How to Keep Your Secrets in a Post-Quantum World

Kristin Lauter

Cryptography

Cryptography is the science of keeping secrets. But it is more than that. It is now a flourishing branch of mathematics that, in addition to encryption, also provides other tools to protect security and privacy of individuals, enterprises, data, systems, and transactions. Cryptographic protocols enable us, for example, to create secure communication channels, to guarantee the confidentiality and integrity of messages and data, to authenticate the identity of the sender of a message or the endpoint in a transaction. Cryptography is the foundation of secure e-commerce in the world today, providing the trustworthy systems that allow enterprises and consumers to transact business online. Digital signatures and key exchange are two important building blocks for public-key cryptography, in addition to encryption.

Cryptographic systems are often built on the premise that certain math problems are very hard to solve, in the sense that known solutions require enormous computational time and resources. Many of these problems, such as factoring certain types of large numbers, have been studied by mathematicians for many decades. In fact, mathematicians often estimate the projected security of cryptographic systems by plotting the evolution in "running time" and "space requirements" of the best-known attacks. These predictions work well, but only in the absence of major disruptions: new algorithms or technologies that drastically improve the expected running time of attacks.

Kristin Lauter is a partner research manager and principal researcher of the Cryptography Group at Microsoft Research. Her email address is klauter@microsoft.com.

Communicated by Notices *Associate Editor Emilie Purvine.*

DOI: https://dx.doi.org/10.1090/noti2004

What Do We Mean by "Hard Math Problem"?

In practical applications of cryptography, we have a relatively well-agreed-upon meaning for the term "hard math problem": if the input is represented by m bits, then the best-known attack on the system runs in

$$\textit{exponential time} \text{ in } m, \text{ e.g., } O(2^m) \text{ time}$$

or

$$\textit{subexponential time} \text{ in } m,$$
$$\text{e.g., } L(\tfrac{1}{3}, c)=O(e^{c\cdot m^{\frac{1}{3}}(\log m)^{\frac{2}{3}}}) \text{ time, where } c \text{ is a constant.}$$

For example, to factor the number $n=p\cdot q$ where $m=\log n$, trial division takes *exponential time*. This is with respect to classical algorithms, which are represented on today's computers with circuits and with inputs and outputs given in terms of "classical" bits, i.e., sequences of 0s and 1s. *Polynomial time* algorithms run in time that is a *polynomial* in m, which often means in practice that an attack based on a polynomial time algorithm will succeed in a realistic amount of time and render the cryptographic system insecure.

Cryptographic Standards

There is a complex process for deploying new cryptographic protocols, especially when based on new hardness assumptions in mathematics. First, the research community needs to reach consensus on the above described process of modeling and giving precise, concrete cost estimates for the best-known attacks that solve the underlying math problem. Second, detailed standards are created through community or government processes such as:

1. a government agency like NIST (National Institute of Standards and Technology) in the United States runs a multiyear, open, international competition, e.g., the block cipher competition that standardized AES or the hash function competition that standardized SHA-3;
2. a professional society such as IEEE (Institute of Electrical and Electronics Engineers) or IETF (Internet Engineering Task Force) convenes a working group or a committee to develop a draft standard, which is updated and revised over time, e.g., the IEEE P1363 that provided a foundational standard for elliptic curve cryptography (ECC);
3. a consortium consisting of researchers from a collection of interested parties in industry, government, and academia works together to publish a draft standard for reference, e.g., the PKCS standards governing the deployment of the RSA system or the new draft standard for Homomorphic Encryption HES 1.0 [HES 2018].

There can be substantial overlap in these first two stages of standardization. Once draft standards have been developed, there is a regulatory layer that is often developed requiring the deployment or adherence to various standards. Specialized standards are often developed for protocols to be used in vertical segments of the economy, such as when ANSI (American National Standards Institute) produced the X9.62 and X9.63 ECC standards for using elliptic curve key exchange and digital signature protocols in the financial services industry. Other examples of protocol-level standards include specifications for secure browser sessions (https: SSL/TLS); signed, encrypted email (S/MIME); virtual private networking (IPSec); and authentication (X.509 certificates).

