryptography
Dr. Delaram Kahrobaei
Queens College, Fall 2024

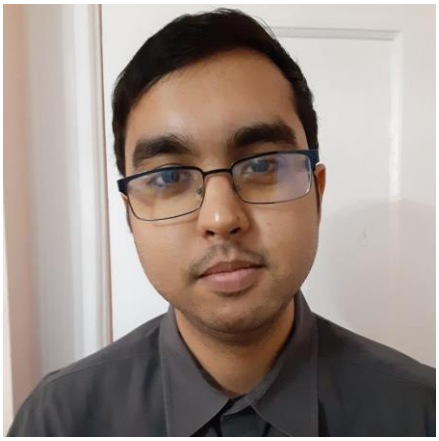
https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/round-1/spec-files/sqisign-spec-web.pdf



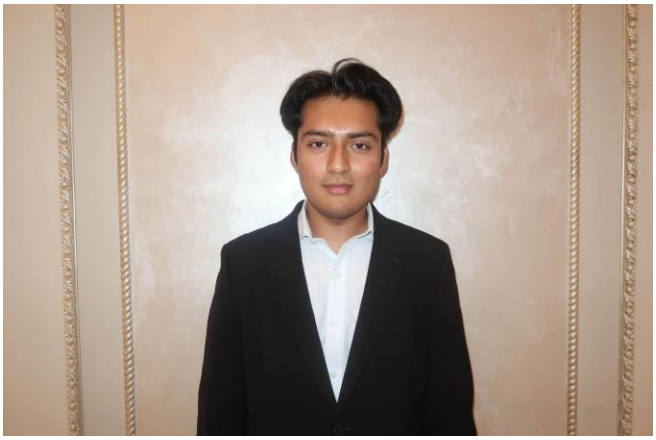
Matt Withee



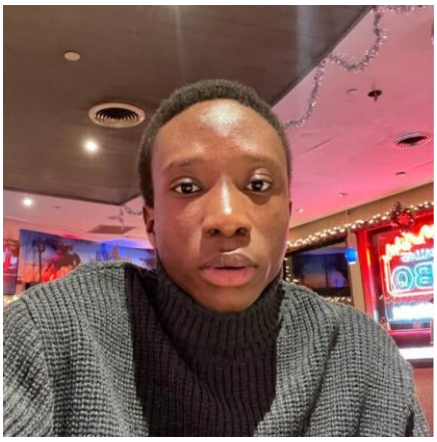
Anjiya Shrestha



Suresh Yhap



Abdul Mutallif



Osasere Imade

Chapter 1 – Introduction

As the promise of quantum computing draws closer and closer to reality, the safeguards in place today allowing society to protect itself in the digital realm are clearly becoming less sufficient. The rapidly emerging field of post-quantum cryptography has been building on the classical concepts and algorithms already in use while finding innovative new ways to defend against the threat of cyberattacks. Classical algorithms such as digital signature schemes are highly effective when used for defending against classical computers, but the computational power from a quantum attack would leave a huge amount of vulnerability, so innovations are required to prepare for the future. One such notable innovation in post-quantum cryptography consists of the development of an isogeny-based digital signature scheme. There are a few types of isogeny-based algorithms, but the one that will be discussed here is something called SQI_{SIGN}, which stands for Short Quaternion Isogeny-based Signature, a scheme where the digital signature is generated using isogenies between supersingular elliptic curves. This technique is gaining a lot of attention due to its efficiency in securing data while maintaining a compact signature size.

First, a brief review of digital signatures. A digital signature is a cryptography mechanism with three major security functions: authentication, integrity of information, and non-repudiation.

1. Authentication refers to the verification of the identity of the sender of information, achieved through the use of public key cryptography where the sender 'signs' a message with their private key and the message is then verified by the receiver using the sender's public key.
2. The second function of a digital signature, integrity, refers to the process of ensuring that the message has not been altered in any way by an 'in-between' party.
3. Non-repudiation, the third security function of a digital signature, is a way to prove that someone sent the message, essentially ensuring that someone that sends a message can't deny it later because they are the one that had the private key. These functions are the cornerstone of secure digital communication.

A digital signature is composed of two important mechanisms, a private key and a public key, both generated by the sender of the message, with the public key available for anyone that wants to view it.

Traditionally, the private and public keys are generated with algorithms such as Rivest-Shamir-Adleman (RSA) or Elliptic Curve Cryptography (ECC). In RSA, the private key consists of two large prime numbers and the sender encrypts either the message or, more commonly, a hash (summary) of the message. The public key has a mathematical relationship to both these prime numbers. Because factoring very large prime numbers is unrealistic for a modern computer, this algorithm ensures that the encryption remains secure. In ECC, the private key is a randomly chosen number and the public key is created by solving the Elliptic Curve Discrete Logarithm Problem to generate a number. The public key in ECC is linked to the private key. While ECC offers smaller key sizes than RSA (thereby making it a more efficient algorithm), it's still vulnerable to a quantum attack with something like Shor's algorithm able to solve the discrete logarithm problem at the center of its security. So, as technical advances in quantum computing become closer to reality, there is a need for algorithms that are quantum resistant.

