# Supersingular Elliptic Curve Cryptosystems

Henry Lambson

*Southern Methodist University*

hlambson@smu.edu

***Abstract*— With the potential of quantum computing quickly arriving, there is a need for newer and stronger cryptosystems that cannot be easily broken by these supercomputers. This need can be met with Supersingular Elliptic Curves, a particular type of subset of traditional elliptic curves. This paper will go into the details behind Supersingular Elliptic Curves and how they can be used for digital signatures, hash functions, and Diffie-Hellman schemes.**

## I. Introduction

Recent developments in quantum computing have spurred the cryptography community and government alike to take action in developing systems that are resilient to both traditional methods of attacking, and quantum computing. In 2016, the National Institute of Standards and Technology (NIST) produced a report on post-quantum cryptography that emphasized the need for experts to begin developing new cryptosystems [1]. The fear of NIST, and the cryptography community as a whole, is that a computer capable of implementing Shor's Algorithm, which is proposed to be capable of finding the prime factors of any integer in polynomial time [2]. Such an accomplishment would render existing cryptographic schemas to be obsolete, as the crux of their security would be broken.

This paper proposes that Supersingular Elliptic Curves (SEC) can be utilized to resist the capabilities of quantum computing, as their unique properties cause them to be much more difficult to work with and perform calculations on. As such, "the only known quantum algorithm for these problems has exponential complexity" [3]. So long as no other more efficient algorithms are found pertaining to SECs, the field shows promise in developing cryptosystems for a post-quantum world.

## II. Definitions and Properties of SECs

In order to implement SECs into cryptosystems, the unique properties they possess that make them so computationally expensive must be explored. These properties that differentiate SECs from regular elliptic curves are what allows the cryptosystems to be resilient to quantum attacks.

### A. Definition

There are many definitions as to what makes an elliptic curve Supersingular, for this paper's purpose, only two will be listed. An elliptic curve is defined to be Supersingular if the curve $E$ over a finite filed with characteristic $p$ contains no points of order $p$ with coordinates in the closure of field $K$. It is said that the curve $E$ is Supersingular if and only if the characteristic of $K$ is 2 or 3 [4].

### B. Endomorphisms

An endomorphism is a function that maps an object onto itself. In terms of elliptical curves, an endomorphism is a function that maps points on the curve to other points in a way that preserves the group structure. The set of endomorphisms of an elliptic curve, End(E), is known as the endomorphism ring. SECs have been proven to have unusually large endomorphism rings, one of the properties that make them viable for cryptosystems. SECs have endomorphism rings or rank 4, whereas regular elliptic curves only have rings of rank 1 or 2. These large rings are contributors to what makes SECs resilient to quantum attacks. It is much, much more difficult to compute and use the endomorphism rings of SECs, which is why they are viewed to be strong assets to SEC cryptosystems.

### C. Point Count

A key characteristic of SECs is their lower number of points compared to regular elliptic curves. For these curves, there is a tight bound on the number of points over a finite field $Fq$ such that the number of points is equal to the square root of $q$. This differs from regular elliptic curves, as they typically have approximately $q$ number of points. This is an important property, as when the point count of an elliptic curve is closer to the square root of the filed size, in the case of SECs, as opposed to closer to field size itself, the size of the DLP can be increased, making it much harder to solve [5].

### D. Isogenies

An isogeny is a type of morphism between two algebraic varieties that preserves the group structure between the varieties. In elliptic curves, an isogeny is a map between two curves, meaning that points from one curve are mapped onto a second curve. Let E, E' be two elliptic curves over a finite field Fq. An isogeny $\varphi: E \rightarrow E'$ is a non-constant morphism from E to E′ that maps the neutral element to the neutral element [7]. From this definition, it can be seen that an endomorphism is purely just an isogeny from a curve to itself.

Both regular and SECs have many isogenies, however, the isogenies for SECs are much more complicated to work with. SEC isogenies are not uniformly distributed like they are in regular elliptic curves, meaning that certain isogenies occur more often than others. Because of this, creating cryptosystems utilizing isogenies in SECs is more difficult because the schemes rely on the assumption that all isogenies are equally likely. That being said, this issue with SEC isogenies increases the complexity of attacks that utilize isogenies to break the cryptosystem. Because of the smaller endomorphism ring that SECs possess, isogeny based schemes become more efficient [6].

### E. Torsion Points

Torsion points are points on an elliptic curve, supersingular or otherwise, that have finite order. A point P on a curve E defined over a field K is said to be a torsion point if there exists a positive integer n such that [n]P = O, where O is the identity element of the curve. If P is a torsion point of order n, then it lies on a cyclic subgroup of E, where if P is added to itself n times, the result would be O. The smallest integer n for which this is true is known as the order of the torsion point.