Finally, there may be an ecosystem of third-party vendors that spring up to respond to the need to verify compliance with regulations. This is the current process for establishing public trust in the cryptographic systems we deploy. It is important that much of this process be public so that everyone can see that the systems were not cooked up in a back alley with some secret trapdoors or weaknesses built in.

The possibility of new, sometimes unexpected, attacks on fundamental cryptographic problems in mathematics, combined with the lengthy and complex standardization process, leaves us in a difficult and sometimes precarious position. Recent advances and substantial new investment in the development of quantum computers represent such a potential threat to our currently widely deployed public key cryptographic systems. This is due to the existence of a polynomial time quantum attack [Shor97] on practically all of our currently deployed public key cryptosystems, which will be feasible to implement once a quantum computer can be built at a large enough scale. In response, NIST has launched a new, multiyear process to standardize post-quantum cryptography (PQC)[1]: i.e., cryptographic systems based on hard math problems for which we do not currently know *polynomial time quantum attacks*. The NIST PQC competition was launched in November 2017, and the twenty-six submissions for key exchange and digital signatures that have advanced to the second round were announced in January 2019.[2] Round 2 is expected to be a 12–18-month process. There may be a third round before NIST announces the post-quantum algorithms that will be recommended.

Pre-quantum (Classical) Systems

The NIST PQC selection process aims to identify candidates to supplement or replace three standards considered to be most vulnerable to a quantum attack: FIPS 186–4,[3] which specifies how to use digital signatures, and NIST SP 800–56A[4] and NIST SP 800–56B[5], which are specifications

[1] https://csrc.nist.gov/Projects/Post-Quantum-Cryptography

[2] https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions

[3] https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf

[4] https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Ar2.pdf

[5] https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Br1.pdf

for key exchange. These currently widely deployed systems are based on "classically" hard problems, for which we do not know any classical polynomial time algorithms: RSA, Diffie-Hellman, and ECC. The RSA cryptosystem for encryption was proposed in the 1970s and is based on the hardness of factoring large integers that are the product of two prime numbers of equal size. Diffie–Hellman key exchange is based on the hardness of solving the discrete logarithm problem in the multiplicative group of integers modulo a large prime number. Elliptic curve cryptosystems are based on the hardness of solving ECDLP, the discrete logarithm problem in the abelian group of points on an elliptic curve over a finite field. Although there is a rich and beautiful mathematical theory of elliptic curves, developed over the course of more than one hundred years by mathematicians, cryptographers often think of an elliptic curve as simply the set of solutions to an affine equation in a finite field F_q. In characteristic not equal to 2 or 3, this equation is given in short Weierstrass form:

$$E\colon y^2 = x^3 + ax + b,$$

where a and b are constants in the base field F_q. The set of affine solutions, along with a "point at infinity" that can be seen in the projective version of the equation, forms a group where the point at infinity is the identity element. The group law can be described with concrete rational functions and has been widely implemented in industry to enable cryptographic systems, starting with Windows Vista and OpenSSL in 2005.

For RSA and Diffie–Hellman systems, classical subexponential attacks are known: the number field sieve and the index calculus attack; see the *Notices* article by Pomerance [Po96] for the history. Current key sizes for ECC systems are much smaller than for RSA or Diffie–Hellman because there are no known subexponential classical attacks on ECDLP for generic, ordinary elliptic curves. In 2006, the NSA published the Suite B algorithms, which provided guidance recommending adoption of ECC and mandated it for systems used by government contractors. In 2016, new guidance was released, recommending larger key sizes for ECC: 384 bits minimum instead of a 256-bit minimum. The revised guidance on the bit size raises the bar on the size of a quantum computer required to mount a successful quantum attack on ECC.

An Emerging Threat: The Quantum Computer

Many researchers, industrial labs, and governments are actively working on developing a quantum computer that can handle large-scale computation, such as the work at Station Q,[6] Microsoft's quantum computing headquarters. While classical computers—phones, tablets, laptops, servers, and so on—store and process information in the form of bits (strings of zeros and ones), quantum computers will process quantum bits, which are two-state quantum mechanical systems called qubits. In contrast to a classical bit, a qubit can simultaneously hold all values between zero and one, with each value having a specified probability. Then, when measured, the state of the qubit collapses to either zero or one. Small-scale quantum computers already exist, and estimates vary as to how many years it will take before researchers and engineers succeed in building a quantum computer that can handle computations involving thousands of qubits. However, when that day arrives, the consequences for the world's e-commerce and security infrastructure will be enormous.