One promising approach to a quantum resistant cryptography scheme comes in the form of what is called isogeny-based cryptography. An isogeny is a special function (essentially a map) that connects elliptic curves while preserving their group structure. Multiple points from the first curve can map to a single point on the target curve depending on the degree of the isogeny, but an isogeny must be surjective. Isogenies are useful in cryptography because computing them between elliptic curves can be very complicated, referred to as the Isogeny Problem. More specifically, many isogeny-based schemes (such as SQI_{SIGN}) rely on the difficulty of computing isogenies between *supersingular* elliptic curves, a problem that is very complex for even a quantum computer. And while it is achievable, reversing the process (finding the isogeny when you're only given the curves) is deemed virtually impossible, which is why it is deemed to be quantum-resistant.

There are many types of isogeny-based cryptography schemes, including Supersingular Isogeny Diffie-Hellman (a key exchange protocol that builds upon classical Diffie-Hellman, but incorporating points on the elliptic curves to compute a shared secret) and Supersingular Isogeny Key Encapsulation (a key encapsulation scheme that actually builds on SIDH by using a key encapsulation mechanism). These algorithms, like most isogeny-based algorithms, have small key sizes when compared to other quantum-resistant algorithms, and, at least in theory, they should be secure and efficient. However, both have been broken already by modern attacks that have shown weaknesses in their general structures.

While these algorithms have their own merits and shortcomings, this paper will discuss in detail a robust and promising new algorithm called SQI_{SIGN}, a digital signature scheme which relies on what's known as supersingular elliptic curves, which are elliptic curves with a special property where their associated endomorphic rings are a quaternion algebra. Quaternion algebra will be broken down further later, but for now it can be described as a four-dimensional, non-commutative algebra that has more complex quadratic fields, making it more complex to be able to compute and resistant to attack, an important feature for a quantum-resistant algorithm. The private key is generated by randomly choosing one of these isogenies. A common analogy is to imagine standing at the beginning of a maze where you can see the, but you don't know the exact path to take through the maze until you start walking it. The path that you walk to finish the maze is essentially your private key. The public key would be a map that you made that shows you found the exit but doesn't show which path you walked to get there. What's happened is the isogeny has been applied to a target curve to generate the public key.

As expected, there are pros and cons to any digital signature scheme and SQIsign is no exception. A benefit to this scheme is that researchers already know the complexity of the attacks that can be made against it, making the level of security adjustment very straightforward and theoretically keeping the algorithm safe while maintaining efficiency. Another important feature (indeed, what could be considered the defining feature) of SQI_{SIGN} is that the keys and signature are quite small when compared to other similar algorithms, making it highly efficient not only in terms of storage but also time it takes to send the message. However, while it is currently accepted that the endomorphism ring problem (the mathematical problem an attacker needs to solve to break the encryption) is extremely complex to solve, consideration must be given that someone could find a shortcut to solving it, making SQI_{SIGN} obsolete.

When looked at with future demands in mind, it becomes increasingly obvious that SQI_{SIGN}, with its smaller key and signature sizes, is one of the most promising and exciting fields in post-quantum cryptography. In this paper, the mathematical properties of SQI_{SIGN} will be examined in greater detail, including the generation of the keys, the process of signing and verification, as well as an analysis of the performance and security of the algorithm, while also discussing the potential challenges and shortcomings.

Chapter 2 - Basic Operations

The secureness of the SQI_{SIGN} Digital Signature scheme involves the difficulty of finding isogenies between supersingular elliptic curves that are defined over finite fields.

2.1 - Finite Fields

To define a field, we can build up from the essential element of algebraic structures, the set. A set is an unordered collection of distinct elements. If you add a binary operation on the elements and have the property of closure (applying the binary operation on two elements of the set results in an element of the same set) you end up with a magma. If the binary operation is associative (the order in which you evaluate the binary operations when there is a chain of the operation i.e. the placement of parentheses doesn't matter) then you form a semigroup. If the set has an identity element (applying the binary operation to any element along with the identity yields that same original element), the result is a monoid. Adding the existence of an inverse for every element (applying the binary operation to any element and its inverse yields the identity) forms a group. Finally, if you add commutativity (you can arrange the elements in any order when applying the binary operation to two elements) your result is an abelian group.

A field is a set enhanced with two binary operations (called addition and multiplication) such that the set paired with each of the two binary operations individually each form an abelian group (but for multiplication we exclude the additive identity, known as 0, because it has no multiplicative inverse in the set). Also, distributivity of multiplication across addition should hold ($a(b + c) = ab + ac$). An example of a field is the rational numbers $\mathbb{Q}$ equipped with the usual notions of addition, subtraction (addition with the inverse), multiplication, and division (multiplication with the inverse). A finite field (aka a Galois field) is a field with finite order (the number of elements or cardinality of the underlying set is finite).

The finite fields considered in this paper are ones of prime order, F_p, or the square of a single prime order, F_{p^2} where $p \equiv 3\ (mod\ 4)$. All finite fields are of prime power (a power of a single prime) order. The characteristic of a field is the minimum positive number of times you must add the multiplicative identity element to get the additive identity. An illustrative example is the field ($\mathbb{Z}_p$, +, ·) where the operations are done modulo p. The characteristic of this field is p because adding 1 p times yields 0 (because it equals p which is congruent to 0 mod p). The characteristic of a field F_q where $q = p^r$ is p.

