Quantum Resistant Public Key Exchange – The Supersingular Isogenous Diffie-Hellman Protocol

Hartwig Mayer
{hartwig.mayer}@coinfabrik.com

CoinFabrik

October 14, 2016

1 Introduction

Quantum computers clearly form part of the most exciting industrial and academic research. Their impact on cryptography would be immense, and according to the Report on Post Quantum Cryptography by the National Institute of Security and Technology it is time to search for quantum resistant cryptographic primitives.

On the other hand, many challenges of the quantum mechanical world must be overcome before large quantum computers become a reallity. One of these challenges is decoherence: the problem of having uncontrolled interaction of the quantum register with the environment. Prototypes for Qubits in the register of a quantum computer may be built by hydrogen atoms or electrons, instead of being built by capacitors of classical computers. Therefore, when the very fragile states of Qubits are manipulated by a quantum gate for computations, quantum entanglements which disturb the quantum computations themselves can occur.

Leaving the practical issues aside, the first rigorous definition of a quantum Turing machine was set forth in [Deu85] by D. Deutsch in the 80s. Based on this definition, Grover, Shor, Simon and others demonstrated that the search for quantum computers was worthwhile. For example, Shor's algorithm [Sho97], [BL95] solves the discrete logarithm problem over finite fields and over elliptic curves on a quantum computer in polynomial time. A classical computer cannot efficiently solve this problem. As a consequence, large quantum computers would make many cryptosystems obsolete and necessitate Post Quantum Cryptography (PQC) (see here).

In this article we will look at the Supersingular Isogenous Diffie-Hellman (SIDH) protocol which is considered a promising candidate for PQC. In contrast to regular Elliptic Curve Cryptography where the points of one elliptic curve are used for secret computations, the SIDH looks at the world of elliptic curves from a far. Loosely speaking, in SIDH the points of individual elliptic curves recede and the curves themselves are used in the construction of shared secret keys. We explain this in section 3 (see Table 1 for a

quick overview). In section four we briefly discuss security aspects and recent work by Costello, Longa, and Naehrig [CLN16] presenting a high-speed implementation of SIDH.

2 The Power of Quantum Algorithms – Two Examples

To lower the expectations about quantum computers a little bit, note that classical computers can simulate them. This implies that all problems a quantum computer can solve, can also be solved on a classical computer. But some problems that a quantum computer can solve in polynomial time would require exponential time on a classical computer. And this is the interesting point.

Let us review the basic difference between quantum and classical computers. In a classical computer one Bit can assume two states which are usually denoted by 0 and 1. The register of a computer with N Bits can hence realize 2^N possible states. In a quantum computer the basic units are called Qubits and their states are written in the form $|0\rangle$ and $|1\rangle$. The difference with Bits is that Qubits also allow superpositions of these states. This is expressed in mathematical terms as $\alpha|0\rangle + \beta|1\rangle$ with $\alpha, \beta \in \mathbb{C}$ normed such that $|\alpha|^2 + |\beta|^2 = 1$. Moreover, a quantum register with N Qubits can represent linear combinations of the individual Qubits, and is hence a 2^N dimensional space. Therefore, a quantum Turing machine can deal in some sense with multiple inputs at the same time, e.g $\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$. This effect is called *quantum parallelism* and allows us to develop more efficient algorithms for some types of problems.

Deutsch's Algorithm:

Problem: Given a function $f : \{0,1\} \longrightarrow \{0,1\}$ as a black box. Decide whether the function f is constant or not.

To solve the above problem, all algorithms for classical computers have to call the function f for 0 and 1, and compare the values. Deutsch's algorithm for quantum computers only need to call f once in order to check whether f is constant or not: Assume a quantum computer with a 2-Qubit register $|x\rangle|y\rangle$ and a quantum gate which performs the following operation

$$U(|x\rangle|y\rangle) := |x\rangle|y + f(x)\rangle$$

where the addition is modulo 2, i.e. in $\mathbb{Z}/2\mathbb{Z}$. The quantum register should be in the superposition:

$$|x\rangle|y\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}} \frac{|0\rangle - |1\rangle}{\sqrt{2}}.$$

If we calculate the effect of U, we derive

$$U(|x\rangle|y\rangle) = \frac{1}{2} U\left(|0\rangle|0\rangle - |0\rangle|1\rangle + |1\rangle|0\rangle - |1\rangle|1\rangle\right) = \frac{|0\rangle \pm |1\rangle}{\sqrt{2}} \frac{|f(0)\rangle - |\overline{f(0)}\rangle}{\sqrt{2}}$$

where $\overline{f(0)}$ is 0 or 1 depending on whether $f(0)$ was 1 or 0, respectively. Deutsch's algorithm is based on the fact that the first Qubit is $\frac{|0\rangle+|1\rangle}{\sqrt{2}}$ when f is constant and $\frac{|0\rangle-|1\rangle}{\sqrt{2}}$ if it is not, so only one call of f is required.