Basic arithmetic on a quantum computer is different than on a classical computer. Computation on qubits is specified via quantum circuits consisting of quantum gates. Quantum logic gates are represented by unitary matrices. It remains to be seen which quantum gates and architectures will be achieved and scaled up in practice.

In 1994, Shor [Shor97] introduced a quantum algorithm that can factor large integers in polynomial time, given a quantum computer that can accurately process those computations on a large enough number of qubits. A variant of this idea also allows polynomial-time quantum attacks on all of the other currently widely deployed public key cryptosystems used in industry and government today. Shor's algorithm for factoring on a quantum computer runs in $4m^3$ time and requires $2m$ qubits, where m is the number of bits required to represent the number to be factored ([PZ03]). The current standard minimum for RSA moduli is $m = 2048$ bits. The Proos–Zalka estimates for attacking the elliptic curve discrete logarithm problem were updated in [RNSL17] to $9n + 2\log_2 n + 10$ qubits using a quantum circuit of at most $448n^3 \log_2 n + 4090n^3$ Toffoli gates for an ordinary elliptic curve over F_q where $n = \log_2 q$. The conclusion is that 2048-bit RSA and elliptic curve cryptography for $n = 256$ or 384 will not be resistant to quantum attacks once a quantum computer exists at scale.

Post-Quantum Cryptography

The NIST Post-Quantum Cryptography (PQC) competition aims to select post-quantum cryptosystems that are not currently known to be breakable in polynomial time by a full-scale quantum computer. The following are the four main types of proposals for post-quantum systems based on hard math problems, in order of when the hard problem was first proposed in cryptography. Code-based cryptography has been studied for more than four decades, for example, whereas supersingular isogeny graphs have been studied for only about fifteen years. There are tradeoffs in size, performance, and security for each proposal.

1. Code-based systems are based on the difficulty of decoding random linear error-correcting codes. McEliece introduced these cryptosystems in [Mc78] using binary Goppa codes. Decoding Goppa codes efficiently is

[6] https://news.microsoft.com/stories/stationq/index.html

possible due to an algorithm of Patterson ([Pa75]). The security of the schemes also relies on disguising the Goppa code as a general linear code.

2. Multivariate cryptosystems are based on the difficulty of solving systems of many nonlinear equations in many variables over a finite field F_q. Imai and Matsumoto introduced the C^* scheme in [MI88], and variants were introduced by Patarin and others in follow-up work. Although many proposed multivariate cryptographic systems have been broken, there are still viable proposals that have been submitted to the NIST PQC competition, such as Rainbow for signature schemes.
3. Lattice-based systems are based on the hardness of finding short vectors in lattices. Lattice-based cryptography was introduced in the mathematics community in 1996, when Hoffstein, Pipher, and Silverman ([HPS98]) proposed the system called NTRU. NTRU can be interpreted as a lattice-based system that is especially efficient because of its description in a special kind of number ring.

   A lattice is a linear space generated by a choice of basis vectors. One can imagine it in Euclidean space, where a random set of linearly independent vectors is specified and the lattice consists of all points that are integer linear combinations of these vectors. Given an arbitrary basis with very long vectors in very large dimensions, it is a hard problem to find the shortest vector in the lattice. The best-known algorithms for solving the shortest vector problem run in exponential time in n, the dimension of the lattice. There are well-known polynomial time algorithms ([LLL82]) for finding approximate solutions, but the ratio of the length of the approximate vector to the length of the shortest vector is exponentially bad.
4. Supersingular isogeny graph (SIG) systems were introduced in [CGL06] based on the hard problem of finding paths between random vertices in large, random-looking graphs. In particular, Charles, Goren, and Lauter proposed and implemented cryptographic hash functions based on supersingular isogeny graphs and presented it at the 2005 NIST Hash Function Workshop.

For more information on the first three proposed approaches and hard problems, see the NIST PQC website or the *IEEE Security and Privacy* magazine issue on post-quantum cryptography, which has short articles on each proposed candidate [BLM17].

The rest of this article is devoted to explaining the mathematics of supersingular isogeny graphs and their applications in cryptography. Although this is the newest proposal among the four main approaches and thus requires further study to gain confidence in the security, the mathematics is interesting and compelling enough to merit exposition.