A quadratic residue is the remainder when a perfect square is reduced modulo p (i.e. it is congruent to a square). To test if an element of the field F_p = ($\mathbb{Z}_p$, +, ·) is a perfect square, that is, there is an element b such that $b^2 \equiv a\ (mod\ p)$, then raising both sides of the congruence by $(p - 1) / 2$ yields $b^{p-1} \equiv a^{(p-1)/2}\ (mod\ p)$. By Fermat's little theorem which states that $c^{p-1} \equiv 1\ mod\ p$ where c and p are relatively prime means that $a^{(p-1)/2} \equiv 1\ mod\ p$. If this is not true, then a wasn't a perfect square to begin with. In the special case where $p \equiv 3\ (mod\ 4)$, the positive square root is given by $\sqrt{a} = a^{(p+1)/4}$. A verifying example is if p = 7, then $\sqrt{a} =$

$a^{(7+1)/4} = a^2$. Testing if $a = 2$ is a square and if so what the square root is follows. $2^{(7-1)/2} \equiv 8 \equiv 1\ (mod\ 7)$ so 2 is a square mod 7. $\sqrt{2} \equiv 2^{(7+1)/4} \equiv 4\ (mod\ 7)$. Indeed, $4^2 \equiv 16 \equiv 2\ (mod\ 7)$. A natural ordering of F_p's elements is considered (0, then 1, then … $p - 1$).

If F_p was analogous to the real integers, F_{p^2} is analogous to the complex integers. The imaginary unit, i , is the root of the equation $i^2 + 1 = 0$. The multiplicative inverse of an element, denoted $1/(a + bi)$ can be written with a "real" denominator by multiplying the numerator and denominator by the complex conjugate, $a - bi$. A lexicographical ordering of F_{p^2}(based on the real parts and then based on imaginary parts if the real parts are equal) is defined.

2.2 - Elliptic Curves

The class of curve we are interested in for SQI_{SIGN} are called elliptic curves. In particular, we are interested in so-called Montgomery curves over the field F_q of the form:

$By^2 = x^3 + Ax^2 + x$ where A and B are elements of the field that satisfy $B(A^2 - 4) \neq 0$. In addition, we include a "point at infinity", ∞. Two curves are isomorphic (there is a bijective mapping between them) if the mapping is of the form: $(x, y) \rightarrow (D(x + R), Cy)$ i.e. there is only shifting and scaling performed. Two elliptic curves are quadratic twists of one another if $C = \sqrt{B/B'}$ where B' is the leading coefficient on the y^2 term of the first curve and are isomorphic if B/B' is a perfect square in F_q.

If N, the number of solutions to the elliptic curve, is congruent to: $1\ mod\ char(F_q)$, then the elliptic curve is called supersingular. We are concerned with the field F_{p^2} where $p \equiv 3\ (mod\ 4)$ so for instance if $p = 7$, the curve is supersingular if it has 1, 8, 15, … many solutions. With B = 1, a supersingular curve has precisely $(p + 1)^2$ points. So for $p = 7$, there are 64 points (verifying this: $64 \equiv 1\ (mod\ 7)$ as expected).

Algorithm 1 (pg. 8) takes in an A value (take B = 1 here for now) and outputs an A' (which corresponds to a curve isomorphic to the curve corresponding to A) and the isomorphic mapping itself.

A very interesting fact is that we can define an addition operation that when paired with the set of points on the Montgomery curve form an abelian group! For an elliptic curve, if a line intersects it at two points, then it must intersect it at a third point given that tangent points are counted twice and the aforementioned "point at infinity" is the third point for vertical lines of intersection.

Addition can be defined as follows. The "point at infinity" serves as the additive identity so if P is a point on the curve, $P + \infty = \infty + P = P$. Now to add two general points on the curve, draw a line of intersection through the two points, $P_1$and P_2. The third point of intersection, P_3, is defined as -$(P_1 + P_2)$. So the three points, P_1, P_2, -$(P_1 + P_2)$ add to "zero" which here is the additive identity, ∞. This summing to identity will always be true. Reflecting this point over the x-axis yields the additive inverse $(P_1 + P_2)$, the desired sum (this is because

a vertical line through -$(P_1 + P_2)$ and $(P_1 + P_2)$ also intersect the point at infinity, ∞, and these three points should sum to "zero" as they do: -$(P_1 + P_2)$ + $(P_1 + P_2)$ + ∞ = ∞ again because the point at infinity is the additive identity). The addition rule is closed because it always results in a point on the curve. The addition rule is commutative because the order of points of intersection we add doesn't matter. Associativity turns out to also be true. Because of these properties, the points on the Montgomery curve with the defined addition operation truly do form an abelian group.

Point doubling can be achieved by drawing the tangent line to the curve at the desired point to be doubled, finding the third point of intersection (since the tangent point counts as two points), and reflecting as before. Note that by this definition a vertical tangent yield $P = -P$ (because this occurs on the x-axis) and $P + -P + \infty = \infty$ (by the addition law) so $[2]P = \infty$ (because $[2]P + \infty = [2]P$ since ∞ is the identity point). General scalar multiplication can be achieved with repeated addition, that is, summing copies of the point: $[k]P = P + P + \ldots + P$ k times, using doubling and addition as defined above. The order of a point is the smallest positive integer m such that $[m]P = \infty$. The geometric definition of addition outlined here can be translated to a coordinate-based algebraic one using the formulas on (pg. 9).

We consider a subgroup of an elliptic curve E, defined over F_{p^2} and integer m, called the m-torsion subgroup denoted $E[m]$ that consists of the points in E that yield $[m]P = \infty$. This means it consists of all points each with the property that when you add m copies of the point using the line intersection and reflection method previously mentioned, you end up on a vertical line, that is, the sum is ∞. If m^2 divides the number of points on the curve, then $E[m]$ is isomorphic to $Z_m \times Z_m$ so it consists of m^2 points. There exist non-unique points R and S (that is, possibly multiple pairs exist) in $E[m]$ that generate the group $E[m]$. (R, S) is called a basis for $E[m]$. Algorithm 3 on pg. 10 essentially finds R and calls Algorithm 2 on pg. 10 which finds S and returns the basis (R, S).

