Introduction to Supersingular Isogeny Diffie-Hellman

Pedro Miguel Sosa

March 18, 2017

Abstract

This paper aims to give researchers an approachable introduction to the Supersingular Isogeny Diffie-Hellman key exchange (SIDH-KEX). SIDH is one of the few post-quantum key exchange algorithms based on elliptic curves that has shown potential due to its small key sizes and quick timing. This paper will briefly describe the history and motivation of SIDH. Furthermore, we will present a simple construction of the algorithm along with the mathematical background needed to understand it. Lastly, we will present the recommended parameters to use for potential implementations.

1 Background

As studies in the area of quantum computing seem to yield promising results, cryptography researchers are seeking new cryptosystems that could withstand attacks from such machines. While quantum computers do not affect secret key cryptography as much[1], most of the popular public key cryptosystems become vulnerable [1].

Most popular public key cryptosystems used today rely on the hardness of the factorization problem. For classical machines, the best known attacks would take exponential time, however, quantum computers are able to do factoring in polynomial time using Shor's algorithm.

As a result, there has been an important push towards the study and implementation of Post-Quantum cryptographic protocols in these past few years. This has yielded many different Post-Quantum Cryptographic (PQC) schemes based on lattices, codes, hashes, multivariate equations, and most recently elliptic curves.

This paper will focus on Supersingular Isogenies Diffie-Hellman (SIDH), which is the one of the few PQC algorithm based on elliptic curves. It was created in 2011 by De Feo, Jao and Plut [2] and has since been implemented and optimized by others. In 2016, researchers at Microsoft published a version of SIDH which ran in constant time and used public keys of size 564 bytes, showing that SIDH had "strong potential as a PQC candidate"[3].

2 Security and Efficiency

Security Although both ECDH and SIDH use elliptic curves, their underlying problem is different. ECDH's hardness rests upon the discrete logarithm problem, whereas SIDH's hardness rests up the difficulty of finding the isogeny mapping between two supersingular elliptic curves with the same number of points.

It has been proven [4, 5] that SIDH's security against classical computers is $O(p^{\frac{1}{4}})$, while the security against quantum computers has been theorized to be $O(p^{\frac{1}{6}})$.

Efficiency SIDH has become substantially more efficient since its release in 2011. The latest implementations published in 2016 runs in constant time and has a public key size of 564 bytes. This key size is among the smallest in comparison to other PQC

[1] It is enough for most secret key cryptographic schemes to increase their key sizes to provide protection against quantum adversaries.

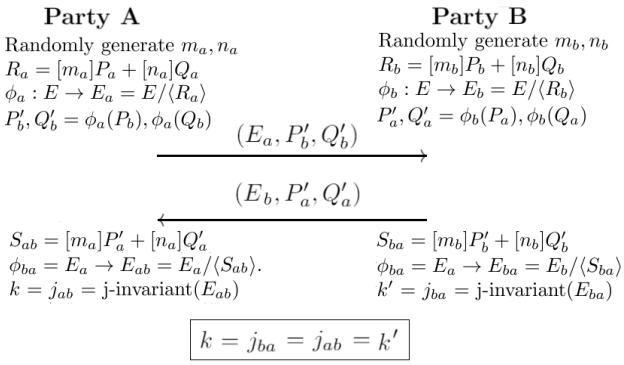

Figure 1: Visual Representation of SIDH-KEX

key exchange alternatives. This latest implementation has been added to the Open Quantum Safe's "OpenSSL" project to be used for Internet applications [2].

3 Preliminaries

In this section we will discuss some of the mathematical constructs and finer details necessary to further understand the SIDH-KEX.

Elliptic Curves Similar to other elliptic curve cryptographic schemes, we will assume our chosen elliptic curve E over F_{p^2} to be non-singular and of the Weierstrass form: $y^2 = x^3 + ax + b$. As such, all point addition and multiplication must be calculated by using the functions defined for these types of curves.

Isogeny An isogeny is a surjective and homomorphic structure preserving function that maps two groups together. In the case of elliptic curves case, a isogeny ϕ will map points on the domain curve E to points on a co-domain curve E'. These isogenies are calculated using Vélu's formulas [6] as $\phi : E \to E/\kappa$, where κ is defined as the kernel.

[2]Repository: github.com/open-quantum-safe/openssl

j-Invariant The j-invariant is a descriptor that can be computed for any particular curve using said curves parameters. Most importantly, isomorphic curves will always share the same j-invariant value. The exact equation for the j-invariant will vary depending on the underlying form of the elliptic curve. In the case of elliptic curves in the Weierstrass form, the j-invariant is calculated as follows:

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

Supersingular curves While it might seem a bit confusing, supersingular curves are non-singular elliptic curves as one would expect to find in other elliptic curve scheme. The term *"supersingular"* actually refers to the fact that they have "singular" values of the j-invariant and its Hasse invariant is 0. Refer to section 4.3 for a concrete curve suggestion.