Supersingular Isogeny Graphs

Supersingular isogeny graphs (SIG) were introduced as a hard problem into cryptography by Charles, Goren, and Lauter at the NIST Hash Function competition in 2005. The hard problem is *routing* in these graphs; i.e., given two nodes or vertices in the graph, find a path between them (or find a path of a *certain length* between them).

We will define these graphs precisely later, but first, Figure 1 is a picture of a very small SIG, which we produced to appear in *Science* magazine in 2008 [Ma08]. To get a feel for the hard problem underlying this proposal, pick two random points in the graph and try to find a path between them. Then try to imagine this same problem in a graph that has 10^{75} times as many vertices as this one.

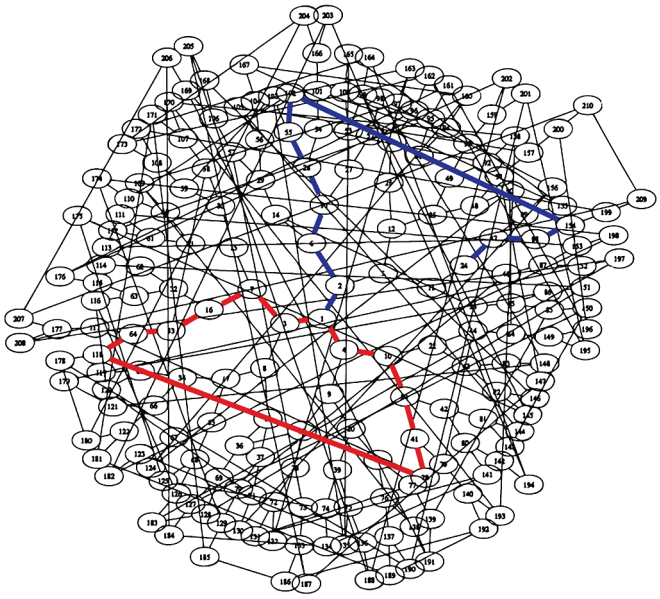

Figure 1. Supersingular isogeny graph for $p = 2521$.

Definition of Supersingular Isogeny Graphs

Supersingular elliptic curves. Let p and ℓ be two distinct prime numbers. For our cryptographic applications, p will be the characteristic of a finite field, which is a very large prime of cryptographic size, while ℓ will be the degree of a map and very small, typically $\ell = 2$ or 3. Elliptic curves were described above, and since the characteristic of the finite field is not equal to 2 or 3, we can work with the short Weierstrass equation for the elliptic curve: $E : y^2 = x^3 + ax + b$.

An elliptic curve over a finite field of characteristic p is *supersingular* if it has no p-torsion over its base field or any extension field. It is known that each isomorphism class of supersingular elliptic curves modulo p has a representative over the finite field of p^2 elements. Elliptic curves that are not supersingular are called ordinary. The *j-invariant* is an isomorphism invariant of an elliptic curve, and it can be easily computed as a rational function of the coefficients of the curve equation:

$$j(E) = 1728\frac{4a^3}{4a^3+27b^2}.$$

An *isogeny* between two elliptic curves is a morphism that preserves the group structure. The degree of a separable isogeny is the size of its kernel, so to construct an isogeny of degree ℓ from one elliptic curve E to another, take a subgroup C of size ℓ, and take the quotient E/C. In our setting, the prime ℓ is different from p, so the isogenies are all separable. Explicit formulae for isogenies of degree ℓ and the equation for E/C were given by Velu [Velu71].

The graphs. Define the supersingular isogeny graph $G(p,\ell)$ to have vertex set equal to the set of isomorphism classes of supersingular elliptic curves over the algebraic closure of the finite field with p elements. The number of vertices of $G(p,\ell)$ is the Eichler class number, which is roughly $\frac{p}{12}$ and depends on the congruence class of p modulo 12 ([Sil09]). Vertices are labeled with their j-invariants, which can be computed directly from the curve equation.