Since we are working with repeated applications of an abelian group's operator, we can naturally define a discrete logarithm problem. That is, given a point P and the point $[k]P$ (with scalar multiplication as defined above), extract the value of k, the number of times the addition operator was applied. You can imagine the difficulty of this because as defined above, the sum of two points on an elliptic curve seem to have little relation to what the initial points were (so repeatedly applying this is like bouncing around the elliptic curve). All you have is the initial point (P) and the final point ($[k]P$) and you need to find out how many times, k, the operator was applied. For points of large prime order (that is, they bounce around a lot before reaching the "steady state" of ∞ through a vertical line), the DLP is believed to be difficult for classical computers because you can basically only do an exhaustive search (try all possible k).

However, for points of smooth order (that is, the order is a composite number with small prime factors), the problem is efficiently solvable. For a particular m-torsion subgroup where m is a power of 2 or 3 (therefore the points in this subgroup $E[m]$ are of smooth order) where a basis of this group (R, S) is known, any point on the elliptic curve in $E[m]$ can be represented as a "linear combination" of the basis points, that is, $P = [a]R + [b]S$. So, such points are characterized by two integers, a and b, in a way similar to how vectors are described when the basis is implicitly understood. An algorithm based on the Pohlig-Hellman algorithm (which

solves the DLP for a finite abelian group with a smooth number of elements) is used to find a and b.

This analysis of supersingular elliptic curves defined over finite fields gives the background required to understand isogenies between elliptic curves and eventually the correspondence between quaternion ideals and isogenies.

2.3 – Isogenies

Remember, an isogeny is a special type of function that connects two elliptic curves in a way that preserves their algebraic structure. Given two elliptic curves E_1 and E_2 over a finite field F_q, an isogeny φ: $E_1 \rightarrow E_2$ is a non-constant map defined by rational functions (ratios of polynomials) with coefficients in F_q. This map satisfies $\varphi(\infty) = \infty$, where ∞ represents the point at infinity on the elliptic curve. An isogeny is also a group homomorphism which means it preserves the group operation (point addition) on the elliptic curves and when two curves are connected by an isogeny, they are said to be isogenous. Interestingly, two curves over F_q are isogenous if and only if they have the same number of F_q rational points, denoted f $\#E_1(F_q) = \#E_2(F_q)$.

The kernel of an isogeny φ, denoted as ker(ϕ), includes all points on E_1 that get mapped to the point at infinity on E_2: ker(φ) = {P ∈ E_1| $\varphi(P) = \infty$ }. If you know a subgroup G of E_1 with size N, there is a unique isogeny φ of degree N with the subgroup G as its kernel, ker(φ) = G. Furthermore, For every isogeny φ: $E_1 \rightarrow E_2$, there exists a dual isogeny φ':$E_2 \rightarrow E_1$ of the same degree and every isogeny has a dual isogeny φ':$E_2 \rightarrow E_1$, which reverses the direction of the original map. The composition of the isogeny and its dual gives a scalar multiplication by N on the original curve: $\varphi' \circ \varphi$ = [N].

An elliptic curve can be represented using a specific type of equation called a Weierstrass equation and the general form look like: $y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$ where x and y are the variables that represent the coordinates of points on the curve and the numbers a_1,a_2,a_3,a_4,a_6 are constants (called coefficients) that come from the field and these coefficients define the specific shape and properties of the curve. One of the methods for computing isogenies is through Vélu's formulas which allow us to calculate the specific map between two elliptic curves, given the kernel of the isogeny. If we are given a point Q on E_1 that generate a subgroup of order N, Vélu's formulas provide us with the rational functions that define the isogeny φ and can be written as φ ((x, y)) = (f(x), y · g(x)).

f(x) and g(x) are ratios of polynomials, and their degrees are related to the degree of the isogeny and the degree of the isogeny related to the size of the kernel. Vélu's formulas are only practical to compute for relatively smaller values of N, so when N becomes large, especially if N is a composite number, we decompose the isogeny into smaller isogenies, each with a smaller prime degree. For example, if N has a prime factorization: N = $l_1^{e1} \cdot l_2^{e2}$... where l_1 ,l_2 ,… are primes, we can compute the isogeny as a composition of isogenies of degree l_1 ,l_2 ,… This method is feasible as long as the prime factors of N are sufficiently small.

2.3.1 - 2-Isogenies

A 2-isogeny is a specific type of isogeny with degree 2 which means that it maps points between two elliptic curves in such a way that the kernel of the isogeny contains exactly two

points (including the identity point). Given an elliptic curve $E_{A,B}$ and a point Q of order 2 on this curve, we can generate a 2-isogeny φ: $E_{A,B} \rightarrow E_{A',B'}$. The point Q generates the kernel of the isogeny, which is the set of points that are mapped to the point at infinity on the target curve. Consider a point Q of order 2 on an elliptic curve $E_{A,B}$. There are two cases: Case 1, When Q=(0,0), The isogeny φ and the new curves $E_{A',B'}$ are defined using specific formulas involving A and B. The formulas calculate φ (x,y) and update the curve coefficients to get (A′,B′). In Case 2, When Q=(x_Q,0) with $x_Q \neq 0$, different formulas are used to compute φ (x,y) and (A′,B′), involving x_Q. In both cases, the point (0,0) on $E_{A',B'}$ lies in the kernel of the dual isogeny φ, which is needed for chaining isogenies. The dual isogeny is the inverse map of the isogeny, and its kernel consists of points that are mapped to the point at infinity.