Note that it is still impossible to know all values of f with one call. What we gain in general is only some knowledge about *global properties* – in this example, whether f is constant or not.

SHOR'S ALGORITHM: Let $E/\mathbb{F}_q$ be an elliptic curve over a finite field $\mathbb{F}_q$ (see Figure 1 below), and assume that the cardinality $|E(\mathbb{F}_q)| = r$ is a prime. Fixing a basepoint $P_0 \in E(\mathbb{F}_q)$, the Discrete Logarithm Problem (DLP) reads as follows:

Problem: Given a point $Q \in E(\mathbb{F}_q)$ find the smallest positive integer a such that $Q = a \cdot P_0$.

D. Boneh and R. Lipton generalized Shor's algorithm so that DLP can also be solved over elliptic curves in random quantum polynomial time, i.e. efficiently on quantum computers. In order to do so, they restate the problem as: Let f be a function

$$f : \mathbb{Z}^2 \longrightarrow E(\mathbb{F}_q)$$

given by $f(x, y) := h(x + \alpha y)$ ($\alpha \in \mathbb{Z}$) where $h : \mathbb{Z} \longrightarrow E(\mathbb{F}_q)$ is the homomorphism $\alpha \mapsto h(\alpha) := \alpha \cdot P_0$. This type of function is referred to as a *hidden linear* function. Theorem 2 in [BL95] states that for such functions the integer α can be computed in quantum polynomial time with respect to r. To solve the initial problem, they consider the function $f(x, y) = h(x) + y \cdot Q = h(x) + y \cdot h(a) = h(x + ay)$, which is a hidden linear function, so their algorithm can determine a in polynomial time.
The proof of their theorem has the following basic structure: One does a quantum experiment $\mathcal{Q}$ and defines a subset $\mathcal{V}$ of all possible observable values of $\mathcal{Q}$ such that:

1. If we observe $v \in \mathcal{V}$, the integer α can be determined in polynomial time (on both classical and quantum computers). See [BL95], 6.2.

2. The probability of observing an output $v \in \mathcal{V}$ is great enough that the necessary repetitions can on average be performed in polynomial time. See [BL95] 6.3 and 6.4.

The essential part of the quantum experiment $\mathcal{Q}$, is the application of the *Quantum Fourier transform* F_W to the function f at random values for x and y. Shor's original insight allows the Quantum Fourier Transorm to be performed in quantum polynomial time (see [Sho97], chapter 4 for details). Finally, one has to define the set $\mathcal{V}$ and prove that it has the properties stated in 1 and 2. This is beyond the scope of this article and is clearly explained in section 6 of the article [BL95].

3 Supersingular Isogenous Diffie-Hellman Protocol

The Diffie-Hellman protocol works generally for any finite (commutative) group, e.g. $\mathbb{Z}/p\mathbb{Z}$. But as we have just seen, Shor's algorithm can be used to efficiently attack this

protocol. Therefore, the idea of SIDH is to mimic the protocol but to avoid working with a group.

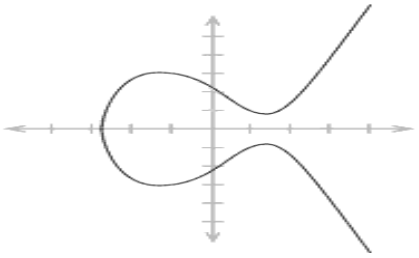
When applied to elliptic curves, the Diffie-Hellman protocol works on an elliptic curve E with a fixed basepoint $P_0 \in E$. The two participants A and B then share a secret key S_P which is a new point on the elliptic curve E obtained by group operations applied to the point P_0 (see left column of Table 1). The Supersingular Isogenous Diffie-Hellman protocol works on the set of elliptic curves and fixes a 'basecurve' E_0. The two participants A and B then share a secret key S_E which is a new elliptic curve obtained by applying a special map to the curve E_0 (see right column of Table 1). This map is an isogeny – the I in SIDH – and 'replaces' the group operation. Figure 1 defines isogenies.