The edges of the graph $G(p,\ell)$ are the isogenies of degree ℓ between elliptic curves, up to composing with an automorphism of the target curve. Since ℓ is prime and not equal to p, the number of distinct edges coming out of each vertex is $\ell+1$, because there are $\ell+1$ distinct subgroups of order ℓ of the ℓ-torsion of E. To make the graph undirected, we can associate an isogeny in one direction with its dual isogeny in the opposite direction. If we impose the congruence condition $p \equiv 1 \pmod{12}$, then there is no ambiguity and we can consider the graphs to be undirected [Pizer90, CGL06].

Expansion and Ramanujan Properties of the Supersingular Isogeny Graphs

We now summarize the basic properties of $G(p,\ell)$. These are connected graphs (see [Mestre86] or a special case of [CGL09, Theorem 4.1]) with roughly $\frac{p}{12}$ vertices [Sil09, Theorem 4.1]. If $p \equiv 1 \pmod{12}$, then they are undirected and $\ell+1$–regular, with vertices labeled by j-invariants.

In the next section we will describe cryptographic applications of these graphs. In particular [CGL06] defined a hash function based on random walks on the graphs $G(p,\ell)$ for which the output should be as close to uniformly distributed as possible. But first we must define the concept of an expander graph and its expansion constant, which are closely correlated to this property. An *expander graph* with vertex set V and N vertices has expansion constant $c>0$ if for any subset U of V of size

$$|U| \le \frac{N}{2},$$

the boundary (neighbors of U not in U) satisfies

$$|\Gamma(U)| \ge c|U|.$$

The adjacency matrix of an undirected graph is symmetric, and therefore all its eigenvalues are real. For a connected k-regular graph, the largest eigenvalue is k, and all others are strictly smaller:

$$k > \mu_1 \ge \mu_2 \ge \cdots \ge \mu_{N-1}.$$

The expansion constant c can be expressed in terms of the eigenvalues as follows:

$$c \ge 2\frac{k-\mu_1}{3k-2\mu_1}.$$

Therefore, the smaller the eigenvalue μ_1, the better the expansion constant, and the distance between the first and second eigenvalues, $k-\mu_1$, is referred to as the *spectral gap*. A theorem of Alon–Boppana says that for an infinite family of connected, k-regular graphs, X_m, indexed by m, with the number of vertices in the graphs tending to infinity,

$$\liminf_{m\to\infty} \mu_1(X_m) \le 2\sqrt{k-1}.$$

We define a *Ramanujan graph* to be a k-regular connected graph with optimal expansion properties in the sense that it satisfies

$$\mu_1 \le 2\sqrt{k-1}.$$

A random walk on an expander graph mixes very fast, so the output of the hash function will be roughly uniform, provided the walk is long enough. The output of a random walk on an expander graph with N vertices tends to the uniform distribution after roughly $O(log(N))$ steps, where the exact distance from the uniform distribution depends in a precise way on the expansion constant.

Supersingular isogeny graphs $G(p,\ell)$ are optimal expander graphs when $p \equiv 1 \pmod{12}$ in the sense that they are Ramanujan graphs (see [Pizer90, Prop. 4.7] or a special case of [CGL09, Theorem 4.2]). The Ramanujan property of this graph follows from the fact that the adjacency matrix (called the Brandt matrix) gives the action of a Hecke operator on the space of weight 2 cusp forms of level p. So the bound on the eigenvalues follows from the corresponding result for modular forms (the Ramanujan–Petersson conjecture).

Applications

Cryptographic hash functions. A *hash function* maps bit strings to bit strings:

$$h: \{0,1\}^n \to \{0,1\}^m.$$

A hash function h is said to be *collision resistant* if it is computationally infeasible to find two distinct inputs, x,y, that hash to the same output, $h(x)=h(y)$. It is *preimage resistant* if, given any output of h, it is computationally

infeasible to find an input, x, that hashes to that output. To be useful in cryptographic applications and protocols, hash functions should have at least the following properties: they should be easy to compute, unkeyed (do not require a secret key to compute output), collision resistant, and preimage resistant, with an approximately uniformly distributed output.

The cryptographic hash function proposed in [CGL06] based on hardness of routing in supersingular isogeny graphs was defined as follows. A fixed vertex in the graph is specified as the starting point. The input bit string is divided into blocks and used as directions for walking around the graph. At each step in the walk, the choice of the next edge to follow is determined by the next block of bits of the input. No backtracking is allowed, since that would allow for trivial collisions of walks that go forward and backward along an edge at two different steps in the walk! The output of the hash function is the label for the final vertex of the walk. A family of hash functions can be defined by allowing the starting vertex to vary. For a k-regular expander graph with $k-1=2^e$ being a power of 2, the bits are read off in chunks of length e. For example, if $k=3$, then $e=1$ and bits are processed one at a time.