### F. J-Invariant

The j-invariant of an elliptic curve is an analytic function that is computed from the coefficients of the curve's standard form, or Weierstrass form, equation, which provides a way of classifying curves up to isomorphism. A curve of the form $y^2 = 4x^2 + g2x - g3$ where g2 and g3 are complex numbers, the j-invariant is calculated to be $j = 1728 * (4g2^3) / (4g2^3 + 27g3^2)$. Isomorphic curves are essentially the same curve but viewed from a different angle. Two elliptic curves can be said to be isomorphic if they have the same j-invariant.

### G. Kernel

The kernel of an elliptic curve is the set of points when added to any other point on the curve, the result is the identity element at infinity. The kernel of an isogeny is the set of points on the first curve that are mapped to the identity element of the second curve.

## III. Isogeny Based Digital Signature

Because of the nature of the isogenies of SECs, they have become strong candidates for designing cryptosystems that are resilient to quantum algorithms. One of these schemes is for digital signatures to verify information that is being sent. This scheme relies on the difficulty of finding and computing specific isogenies for a chosen SEC. Two of these hard problems will be explored further:

1) *Problem 1:* Let $p, \ell$ be distinct prime numbers. Let $E, E'$ be two SECs over $Fp^2$ with $\#E(Fp^2) = \#E'(Fp^2) = (p+1)^2$, chosen uniformly at random. Find $k \in N$ and an isogeny of degree $\ell^k$ from $E$ to $E'$ [7]. Note that the "#" symbol represents the cardinality of the elliptic curve, i.e., the number of points on that curve.

2) *Problem 2:* Let $p,\ell$ be distinct prime numbers. Let $E$ be an SEC over $Fp^2$, chosen uniformly at random. Find $k1, k2 \in N$, an SEC $E'$ over $Fp^2$, and two distinct isogenies of degrees $\ell^{k1}$ and $\ell^{k2}$, respectively, from $E$ to $E'$ [7].

These two problems rely on the difficulty of finding specific isogenies for a given SEC. Finding these isogenies has been proven to be sufficiently difficult even with proposed quantum algorithms. With these as the crux of a cryptosystem, we can design a digital signature verification algorithm. There are many more of these types of problems explored further in [7], however for the sake of this paper, only the two listed above will be used.

The signature schema utilizing these hard problems takes the form of public key digital signatures, in which the public key is a pair of SECs (E0,E1) and the private key is an isogeny $\varphi$: E0→E1 between the two curves. In order to prove knowledge of $\varphi$, the prover chooses a random isogeny $\psi$:E1→E2 and sends E2 to the verifier. The verifier will send back a bit b, either 0 or 1 to the prover. If b = 0, the prover will reveal $\psi$, if b = 1 the prover will reveal a third isogeny $\eta$:E0→E2. The verifier then checks if the response is correct, and this back and forth continues until the verifier is convinced that the prover knows an isogeny from E0 to E1. With either of the two problems as the crux of finding the isogeny from E0 to E1, this signature algorithm can be secure to quantum computers. The prover must know of the isogeny in order for them to be verified by the verifier, and because it is so difficult to find the specific isogeny chosen without prior knowledge of what it is, the verifier can be confident that the prover rightfully has the knowledge that they say they have.

To generate the public and private keys of this scheme, a random SEC is chosen, and a random isogeny of that curve it chosen. It must be noted that there are some SECs in which computing the endomorphism ring End(E) is much easier than most other curves, so care must be taken to avoid these specific curves for constructing this scheme.

Before the next step in the process, a key relationship between endomorphism rings and isogenies must be explored. End(E) can be computed using the isogeny $\varphi$:E→E′. This is possible because End(E) is equal to the image of the ring End(E′) under the map influenced by $\varphi$. This computation is by no means trivial, but it is workable in setting up this scheme. This is why it is integral to the scheme that the specific SECs that have easy to compute endomorphisms are not used. For this scheme to work, the only way that End(E) should be able to be found is by using the isogeny to do this computation.

Continuing on with the key generation, using the random isogeny chosen, End(E1) can be computed. This becomes the secret key, along with the isogeny $\varphi$: E0→E1. Under the assumption that none of the curves in which computing End(E1) is easy are being used, the secret key cannot be computed from the public key of E0 and E1.

This scheme relies on the fact that the prover must know the isogeny $\varphi$: E0→E1 in order to generate both the $\psi$:E1→E2 isogeny and the $\eta$:E0→E2 isogeny. If the prover does not know $\varphi$, then the verifier will be able to find out after testing a sufficient amount of times, as the prover will eventually fail to produce $\psi$ and $\eta$ that are correct. Figure 1 will show a summarized version of how this interaction works.