2.3.2 - 4-isogenies

A 4-isogeny is a specific type of isogeny with degree 4, meaning that the kernel of the isogeny contains exactly four points, including the identity point. 4-isogenies are more complex than 2-isogenies but for efficiency purposes, some implementations may prefer to use 4-isogeny formulas more because they can be equivalent to combining two 2-isogenies formulas in a sequence. On page 12, Case 1: Q=(1,y_Q), then 2[Q]=(0,0), in this case, Q has the x-coordinate 1, and its second multiple is (0,0). This formula computes the new x- and y-coordinates of points and defines the image curve's parameters A′ and B′. The transformation is based on the elliptic curve's initial parameters and accounts for the 4-isogeny structure. The formula effectively transforms points on the original curve $E_{A,B}$ into points on the new curve $E_{A',B'}$, which has different parameters for A′ but the same B.On pg 12, Case 2: Q=(−1,y_Q),in this case, Q has the x-coordinate of −1, and this is similar to the first case, but the sign changes in the formulas so, we follow a similar process to derive the isogeny. This transformation also adjusts the point coordinates and the curve's parameters but in this case, the biggest difference here is that the formulas take into account the different x-coordinate of Q=(−1,y_Q).On pg 12, Case 3: Q=(x_Q,y_Q) with $x_Q \neq \pm 1$. In this third case, the point Q has coordinates (x_Q,y_Q) and x_Q is neither 1 nor -1. The new curve parameters in this case are (A′,B′)=(2−4xQ2,B)(A', B') = (A′,B′)=(2 $-4x_Q^2 x$,B). This formula is the most general and applies to points that are not at specific locations like 1 or −1 but it still serves the same purpose which is mapping points from one elliptic curve to another while preserving the group structure.

2.3.3 - Other odd-degree isogenies using Vélu's formulas

Odd-degree isogenies are those with a kernel of odd size. Let ℓ be an odd prime, and let Q be a point on the elliptic curve $E_{A,B}$ of order ℓ. This point Q generates the kernel of the isogeny φ:$E_{A,B} \rightarrow E_{A',B'}$, where $E_{A',B'}$ is the codomain curve. The main idea here is that the isogeny is determined by its kernel, and knowing the point Q allows us to compute the isogeny more clearly and to compute this isogeny, we define the polynomial h_S(x), which encodes the structure of the kernel. The polynomial is defined as the product, h_S(x) = s∈S Π (x − $x_{[s]Q}$), where S={1,2,…,(ℓ−1)/2}, and $x_{[s]Q}$ is the x-coordinate of the point $[s]Q$, which is the scalar multiple of Q and this polynomial collects all the important points in the kernel of the isogeny.The parameter A′ defines the new curve given by A′=2·1+d/1−d, where d, a value computed from the original elliptic curve parameter A is computed as d= (A-2/A+2)^l · (h_S(1)/h_S(-1))^8. The parameter B remains unchanged so , $B'=B$ and it allow us to compute the new elliptic curve $E_{A',B'}$ that is isogenous to the original curve $E_{A',B'}$.

2.3.4 - Odd-degree isogenies using √élu

The square root Vélu algorithm, also called the √élu algorithm, computes isogenies of elliptic curves in time O~(ℓ) rather than O(ℓ), where ℓ is the degree. The main idea is to reindex the points in the kernel subgroup in a baby-step-giant-step manner, the baby-step giant-step algorithm computes discrete logarithms in a group of order q using O(√q) group operations and this algorithm breaks down the computation into smaller steps, making a faster evaluation of the kernel polynomial h_S(x). Instead of S={1,2,…,(ℓ−1)/2}, we use S={1,3,5,…,ℓ−2} redefining the index Sets I,J,K then continue by defining an index system (I,J) for S, where, I = {2m(2i+1) | 0 ≤ i ≤ m′} and J = {2j+1 | 0 ≤ j < m}. m is defined as m = √(ℓ−1/2) and m′ = 0 if m = 0 , if not then m′ = (ℓ+1)/4m and K = S\(I±J) , where I ± J = {i+j, i-j | i ∈ I, $i \in J$ }. The sizes of these sets are all in O(√ℓ), which makes the algorithm better.

Biquadratic polynomials are polynomials of degree 4 that can be written in the general form P(x)=ax^4+bx^2+c . The term ax^4 represents the highest degree (degree 4), and there are no x^3or x^1 terms, which differs the biquadratic polynomials from general quartic polynomials. To evaluating $h_S(\alpha)$ in Steps First, defining Biquadratic Polynomials F_0,F_1,F_2: F_0 (x_1,x_2)=(x_1−x_2)^2 , F_1 (x_1,x_2) = 2((x_1x_2+1)(x_1+x_2) + 2ax_1x_2), F_2 (x_1,x_2) = (x_1x_2-1). On pg 13, for polynomial h_I(x), this polynomial contains roots at the x-coordinates of multiples of Q in set I. For D_j(x), squares the differences, focusing roots at x = $x_{[j]Q}$. For the Resultant Δ I,J , it measures the common factors between h_I(x) and D_j(x) and results in an element of F_q. For E_j(x), this incorporates the input α and the polynomials F_0,F_1,F_2. For Resultant R, ΔI,J is an element of F_Q and contains information about α. For h_K and output $h_S(\alpha)$, this gives the value of $h_S(\alpha)$ needed for the isogeny computation.