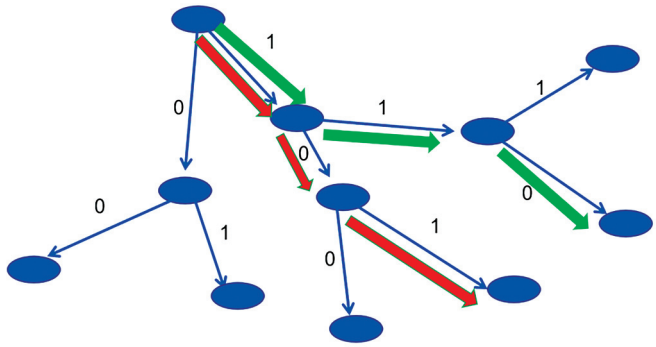

Figure 2. Walk on a 3-regular graph: 110 is the green path and 101 is the red path.

In order to avoid collisions in cryptographic hash functions based on isogeny graphs, it is best if the graph has no short cycles. Charles, Goren, and Lauter show in [CGL06] how to ensure that isogeny graphs do not have short cycles by carefully choosing p to satisfy various congruence conditions. For example, they compute that a 2-isogeny graph does not have double edges (i.e., cycles of length 2) when working over F_p with $p \equiv 1 \bmod 420$.

The security of the hash function relies on the hardness of finding paths, or *routing*, in this graph. If you can find a path between two given vertices of this graph, then you have found a *preimage* for the hash function specified by that starting point. Collisions and preimages can be found in the graph using the generic *birthday attack*, which involves randomly walking around the graph from two different starting points until a collision is detected. The birthday attack runs in time proportional to the square root of the size of the graph, $O(\sqrt{p})$. No better classical attacks are currently known. To achieve 128-bits of security against the birthday attack, in practice we pick p so that $\log p \approx 256$. The best-known quantum algorithm for computing isogenies between supersingular elliptic curves runs in time $O(p^{1/4})$, ignoring log factors [BJS14].

Key exchange. One of the fundamental public key protocols being standardized in the NIST post-quantum cryptography competition is key exchange. Key exchange refers to a protocol for two parties to: 1. specify their public parameters; 2. each pick a secret; 3. publicly exchange information with each other; and 4. compute a common key that only the two parties know. The following key exchange protocol was proposed in [DFJP14].

Let E be a supersingular elliptic curve defined over the finite field with p^2 elements, where

$$p = \ell_A{}^m \ell_B{}^n \pm 1$$

and ℓ_A and ℓ_B are distinct small primes and m and n are balanced. In practice $\ell_A = 2$ and $\ell_B = 3$. In that case, m and n are roughly equal to $\frac{1}{2}\log_2 p$ and $\frac{1}{2}\log_3 p$, respectively.

Suppose two parties, A (for Alice) and B (for Bob), wish to engage in a key-exchange protocol with the goal of establishing a shared secret key by communicating via a (possibly) insecure channel. Alice and Bob generate their public parameters: Alice picks two points P_A and Q_A that generate the $\ell_A{}^m$-torsion, and Bob picks two points P_B and Q_B that generate the $\ell_B{}^n$-torsion.

Alice then secretly picks two random positive integers m_A and n_A, which will be her secret parameters. She then computes the isogeny Φ_A from E to another curve E_A, which corresponds to taking the quotient of E by the subgroup generated by $m_A P_A + n_A Q_A$. Bob does the same and secretly picks two random positive integers m_B and n_B. He then computes the secret isogeny Φ_B by taking the quotient of E by the subgroup generated by $m_B P_B + n_B Q_B$.

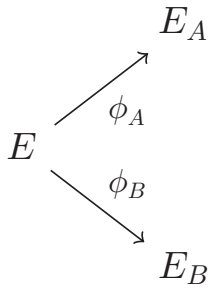

Figure 3. First stage of supersingular isogeny key exchange.

So far, Alice and Bob have constructed the diagram shown in Figure 3.