4 Supersingular Isogeny Diffie-Hellman

Intuitively, the SIDH key exchange works by having both parties generate a secret key, which will be an isogeny based on the known public curve E and points P_a, Q_a, P_b, Q_b (which reside on E). Then each party will create and exchange a new curve derived from their secret isogenies. This new curves can be considered the "public keys". Afterwards, by merging their "private key" isogenies and the other party's "public key" curve, each party will be able to generate a final curve. For both parties, these final curves will have the same j-invariant (since they are isogenous with respect to each other). As such the value of this j-invariant becomes the shared secret.

4.1 Public Parameters

Initially there will be 4 global public parameters:
- A prime p
- A supersingular elliptic curve E over F_{p^2}
- Four fixed points P_a, Q_a, P_b, Q_b on E

4.2 Key Exchange

- Party A
1. Randomly generate m_a, n_a
2. $R_a = [m_a]P_a + [n_a]Q_a$
3. $\phi_a : E \rightarrow E_a = E/\langle R_a \rangle$
4. $P'_b, Q'_b = \phi_a(P_b), \phi_a(Q_b)$
5. Send (E_a, P'_b, Q'_b) to Party B
6. Receive (E_b, P'_a, Q'_a) from Party B
7. Calculate $S_{ab} = [m_a]P'_a + [n_a]Q'_a$
8. $\phi_{ab} = E_a \rightarrow E_{ab} = E_a/\langle S_{ab} \rangle$.
9. Compute $k = j_{ab} = \text{j-invariant}(E_{ab})$

- Party B
1. Randomly generate m_b, n_b
2. $R_b = [m_b]P_b + [n_b]Q_b$
3. $\phi_b : E \rightarrow E_b = E/\langle R_b \rangle$
4. $P'_a, Q'_a = \phi_b(P_a), \phi_b(Q_a)$
5. Send (E_b, P'_a, Q'_a) to Party A
6. Receive (E_a, P'_b, Q'_b) from Party A
7. Calculate $S_{ba} = [m_b]P'_b + [n_b]Q'_b$
8. $\phi_{ba} = E_a \rightarrow E_{ba} = E_b/\langle S_{ba} \rangle$.
9. Compute $k' = j_{ba} = \text{j-invariant}(E_{ba})$

Notice that since both curves are isogneous to each other, they will have the same j-invariant. As such, the shared key k calculated by both parties will be the same.

4.3 Parameter Selection

The latest work by Costello et al. [3], defined the curve $E = y^2 + x^3 + x$ and the prime $p = 2^{372} \cdot 3^{239} - 1$. Furthermore they established the four points on E to be:
- $P_a = [3^{239}](11, \sqrt{11^3 + 11})$
- $Q_a = \tau(P_a)$
- $P_b = [2^{372}](6, \sqrt{6^3 + 6})$
- $Q_b = \tau(P_b)$

where τ is a distortion map from $E(F_{p^2}) \rightarrow E(F_{p^2}) : (x, y) \rightarrow (-x, iy)$. This was done purposely so as to avoid having to store both Q_a and Q_b in memory and instead derive them from their respective P_i points.

5 Future Work

While SIDH seem very promising, there are still some issues that remain to be solved. Current implementations assume both parties are honest peers, and thus, an misbehaving/attacking party could leak information and weaken then security of the scheme. Another issue is the lack of crypto-analitic research done for this scheme, as the literature in which one can rely is rather limited.

Being a relatively new field, there are still many unknowns as to what supersingular isogenies can also be used for. Aside from key exchange, some researchers have also worked on applying supersingular curves to build digital signature schemes [7, 8].

However, with the recent promising developments in SIDH, work with supersingular curves is bound to gather more research in the upcoming years.

References

[1] D. J. Bernstein, "Introduction to post-quantum cryptography," in *Post-Quantum Cryptography*, pp. 1–14, Springer, 2009.

[2] D. Jao and L. De Feo, "Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies," in *QCrypto 2011*, pp. 19–34, Springer Berlin Heidelberg, 2011.

[3] C. Costello, P. Longa, and M. Naehrig, "Efficient algorithms for supersingular isogeny diffie-hellman," in *Annual Cryptology Conference*, pp. 572–601, Springer, 2016.

[4] C. Delfs and S. D. Galbraith, "Computing isogenies between supersingular elliptic curves over f_p," in *Designs, Codes and Cryptography*, vol. 78, pp. 425–440, Springer, 2016.

[5] J.-F. Biasse, D. Jao, and A. Sankar, "A quantum algorithm for computing isogenies between supersingular elliptic curves," in *International Conference in Cryptology in India*, pp. 428–442, Springer, 2014.

[6] J. Vélu, "Isogénies entre courbes elliptiques," in *CR Acad. Sci. Paris Sér. AB*, vol. 273, pp. A238–A241, 1971.

[7] D. Jao and V. Soukharev, "Isogeny-based quantum-resistant undeniable signatures," in *International Workshop on Post-Quantum Cryptography*, pp. 160–179, Springer, 2014.

[8] X. Sun, H. Tian, and Y. Wang, "Toward quantum-resistant strong designated verifier signature," in *International Journal of Grid and Utility Computing 4*, vol. 5, pp. 80–86, Inderscience Publishers Ltd, 2014.