When using efficient polynomial arithmetic, computing polynomials using product trees for efficient polynomial multiplication and remainder trees for multi-point evaluation speeds up polynomial and resultant computations. Product trees are used to multiply a series of polynomials efficiently and the polynomials are grouped in pairs and multiplied in parallel, reducing the number of steps. Remainder trees are used for multi-point evaluation which is a method that allows us to evaluate a polynomial at multiple points at once by breaking the evaluation into smaller steps which speeds up the calculation.

In SQI_{SIGN}, isogenies and methods like Vélu's and √élu's algorithms help create secure and practical post-quantum signatures. They rely on the difficulty of solving isogeny problems on supersingular elliptic curves, providing strong protection against quantum attacks.

2.4 – Quaternions and Ideals

A quaternion is a mathematical object, essentially a set of four numbers, used to represent rotations in three-dimensional space. Additionally, quaternion defines the quotient of two vectors in a three-dimensional space. A quaternion algebra over a field K is a central simple algebra of dimension 4 over K. It is defined by two parameters $a, b \in K$ and it is thus denoted as $B = (\frac{a,b}{K}) = K + Ki + Kj + Kk$ where the multiplication rules are:

$i^2 = a, j^2 = b, ij = k, ji = -k$. Quaternion algebras are generalizations of Hamilton's quaternions. There are four-dimensional spaces generated by four elements $\{1, i, j, k\}$ and non-commutative algebras with the multiplication rules defined previously.

2.4.3.1 - Finding a Short Basis

The short basis algorithm involves finding a short basis for a lattice with respect to $N_q(u) = u_0^2 + qu_1^2$. Simply put, "the goal of lattice basis reduction is to find a basis with short, nearly orthogonal (perpendicular) vectors when given an integer lattice basis as input" (Wikipedia). The input for the algorithm includes two vectors b_0 and b_1, which form an integral lattice basis, along with a positive integer q. The output should be two vectors, β_0 and β_1, that form a short basis for the same lattice. The input must also meet these requirements: $N_q\ (\beta_1) \leq N_q\ (\beta_0)$. Now that we know what to input, let's talk about the process. The process starts by initializing and swapping the basis vectors to maintain the condition $N_q\ (\beta_1) \leq N_q\ (\beta_0)$. The algorithm then enters a loop, where it computes a value r that is rounded by dividing $N_q(\beta_{1)}$. The goal of the loop is to reduce the length of the vectors. The loop continues It updates the gamma vector by subtracting rβ_1 from β_0, and if the norm of gamma is smaller than β_1, the vectors are updated. Finally, the loop terminates, and the short basis vectors β_0 and β_1 are returned.

2.4.3.2 - Finding the Closest Vector

After computing a reduced basis, the next step is to find a vector in the lattice that is closest to the target vector t. This is accomplished using Babai's nearest plane algorithm. The nearest plane algorithm was developed by L. Babai in 1986, and it "obtains a 2(2/√3) ^n approximation ratio, where n is the rank of the lattice" (Regev). In other words, the algorithm projects the target vector onto the reduced basis, and then adjusts the result to produce a vector c that is close to t with respect to the norm N_q.

2.4.3.3 - Enumerating Close Vectors

The final step is to enumerate short vectors in the lattice that are close to the target vector t, using the Fincke-Pohst algorithm. In other words, "the running time of enumeration algorithms greatly depends on the quality of the input lattice basis. So, suitably preprocessing the input lattice using a basis reduction algorithm is an essential part of lattice enumeration methods" (Lattice Cryptography). Furthermore, the algorithm takes as input a lattice L, a target vector t, and an initial close vector c. It then iterates through possible values for the coordinates x and y, generating vectors that are within bound B from the target vector. By updating both x and y within nested loops, the algorithm yields vectors that satisfy the distance condition. This process continues until either the specified number of tries is reached, or all close vectors are found.

2.4.4.1 - Basic Quaternion Arithmetic

Quaternions are generalizations of complex numbers, which are represented by the expression: α = a + bi +cj + dk / r. The basic arithmetic operations like addition and multiplication are computed by reducing common denominators where necessary. Specifically, multiplication follows the axioms: i^2 = -1, j^2 = -p, ij = -ji = k. We also have the conjugate operation where α' = a- bi - cj -dk / r. There's also the reduced trace that simplifies to: tr(α) = 2a / r. Finally, we have the reduced norm: nrd(α) = a^2 + b^2 + p (c^2 + d^2) / r^2.

2.4.4.2 - Lattices

A lattice is a collection of quaternions that form a grid-like structure. Additionally, lattices are defined by a basis of quaternions that are represented as columns of a matrix. You can also perform operations on lattices such as:

Equality: Whether two transactions are similar by comparing their HNF.

Union and Intersection: The union of two lattices is formed by merging their initial matrices into HNF. The segmentation of the dual network is obtained by computing the HNF.

Multiplication: In lattice multiplication, the base matrices are multiplied by the corresponding quaternary algebra generators.

Containment: This function ensures that an element is contained in the network by solving a system of linear equations.

Index: The index of one network in another is the mean value of their basis matrices.

Right Transporter: The right transporter of a lattice is a set of objects that, when connected by a grid, still belong to another grid. It is one of the most complex operations on networks. (SQI Sign)

2.4.5 - Quaternion orders and ideals

Quaternaries play an important role in algebra, and their structure includes some small orders and ideals. An order is a special type of lattice that forms a subring. Elements of an order O are integral because their norm and trace are integers. An order is maximal if it isn't contained in any larger order.