Objects: **Elliptic curves**

An elliptic curve $E/\mathbb{F}_q$ is the set of all points $(x, y) \in \overline{\mathbb{F}}_q \times \overline{\mathbb{F}}_q$ satisfying

$$E : y^2 = x^3 + Ax + B \ \ (A, B \in \mathbb{F}_q).$$

The following graphic illustrates a possible elliptic curve:



The set E carries a group structure '+' with a neutral element O_E. The set of points with coordinates in $\mathbb{F}_q$ is denoted by $E(\mathbb{F}_q)$.

Maps between Objects: **Isogenies**

Let E, E' be two elliptic curves over $\mathbb{F}_q$. An isogeny is a rational map

$$\begin{aligned} \phi : E &\longrightarrow E' \\ (x, y) &\mapsto (f(x, y), g(x, y)) \end{aligned}$$

with $f, g : E \longrightarrow \overline{\mathbb{F}}_q$ rational functions such that ϕ respects the group structures on E and E', i.e.

$$\phi(P + Q) = \phi(P) + \phi(Q)$$

for all $P, Q \in E$. Note:

- isogenies are surjective, i.e. $\phi(E) = E'$
- a finite subgroup of E defines an isogeny, and vice versa.

Figure 1: Basic ingredients of SIDH.

The SIDH protocol utilizes the fact that there is a one-to-one correspondence between finite subgroups of an elliptic curve E and isogenies $\phi : E \longrightarrow E'$ starting from E. To construct an isogeny from a finite subgroup one can use Velu's formulas which provide an explicit equation for the target curve E'. To obtain a finite subgroup out of ϕ one can simply take the kernel of ϕ, i.e. $\ker(\phi) := \{(x, y) \in E \,|\, \phi(x, y) = O_{E'}\} \subseteq E$.
Then, choose a prime $p = \ell_A^{e_A} \ell_B^{e_B} \cdot f \pm 1$ where ℓ_A and ℓ_B are small primes and f is a cofactor. A supersingular elliptic curve E_0 can be constructed over the finite field $\mathbb{F}_{p^2}$ with these parameters. We choose to use supersingular elliptic curves for two reasons: security which we explain in the next section, and faster computation of isogenies since

	EC-DH	SI-DH
PUBLIC	E elliptic curve Basepoint $P_0 \in E$	$\mathcal{E} = \{ E \mid E \text{ elliptic curve } \}$ 'Basecurve' $E_0 \in \mathcal{E}$
PRIVATE KEYS	$n_A \in \mathbb{Z}$ $n_B \in \mathbb{Z}$	$\phi_A : E_0 \longrightarrow E_A$ isogeny $\phi_B : E_0 \longrightarrow E_B$ isogeny
PUBLIC KEYS	$Q_A = P_0^{n_A} \in E$ $Q_B = P_0^{n_B} \in E$	$E_A = \phi_A(E_0) \in \mathcal{E}$ $E_B = \phi_B(E_0) \in \mathcal{E}$
SHARED KEY	$Q_{AB} = (Q_B)^{n_A}$ $Q_{BA} = (Q_A)^{n_B}$ $Q_{AB} = Q_{BA} =: S_P$	$E_{AB} = \phi'_A(E_B)$ (!not ϕ_A!) $E_{BA} = \phi'_B(E_A)$ (!not ϕ_B!) $E_{AB} \cong E_{BA} =: S_E$
KDF	E.g x-coordinate of $S_P \in \mathbb{F}_q$	$j(S_E) \in \mathbb{F}_q$

Table 1: Analogy between ECDH and SIDH with participants A and B.

E_0 has smooth order. The cardinality of E_0 is $|E_0(\mathbb{F}_{p^2})| = (\ell_A^{e_A}\ell_B^{e_B} \cdot f)^2$. Moreover, the torsion elements $E_0[\ell_A^{e_A}] := \{P \in E_0 \mid \ell^{e_A} P = O_E\}$ of supersingular elliptic curves have coordinates in $\mathbb{F}_{p^2}$. As a result, the computations of isogenies are fast (compare [DJP14], section 2). Since $\ell_A \not| p$ and $\ell_B \not| p$ we have that

$$E[\ell_A^{e_A}] = \mathbb{Z}/\ell_A^{e_A}\mathbb{Z} \oplus \mathbb{Z}/\ell_A^{e_A}\mathbb{Z} = \langle P_A, Q_A \rangle$$
$$E[\ell_B^{e_B}] = \mathbb{Z}/\ell_B^{e_B}\mathbb{Z} \oplus \mathbb{Z}/\ell_B^{e_B}\mathbb{Z} = \langle P_B, Q_B \rangle$$

with generators P_A, Q_A, P_B, Q_B which have smooth order and are defined over $\mathbb{F}_{p^2}$.