In the next stage of the exchange protocol, Alice computes $\Phi_A(P_B)$ and $\Phi_A(Q_B)$ and sends $\{\Phi_A(P_B), \Phi_A(Q_B), E_A\}$ to Bob. Similarly, Bob computes and sends $\{\Phi_B(P_A), \Phi_B(Q_A), E_B\}$ to Alice. Both players now have enough information to construct the diagram shown in Figure 4, where $E_{AB} = E/\langle m_A P_A + n_A Q_A,\ m_B P_B + n_B Q_B \rangle$. Alice can use the secret information m_A and n_A to compute the isogeny Φ'_B by taking the quotient of E_B by the subgroup generated by $m_A \Phi_B(P_A) + n_A \Phi_B(Q_A)$ to obtain E_{AB}. Bob can use the secret information m_B and n_B to compute the isogeny Φ'_A, taking the quotient of E_A by the subgroup generated by

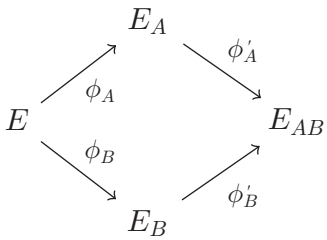

Figure 4. Completion of supersingular isogeny key exchange.

$m_B n_A(P_B) + n_B \Phi_A(Q_B)$ to obtain E_{AB}. A separable isogeny is determined by its kernel, and so both ways of going around the diagram from E result in computing the same elliptic curve E_{AB}. Alice and Bob can both compute the curve E_{AB} and use its j-invariant as a shared secret.

A Security Reduction

The security of the supersingular isogeny key-exchange protocol (SIKE) is based on a hardness assumption stated in [DFJP14], called the supersingular computational Diffie–Hellman (SSCDH) problem. However, the connection with the path-finding problem introduced in [CGL06] was not published until the paper by Costache, Feigon, Lauter, Massierer, and Puskas [CFLMP19], which showed that the SSCDH problem is no harder than the CGL-path-finding problem, and it is entirely possible that it is easier to solve, given that there is more auxiliary information available in the SSCDH problem.

Theorem [CFLMP19]. *Assume as for the key exchange setup that* $p = \ell_A^m \ell_B^n \pm 1$ *is a prime of cryptographic size, i.e.,* $\log p \geq 256$, ℓ_A *and* ℓ_B *are distinct small primes, and* m *and* n *are balanced so that* ℓ_A^m *is approximately* ℓ_B^n. *In practice* $\ell_A = 2$ *and* $\ell_B = 3$. *Given an algorithm to solve the CGL path-finding problem in supersingular isogeny graphs, it can be used to break the supersingular key exchange with overwhelming probability. The failure probability is roughly* $\frac{1}{\sqrt{p}}$.

Conclusion

While mathematicians have been researching the hard problem of factoring large integers for centuries, we are now faced with the prospect that our future security may depend on the hardness of mathematical problems that have been studied by mathematicians for only a matter of decades. This disconcerting fact is made worse by the fact that there is an urgent need to understand both the classical and the quantum security of these new proposals. So the current answer to the question in my title is "We don't know yet!" It is clear, though, that there are very interesting mathematical problems that could serve as the basis of the next generation of post-quantum secure cryptosystems—we just need more mathematicians working on them to understand the security!

To apply for NSF funding for research projects in cryptography and cybersecurity, visit the program solicitation for Secure and Trustworthy Cyberspace (SaTC).[7]

[7] https://www.nsf.gov/funding/pgm_summ.jsp?pims_id=504709

References

[BJS14] Biasse J-F, Jao D, Sankar A, A quantum algorithm for computing isogenies between supersingular elliptic curves, *Progress in Cryptology—INDOCRYPT 2014*, 428–442, Lecture Notes in Comput. Sci., 8885, Springer, 2014. MR3296936

[BLM17] Buchmann J, Lauter K, Mosca M, editors, Postquantum cryptography—state of the art, *IEEE Security & Privacy*, vol. 15, July/August 2017. DOI:10.1109/MSP.2017.3151326

[CGL06] Charles DX, Goren EZ, Lauter KE, Cryptographic hash functions from expander graphs, *J. Cryptology* **22** (2009), no. 1, 93–113. MR2496385. Available at https://eprint.iacr.org/2006/021.pdf, NIST presentation.