An ideal is a sublattice of an order. The left order of an ideal I is the set of elements that, when multiplied with I, still belong to I. Similarly, the right order is defined in the same way for the right side. A connecting ideal links two orders O_L and O_R.

The norm of an ideal is the greatest common divisor (gcd) of the norms of its elements. This norm is an integer and can be computed from both the left and right orders. Any ideal can be expressed using the elements of its left order, and when two ideals are multiplied together, the result is another ideal. Ideals are multiplicative, and their norms follow this rule too. Two orders O_1 and O_2 are equivalent if there's a quaternion element that links them. In addition, two left ideals I and J are equivalent if they can be scaled into each other by multiplication. When this happens, the left and right orders of both ideals are also equal.

2.4.5.1 - Basic Operations on Ideals

Now that we know about the importance of orders and ideals, it is also important to note that various operations can be performed on them. Simply put, ideals are used in lattices to perform various operations like equality, union, intersection, and multiplication.

Left and Right Orders: The left and right orders of an ideal determine how it relates to itself. These are the sets of elements that, when multiplied with the ideal, still belong to the ideal.

Isomorphism of Ideals: Two left ideals are isomorphic if there is a quaternion β that scales one ideal into the other, like I = Jβ. This can be calculated using the transporter of the ideals and checking their norms.

Connecting Ideals: To connect two orders O_L and O_R, we compute a connecting ideal using the formula: I = (O_L + O_R)N, where N is the norm of their intersection.

Pullback and Pushforward Ideals: If two ideals have coprime norms, the pullback combines them into a new ideal, while the pushforward reduces them using intersection and inversion.

2.4.5.2 - Finding Equivalent Ideals of Small Norm

Sometimes, we need to find an equivalent ideal with a smaller norm, particularly when working with large ideals. When working with SQI_{SIGN}, we often need to find an equivalent ideal J that has the same structure as a given ideal I but with a different, smaller norm. This is achieved using the surjection: χI (α) = Iα' / nrd(I), where α is a quaternion, and the set of ideals J is equivalent to I. To do this, we use an algorithm, which constructs a new ideal J = χ(β) where β is formed from the reduced basis of I. The process repeats up to a set number of times to check if the norm of J is prime.

2.5 - Solving Norm Equations

A norm is a function from a real or complex vector space to the non-negative real numbers that behave in a certain way like the distance from the origin. Various kinds of norms include the equivalent norm, the absolute value norm, and the Euclidean norm.

The basis for this is that solving norm equations in quaternion orders has applications in cryptography. There are three KLPT-based procedures for solving norm equations which are ***KeyGenKLPT*** which is used within a key generation, ***SpecialEichlerNorm*** which solves a norm equation O and ***SigningKLPT*** which solves a norm equation in the left O-ideal which O represents quaternion order. The left O-ideal is a non-zero subset $I \subseteq B \; such \; that \; OI \subseteq I \; where \; O \; is \; the \; maximal \; order$.

KeyGenKLPT Algorithm: Given a left O-ideal I in a quaternion algebra Bp,∞, the KLPT algorithm finds an equivalent ideal of a smaller norm

Input: A left ideal $I \subseteq O$ where O is a maximal order in the quaternion algebra Bp,∞.

Process: Find an equivalent ideal J that J has a smaller norm than I. The algorithm uses the reduction theory of quaternions to efficiently find J by solving norm equations. Ensure that J corresponds to an isogeny with a reasonable degree that makes further cryptographic computations feasible. Output: An equivalent ideal J of a smaller norm.

The KLPT algorithm helps in minimizing computational overhead in constructing and verifying isogenies, making isogeny-based cryptographic schemes more practical.

SpecialEichlerNorm Algorithm explains the process of solving norm equations efficiently in any maximal order O the idea behind the algorithm is to find $O1 \cong O\ such\ that\ I(O0, O1)$ satisfies the constraints on the input of EichlerNorm so we can solve in $(O0 \cap O_1)$ and then transport the output O using the isomorphism between O and O_1.

The SigningKLPT algorithm is done by the generalized KLPT algorithm. The output of a SigningKLPT is of a constant degree. Some randomization is also included to ensure a good distribution.

The process of solving norm equations in quaternion orders and ideals can be seen in various algorithms which include Cornacchia's algorithm, representing integers by special extremal order, and reduction to linear systems.

2.5.1.1 – Cornacchia's Algorithm

Cornacchia's algorithm allows us to efficiently solve norm equations of the form $x^2 + ny^2 = m$. The algorithm inputs m which is an integer and gives a true or false value if a solution was found or not. The algorithm goes through a sequence of numbers to make sure m is a prime if it is a prime, Cornacchia's algorithm would run and return the value of x and y when found.

2.5.1.2 - Representing integers by special extremal order

This is a follow-up from Cornacchia's algorithm. The idea of this algorithm is to find elements in a given norm. This process allows us to solve norm equations in the sub-order $R + jR \subseteq \vartheta_0$ where $R = Z[i]$ the quadratic substring of ϑ_0 whose norm for is given by $F(t, x, y, z) = x^2 + y^2 + p(z^2 + w^2)$. The general form is to sample z, w before using Cornacchia's to see if we can find x, y the following: $x^2 + y^2 = T - p(z^2 + w^2)$. The algorithm returns a boolean if a solution was found.