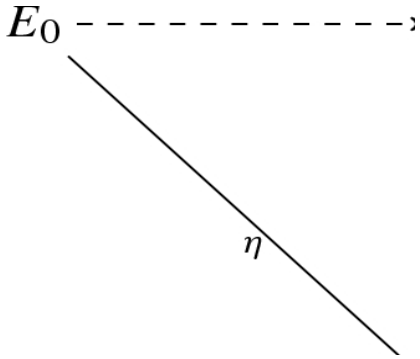


Fig. 1. Illustration of the relationship between E0, E1, and E2 using the isogenies φ, ψ and η [7]

Even if E0 and E1 are known, which they are since they belong to the public key, the isogeny from E0 to E1 must be known in order to generate End(E1). This can then be used to generate a unique isogeny from E0 to E2 that uses the order of End(E1).

One the verifier is satisfied that the prover knows φ: E0→E1, the interaction is concluded, and the prover has shown that the signature is indeed theirs.

## IV. Hashing Using Expander Graphs

Implementing the properties of SECs into a hashing algorithm requires the use of Ramanujan Graphs. These graphs are as set of SECs over $Fp^2$ with ℓ - isogenies, with ℓ being a prime that is distinct from p. By utilizing these graphs for a hash function, the collision resistance of the function follows from the hardness of computing isogenies between SECs, as described in problems 1 and 2 above. This graph is also known as an expander graph, which gives it certain desirable properties. Expander graphs have strong connectivity despite being spare graphs, meaning that the graph has relatively few edges, but any two subsets of vertices have many edges connecting them. Constructing a hash function from this type of graph means that the outputs of the function are close to uniformly distributed, meaning that the output is indistinguishable from random bit sequences [8]. This property, along with the collision resistance, is what allows for Ramanujan Graphs, and subsequently SECs, to be suitable for hashing functions.

For this schema, the input to the hash function is used as a direction for walking around a graph without backtracking, with the output being the destination vertex. The vertices of the Ramanujan Graph are the set of isomorphism classes of SECs over $Fp^2$, labelled by their j-invariants. The edges of the graph are defined as follows:

- Given a Supersingular j-invariant j1, choose a curve where j(E1) = j1 and a subgroup of E1, H1.
- Form an edge between E1 and E2, where E2 = E1/H1, with the edge being from j1 to j2 = j(E2)

Given a subgroup C of an SEC E, the isogeny E → E/C can be easily computed as shown in [8]. Using this isogeny, the curve E can be traversed, leading to an output vertex. This can be utilized to create a hash function, as every distinct C will produce a different isogeny, leading to a different output vertex. The same isogeny can also be used to produce different results because the graph is an expander graph. A graph of with subgroups of order ℓ has vertices with ℓ + 1 edges, meaning that the same isogeny can lead to a different result depending on the path taken. This is what allows these graphs to be suitable candidates for hash functions, as each input will lead to a different output.

A collision will occur in this schema if two distinct isogenies create a path from the same starting vertex to the same ending vertex. This means that in order for an attacker to find a collision, they must be able to solve the problem of finding a pair of SECs E1 and E2 and calculate two distinct isogenies between the curves that reach the same location, meaning that the isogenies have the same degree. This is a similar problem to Problem 2, and relies on the difficulty of computing isogenies between curves.

To find a preimage for this hash function, an attacker must be able to solve Problem 1. Given the output y = h(x), let E2 be the SEC with j-invariant corresponding to y. To find x from this, an attacker must find a path in the graph of ℓ-isogenies from E1 to E2 [8]. As described above in Problem 1, this is a hard problem to solve even in quantum time.

Problems 1 and 2 ensure that this schema is secure so long as a more efficient method of computing isogenies between curves is not found. Since the best known solutions to these two problems have exponential running time when utilizing quantum computing, this schema can be used to produce strong hashes that are resilient to collisions and preimages. The exact running time of this known method of computeing isogenies is $O(\sqrt{p} \log^2 p)$, so by setting p = 256, the resulting hash function will receive 128 bits of security for when dealing with quantum computers.

## V. Isogeny Based Diffie-Hellman

To make use of SECs and their isogenies in a DH schema, an even more select version of the curves must be chosen: SECs with smooth order. This means that the order of the curve can only be divisible by small primes. The SECs that will be taken advantage of for this schema can be defined as follows:

- Let $\ell_A$ and $\ell_B$ be two small primes, $f$ be an integer cofactor.
- Let p be a prime of the form $(\ell_A^{eA} \ell_B^{eB} f)^2$
- Construct a SEC over $F_p^2$

It is also necessary for the schema to define the points on the curve $P_A$ and $Q_A$ which generate the torsion group $E[\ell_A^{eA}]$ and the points $P_B$ and $Q_B$ which generate the torsion group $E[\ell_B^{eB}]$.