[CGL09] Charles DX, Goren EZ, Lauter KE, Families of Ramanujan graphs and quaternion algebras, *Groups and Symmetries*, CRM Proc. Lecture Notes, vol. 47, Amer. Math. Soc., Providence, RI, 2009, pp. 53–80. MR2500554

[CFLMP19] Costache A, Feigon B, Lauter K, Massierer M, Puskas A, Ramanujan graphs in cryptography, *Research Directions in Number Theory: Women in Numbers IV*, Association for Women in Mathematics Series, Vol. 19, 2019.

[DFJP14] De Feo L, Jao D, Plut J, Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies, *J. Math. Cryptol.* **8** (2014), no. 3, 209–247. MR3259113

[HES18] *Homomorphic Encryption Standard*, Albrecht M, Chase M, Chen H, Ding J, Goldwasser S, Gorbunov S, Halevi S, Hoffstein J, Laine K, Lauter K, Lokam S, Micciancio D, Moody D, Morrison T, Sahai A, Vaikuntanathan V, November 2018. Available at: https://eprint.iacr.org/2019/939.pdf.

[HPS98] Hoffstein J, Pipher J, Silverman JH, NTRU: A ring-based public key cryptosystem. *Algorithmic Number Theory*. ANTS 1998. Lecture Notes in Computer Science, vol. 1423. Springer, Berlin–Heidelberg. MR1726077

[LLL82] Lenstra AK, Lenstra Jr. HW, Lovasz L. Factoring polynomials with rational coefficients. *Math. Ann.* 261(4), 515–534, 1982. MR0682664

[Ma08] Mackenzie D, Cryptologists cook up some hash for new 'bake-off,' *Science Magazine*, Vol. 319, March 14, 2008, 1480–1481. www.sciencemag.org.

[MI88] Matsumoto T, Imai H, Public quadratic polynomial-tuples for efficient signature-verification and message-encryption, *Advances in Cryptology—EUROCRYPT '88 (Davos, 1988)*, Lecture Notes in Comput. Sci., vol. 330, Springer, Berlin, 1988, pp. 419–453. MR0994679

[Mc78] McEliece RJ, *A Public-Key Cryptosystem Based on Algebraic Coding Theory*, DSN Progress Report, 44, pp. 114–116 (1978). Bibcode:1978DSNPR..44..114M

[Mestre86] Mestre J-F, *La méthode des graphes. Exemples et applications*, Taniguchi Sympos., Kyoto, 1986.

[Pa75] Patterson NJ, The algebraic decoding of Goppa codes. *IEEE Trans. Information Theory*. IT–21 (2): 203–207 (1975). MR0379009

[Pizer90] Pizer A, Ramanujan graphs and Hecke operators, *Bull. Amer. Math. Soc.*, 23, no. 1, July 1990. MR1027904

[Po96] Pomerance C, A tale of two sieves, *Notices Amer. Math. Soc.*, 43, no. 12, 1473–1485, December 1996. MR1416721

[PZ03] Proos J, Zalka C, Shor's discrete logarithm quantum algorithm for elliptic curves, *Quantum Information & Computation*, 2003. MR2003569

[RNSL17] Roetteler M, Naehrig M, Svore KM, Lauter KE, *Quantum Resource Estimates for Computing Elliptic Curve Discrete Logarithms*, International Conference on the Theory and Application of Cryptology and Information Security ASIACRYPT 2017: Advances in Cryptology – ASIACRYPT 2017, LNCS vol. 10625, Springer, 2017, pp. 241–270. MR3747727

[Shor97] Shor P, Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, *SIAM J. Comput.*, 26, no. 5, 1997, 1484–1509. MR1471990
[Sil09] Silverman JH, *The Arithmetic of Elliptic Curves*, second ed., Graduate Texts in Mathematics, vol. 106, Springer, Berlin–Heidelberg–New York, 2009. MR2514094
[Velu71] Velu J, Isogenies entre courbes elliptiques, *C. R. Acad. Sci. Paris Ser. A–B* 273 (1971), A238–A241. MR0294345

Kristin Lauter

Credits

Article opening image © Getty Images.
Figures 1–4 and author photo are courtesy of the author.