SIDH Protocol: The supersingular elliptic curve $E_0/\mathbb{F}_{p^2}$, the basis $\{P_A, Q_A\}$ for $E[\ell_A^{e_A}]$, and the basis $\{P_B, Q_B\}$ for $E[\ell_B^{e_B}]$ are all public.

SECRETE KEYS:

I. A secretly chooses $m_A, n_A \in \mathbb{Z}/\ell_A^{e_A}\mathbb{Z}$, not both divisible by ℓ_A, and computes an isogeny $\phi_A : E_0 \longrightarrow E_A$ with kernel $\langle m_A P_A + n_A Q_A \rangle$.

II. B secretly chooses $m_B, n_B \in \mathbb{Z}/\ell_B^{e_B}\mathbb{Z}$, not both divisible by ℓ_B, and computes an isogeny $\phi_B : E_0 \longrightarrow E_A$ with kernel $\langle m_B P_B + n_B Q_B \rangle$.

PUBLIC KEYS:

I. A publishes E_A and the points $\phi_A(P_B), \phi_A(Q_B)$ (but keeps ϕ_A secret).

I. B publishes E_B and the points $\phi_B(P_A), \phi_B(Q_A)$ (but keeps ϕ_B secret).

SECRETE SHARED KEY:

I. A computes an isogeny $\phi'_A : E_A \longrightarrow E_{AB}$ with kernel $\langle m_A\phi_B(P_A) + n_A\phi_B(Q_A) \rangle$.

II. B computes an isogeny $\phi'_B : E_B \longrightarrow E_{BA}$ with kernel $\langle m_B\phi_A(P_B) + n_B\phi_A(Q_B) \rangle$.

III. E_{AB} and E_{BA} are isomorphic (as $\phi'_A \circ \phi_A$ and $\phi'_B \circ \phi_B$ are isogenies with the same kernel). Hence, $j(E_{AB}) = j(E_{BA}) \in \mathbb{F}_{p^2}$ serves as a shared secret key.

Note that the constructions of the elliptic curves E_{AB} and E_{BA} do not result in the same elliptic curve, only curves which are *isomorphic* (see [Was08], Proposition 12.12). To obtain a secret key, one can make use of a well-known function in number theory, the j-function $j : \mathcal{E} \longrightarrow \mathbb{F}_q$ defined by

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

for $E : y^2 = x^3 + Ax + B$ which has the property that two isomorphic elliptic curves have the same j-value - in our case $j(E_{AB}) = j(E_{BA}) \in \mathbb{F}_{p^2}$!

4 Security and Performance Aspects of SIDH

Shor's algorithm can only be applied to protocols which work in finite groups, such as the Elliptic Curve Diffie-Hellman protocol. The SIDH protocol works on a set which does not carry a group structure and uses isogenies to construct secret information instead of using group operations. Therefore, Shor's algorithm and its generalization by Boneh and Lipton cannot be applied to SIDH. Thus the 'I' in SIDH protects the protocol against Shor.

To explain the 'S' in SIDH one must remember that elliptic curves over finite fields fall into two categories – ordinary and supersingular elliptic curves (see [Was08], p. 130 for a definition). The main reason to work with supersingular elliptic curves – the S – is that Childs, Jao, and Soukharev' algorithm in [CJS14] can compute isogenies between ordinary elliptic curves in quantum subexponential time. Since ordinary curves are essential for their algorithm (in this case the endomorphism ring is commutative), their ideas cannot be applied to supersingular elliptic curves.

Security of SIDH: The mathematical problem the security level of SIDH can be related to is analogous to the Decisional Diffie-Hellman problem for elliptic curves – the Supersingular Decisional Diffie-Hellman problem (SSDDH). Its precise formulation can be found in Table 2 below. De Feo, Jao, and Plût were able to prove in [DJP14] that SIDH is secure in Canneti-Krawczyk's adversarial model if SSDDH is assumed to be computationally infeasible. It is conjectured that this problem is computationally infeasible, and the best known attacks on SSDDH to date are the following:

(i) To compute isogenies between supersingular elliptic curves:
On classical computers: Running time $O(p^{1/2})$ (Delfs and Galbraith).
On quantum computers: Running time $O(p^{1/4})$ (Biasses, Jao, and Sankar).

(ii) As the order of the subgroup defining the isogeny is public, the problem can be modeled as a 'claw problem':
On classical computers: Running time $O(p^{1/4})$ (Tani and Zhang).
On quantum computers: Running time $O(p^{1/6})$ (Tani and Zhang).