To generate the public key, Alice will choose two integers $m_A$ and $n_A$ such that $R_A = [m_A]P_A + [n_A]Q_A$ with order $\ell_A^{eA}$. Her secret key is then computed to be the isogeny $\varphi_A: E_0 \rightarrow E_A$ whose kernel is $R_A$. Alice's public key then becomes $E_A$ along with the points $\varphi_A(P_B)$ and $\varphi_A(Q_B)$. Bob then does the same calculations using two integers $m_B$ and $n_B$ in order to generate $R_B$ and subsequently $\varphi_B: E_0 \rightarrow E_B$ whose kernel is $R_B$.

Bob's private key is $\varphi_B$, with his public key being $E_B$ and the points $\varphi_B(P_A)$ and $\varphi_B(Q_A)$. Alice and Bob then exchange public keys, as they would in any other Diffie-Hellman protocol. To compute the shared secret key, Alice uses her secret integers along with Bob's public key to compute the degree $\ell_A^{e_A}$ isogeny $\varphi_A' = E_B \rightarrow E_{BA}$ whose kernel is $[m_A]\varphi_B(P_A) + [n_A]\varphi_B(Q_A) = \varphi_B(R_A)$. Bob will then do the same calculations using his secret integers $m_B$ and $n_B$ to generate $\varphi_B' = E_A \rightarrow E_{AB}$ with kernel $\varphi_A(R_B)$. The resulting SECs of $E_{BA}$ and $E_{AB}$ are isomorphic, so the shared secret key becomes the j-invariant of these curves, seeing as isomorphic curves share the same j-invariant [9].

As with the previous implementations of SECs and their isogenies, the security of this Diffie-Hellman schema relies on the hard problem of computing specific isogenies. The shared secret between Alice and Bob can only be kept as long as both Alice's and Bob's isogenies $\varphi_A$ and $\varphi_B$ are unknown to attackers. Should an attacker get their hands on either of Alice's or Bob's secret integers, they still will have no way of computing the shared secret without access to one of their isogenies.

## VI. Conclusion

This paper compiled the knowledge of multiple cryptosystems that utilize Supersingular Elliptic Curves and their isogenies in order create schemas that are resilient to quantum attacks. The use cases of these types of schemas are many, as humanity comes closer and closer to achieving a breakthrough in quantum computing. While only three cryptosystems were examined here, the ideas and foundations of these systems utilizing SECs can be applied further to other schemas. With the hard problem of calculating isogenies of SECs backing these systems, they show great promise in becoming systems that are implemented once quantum computing has been achieved.

## References


[1] Chen, Lily, et al. "Report on Post-Quantum Cryptography." *CSRC*, 28 Apr. 2016, csrc.nist.gov/publications/detail/nistir/8105/final.

[2] "Shor's Algorithm." *IBM Quantum*, quantum-computing.ibm.com/composer/docs/iqx/guide/shors-algorithm. Accessed 2 May 2023.

[3] Galbraith, Steven D., et al. "Identification Protocols and Signature Schemes Based on Supersingular Isogeny Problems." *Journal of Cryptology*, vol. 33, no. 1, 2019, pp. 130–175, https://doi.org/10.1007/s00145-019-09316-0.

[4] *14 Ordinary and Supersingular Elliptic Curves - MIT Mathematics*. https://math.mit.edu/classes/18.783/2019/LectureNotes14.pdf.

[5] Hankerson, Darrel R, et al. *Guide to Elliptic Curve Cryptography*. New York ; London, Springer, 2011.

[6] Jean-François Biasse, et al. "A Quantum Algorithm for Computing Isogenies between Supersingular Elliptic Curves." *Lecture Notes in Computer Science*, 14 Dec. 2014, pp. 428–442, https://doi.org/10.1007/978-3-319-13039-2_25. Accessed 2 May 2023.

[7] Galbraith, Steven D., et al. "Identification Protocols and Signature Schemes Based on Supersingular Isogeny Problems." *Journal of Cryptology*, vol. 33, no. 1, 27 Mar. 2019, pp. 130–175, https://doi.org/10.1007/s00145-019-09316-0. Accessed 13 June 2021.

[8] Charles, Denis X., et al. "Cryptographic Hash Functions from Expander Graphs." *Journal of Cryptology*, vol. 22, no. 1, 15 Sept. 2007, pp. 93–113, https://doi.org/10.1007/s00145-007-9002-x. Accessed 4 Mar. 2020.

[9] Costello, Craig, et al. Efficient Algorithms for Supersingular Isogeny Diffie-Hellman.