2.5.1.3 - Reduction to linear systems

This algorithm in this case can be found in EichlerModConstraint and FindLinearCombination. Both algorithms use linear algebra in Z / NZ to find a quaternion of a special form. In EichlerModConstraint, we consider quaternion elements $\alpha = x + yi + zj + wk$ where $[x, y, z, w] \in Z^4$ as integer column vectors while in FindLinearCombination, we compute a basis $\langle a_1, a_2, a_3, a_4 \rangle = O$ and compute vectors $[x, y, z, w] \in Z^4$.

The final step of solving the norm equation can be seen in the **Strong Approximation algorithm**. This algorithm solves norm equations inside $O0$. The algorithm finds the strong approximation mod $N\ in$ N of some $\mu_0 \in Oj_O$. This algorithm takes in an input of a prime number N, two values, $C, D \in Z$, the output is of the equation $\mu = \lambda\mu 0 + N\mu 1$ with $\mu_0 = j(C +$

$\omega d)\ where\ \mu_1 \in\ O0\ \ such\ that\ n(\mu) \in N$ which then returns a value of $\mu = \lambda j(C + D\omega) + N(t + \omega x + j(y + \omega z))$.

The essence of solving norm equations and how these algorithms and equations are essential in cryptography is to provide for integrity and non-repudiation of cryptographic systems like in SQI_{SIGN}. Integrity provides us with the ability to make sure a message sent is a message received where nothing in the message changes. Non-repudiation provides us with the ability to verify whoever sends a digitally signed message cannot be in denial of the message coming from the person who sent it. Solving norm equations provides us with the ability to ensure that messages can be digitally secure and encoded which enhances the security of digital communications.

Bibliography

Chavez-Saab, J., Santos, M. C.-R., De Feo, L., Eriksen, J. K., Hess, B., Kohel, D., Leroux, A., Longa, P., Meyer, M., Panny, L., Patranabis, S., Petit, C., Rodríguez Henríquez, F., Schaeffler, S., & Wesolowski, B. (2023, June 1). *SQIsign Algorithm Specifications and Supporting Documentation*. Version 1.0. SQIsign.org. https://sqisign.org

"Isogeny." *Wikipedia*, 10 Apr. 2023, en.wikipedia.org/wiki/Isogeny.

OpenAI. "ChatGPT ." *ChatGPT*, OpenAI, 2024, chatgpt.com/.

Pound, Mike. "What Are Digital Signatures? - Computerphile." *Www.youtube.com*, 2021, www.youtube.com/watch?v=s22eJ1eVLTU. Accessed 10 Oct. 2021.

PQShield. "PQC Scheme: SQISign." *YouTube*, 8 Nov. 2023, www.youtube.com/watch?v=9WfENxo82jg. Accessed 30 Sept. 2024.

Wikipedia Contributors. "Digital Signature." *Wikipedia*, Wikimedia Foundation, 4 June 2019, en.wikipedia.org/wiki/Digital_signature.

---. "Quaternion Algebra." *Wikipedia*, Wikimedia Foundation, 21 Feb. 2024, en.wikipedia.org/wiki/Quaternion_algebra. Accessed 30 Sept. 2024.

Jonathan Katz and Yehuda Lindell, *Introduction to Modern Cryptography* (3rd edition), (Chapman & Hall/CRC Cryptography and Network Security Series)(2020)

"Algebraic structure." *Wikipedia*, https://en.wikipedia.org/wiki/Algebraic_structure

"Field." *Wikipedia*, https://en.wikipedia.org/wiki/Field_(mathematics)

"Finite field." *Wikipedia*, https://en.wikipedia.org/wiki/Finite_field

"Elliptic curve." *Wikipedia*, https://en.wikipedia.org/wiki/Elliptic_curve

"Degree of a Polynomial." Wikipedia, 3 June 2024, en.wikipedia.org/wiki/Degree_of_a_polynomial#:~:text=Degree%204%20%E2%80%93%20quartic%20(or%2C.

Santos, Maria. "Maria Santos." PhD Student at UCL, Interested Particularly in Post-Quantum Cryptography and Applications., 7 Nov. 2020, www.mariascrs.com/2020/11/07/velus-formulas.html.

"Lattice Cryptography." *Enum*, cseweb.ucsd.edu/~daniele/LatticeLinks/Enum.html. Accessed 23 Sept. 2024.

"Lattice Reduction." *Wikipedia*, Wikimedia Foundation, 22 Jan. 2024, en.wikipedia.org/wiki/Lattice_reduction.

Regev, Oded. *Lecture 3 CVP Algorithm 1 the Nearest Plane Algorithm*, cims.nyu.edu/~regev/teaching/lattices_fall_2004/ln/cvp.pdf. Accessed 23 Sept. 2024.

"Norm". *Wikipedia,* https://en.wikipedia.org/wiki/Norm_(mathematics)

"Quaternion". *Wikipedia,* https://en.wikipedia.org/wiki/Quaternion

DRP, Turkey. Salim Erdem Kocak “Isogeny Based Cryptography and KLPT Algorithm”. *YouTube,* uploaded by DRP Turkey, 5 Sept. 2024, https://www.youtube.com/watch?v=vNc9svEBCf0&list=WL&index=1&t=751s

Leroux, Antonin. *Quaternion Algebras and Isogeny-Based Cryptography*. 6 Sept. 2022, https://www.lix.polytechnique.fr/Labo/Antonin.LEROUX/manuscrit_these.pdf