In all cases, the running time is exponential in the field size, and SSDDH is infeasible with all known strategies.

Supersingular Decisional Diffie-Hellman Problem

Let $E_A, E_B, \phi_A(P_B), \phi_A(Q_B), \phi_B(P_A)$, and $\phi_B(Q_A)$ be the public parameters and public keys as described in the previous section. Then: Given a tuple with probability 1/2 from the following two distributions:

- $\big(E_A, E_B, \phi_A(P_B), \phi_A(Q_B), \phi_B(P_A), \phi_B(Q_A), E_{AB}\big)$ with

$$E_{AB} \cong E_0/\langle m_A P_A + n_A Q_A, m_B P_B + n_B Q_B\rangle.$$

- $\big(E_A, E_B, \phi_A(P_B), \phi_A(Q_B), \phi_B(P_A), \phi_B(Q_A), E_C\big)$ with

$$E_C \cong E_0/\langle m'_A P_A + n'_A Q_A, m'_B P_B + n'_B Q_B\rangle$$

  where m'_A, n'_A, m'_B, n'_B are chosen randomly with the same restrictions as on m_A, n_A, m_B, n_B.

Decide from which distribution the tuple is sampled!

Table 2: SSDDH

Of course, these are theoretic considerations. Real-world implementation could cause other problems which will be revealed in time. Since there are (almost) no Quantum computers, there is very little experience in this field.

Implementation of Costello-Longa-Naehrig: In [CLN16] the authors present a full-fledged implementation of SIDH (see section 6 in loc.cit. for a compact summary). Their choices for the parameters are: $\ell_A = 2, \ell_B = 3$ and $p = 2^{372} \cdot 3^{239} - 1$. They use the elliptic curve $E_0/\mathbb{F}_{p^2}$ given by

$$E_0 : y^2 = x^3 + x$$

which is a supersingular Montgomery curve. The generators P_A, Q_A and P_B, Q_B are also given explicitly. The great benefits of their work can be summarized as follows:

- High-Speed Performance: 2.8 times faster than previous proposals which brings it into the range for practical use.
- Constant-Time Executions: this is important against side-channeling attacks.
- 128 Qubits quantum security and 192 Bits of classical security with a 564 Byte public key size.

Another interesting feature which they present in their article is the possibility to easily partner the SIDH with the classical ECDH. This *hybrid* scheme would be even stronger against classical attacks on a little extra cost. Of course, they do not use the same

supersingular Montgomery curve for ECDH which is vulnerable to the Weil-pairing attack since supersingular elliptic curves have small embedding degrees (see here, Appendix A). They also propose a strategy to validate public keys in unauthenticated environments. In ECDH this is a little effort - one has to check whether a point lies on the given elliptic curve which can be done fast. For isogenies a more expensive computation is needed.

In comparison to other quantum-secure algorithms the public key size of only 564 Bytes – and it can even be smaller [CJL+16] – makes their SIDH implementation very attractive and encourages to further studies in PQC.

References

[BL95] D. Boneh and R. J. Lipton, *Quantum cryptanalysis of hidden linear functions*, Advances in cryptology - CRYPTO '95, Springer-Verlag, 1995, pp. 424–437.

[CJL+16] C. Costello, D. Jao, P. Longa, M. Naehrig, J. Renes, and D. Urbanik, *Efficient compression of SIDH public keys*, Cryptology ePrint Archive, Report 2016/963, 2016, `http://eprint.iacr.org/2016/963`.

[CJS14] A. Childs, D. Jao, and V. Soukharev, *Constructing elliptic curve isogenies in quantum subexponential time.*, J. Math. Cryptol. **8** (2014), no. 1, 1–29.

[CLN16] C. Costello, P. Longa, and M. Naehrig, *Efficient algorithms for supersingular isogeny Diffie-Hellman*, Cryptology ePrint Archive, Report 2016/413, 2016, `http://eprint.iacr.org/2016/413`.

[Deu85] D. Deutsch, *Quantum theory, the Church-Turing principle and the universal quantum computer*, Proceedings of the Royal Society of London Series A **400** (1985), 97–117.

[DJP14] L. De Feo, D. Jao, and J. Plût, *Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies.*, J. Math. Cryptol. **8** (2014), no. 3, 209–247.

[Sho97] P. W. Shor, *Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*, SIAM J. Comput. **26** (1997), 1484–1509.

[Was08] L. C. Washington, *Elliptic Curves: Number Theory and Cryptography*, Discrete Mathematics and Its Applications, Chapman and Hall/CRC, 2008.