Contents

1 Introduction **3**

1.1 Post-Quantum Cryptography 5

1.1.1 Shor's algorithm 5

1.1.2 NIST's Standardisation Challenge 6

1.2 This Thesis 7

2 Elliptic Curve Cryptography **8**

2.1 Important concepts 10

2.1.1 m-torsion group 10

2.1.2 Isomorphic curves 11

2.1.3 Isogenies 11

2.1.4 Endomorphism 13

2.1.5 Bases and Weil Pairing 14

2.2 Various optimisations 14

2.2.1 Montgomery Curves 14

2.2.2 Weil Pairing 16

2.2.3 Isogenies 17

2.3 Isogeny-based Cryptography 20

3 SIDH-based OT **21**

3.1 SIDH Key Exchange 21

3.2 Oblivious Transfer 22

4 Implementation details **24**

5 LATEX COMMANDS **25**

Bibliography **29**

Chapter 1

Introduction

The word "*cryptography*" derives from the Greek *kryptos*, meaning hidden. Its origin is usually dated in 2000 B.C., with the Egyptian practice of hieroglyphics which full meaning was only known to an elite few.
Formally, cryptography is the science of protecting information between two or more parties. It has advanced exponentially by taking advantage of the most developed mathematical theories and it has became more secure by relying on hard mathematical problems. Its algorithms are designed around computational complexity assumptions: it is theoretically possible to break a cryptosystem but it is infeasible in practice due to excessively long running times. These schemes are therefore said computationally secure.

Introduced in the early 19-th century, Modern Cryptography is a wider field than pure cryptography since it embraces additional properties vital to modern communications that can be summed up in:

- Authenticity - ensures that the information comes from the source it claims to be from;
- Confidentiality - ensure that information is not made available or disclosed to unauthorised individuals, entities, or processes;
- Integrity - ensures the information sent is exactly the same information received;

- Non Repudiation - provides proof of the integrity and origin of transmitted information.

In modern cryptography, Symmetric Key and Public Key encryption schemes are the best known and world-wide used. The symmetric key algorithms make use of a single key to both encrypt and decrypt messages. This key is shared by the communication parties hence the name symmetric. Public key algorithms use a pair of keys: the public key serves to encrypt messages, the private one to decrypt it. These special algorithms are also known as Asymmetric since encryption and decryption keys are different and not shared by the parties.
These schemes are commonly implemented in a 2-parties version in which the users Alice and Bob want to communicate without revealing their messages' content to others. Many protocols have been proposed to this purpose such that today we have RSA as the most world spread algorithm and Elliptic Curve Cryptography (ECC) as most recent and efficient scheme implemented.

In 1982 Richard Feynman observed that simulating quantum physics on classical computers seemed an intractable task. He then followed up by conjecturing that physical devices relying on quantum phenomena would have been good candidates for simulating quantum mechanics. Soon after, David Deutsch formalised the notion of a quantum Turing machine and showed that it was universal: a quantum Turing machine can simulate any quantum mechanical process with small overhead and independently of the substrate.
The basic idea behind a quantum computer is to exploit quantum bits, or qubits for short. As opposed to binary bits, qubits can exist in additional states in between the two binary states. This is defined as a superposition of the digital states.
With these premises it was undeniable wondering about whether quantum computers could solve natural computational problems faster than classical computers. To this question Shor answered in 1994 with a paper in which he presented polynomial-time quantum algorithms to solve the integer factorisation and discrete logarithm problems for which no efficient classical algorithms exist. The impact on modern public key cryptography is obvious: large enough quantum computers will break factorisation-based cryptosystems, such as RSA, as well as cryptosystems based on

the discrete logarithm, such as elliptic curve cryptosystems.

1.1 Post-Quantum Cryptography

With the term "*Post-quantum cryptography*" we refer to the science of protecting information against both quantum and classical attacks, as well as to the collection of tools that accomplish this task. Unfortunately, the adoption of post-quantum cryptography is not cost-free. The post-quantum hard problems, except for hash inversion, have been studied less than integer factorisation and the discrete logarithm problem. Consequently a post-quantum hard problem inevitably conveys less security assurance compared to a classic alternative. The reason is due to the greater potential of future improvements on quantum attacks. Additionally, many of the hard problems that hold promise of resisting attacks on quantum computers require far greater memory and bandwidth impeding their adoption into cryptosystems for low-cost devices.

1.1.1 Shor's algorithm

Theorised in 1994 by Peter Williston Shor, his algorithm stays at the core of post-quantum cryptography. By making use of a quantum computer, Shor's algorithm can efficiently solve both the integer factorisation and the discrete logarithm problem.

The idea behind Shor's algorithm [4, 5] is to utilise quantum computing to compare the phases of prime numbers as sinus waves to factorise great integers. Using number theory, the problem of number factorisation can be converted into a search for the period of a really long sequence, or rather, the length at which a sequence repeats itself. Then this periodic pattern is run through a quantum computer which functions as a computational interferometer creating an interference pattern. This will output the period, which can be processed using a classical computer, finally being able to factorise the initially given number.

1.1.2 NIST's Standardisation Challenge

Over the last decade there has been an intense research effort to find hard mathematical problems that would be at the same time quantum resistant and could be used to build new efficient cryptosystems.
Of great importance we can cite "*The Post-Quantum-Cryptography Standardization Challenge*" a competition started by the National Institute of Standards and Technology (NIST) in April 2016 accepting proposals for post-quantum protocols. It entered a first evaluation stage in November 2017 when NIST stopped accepting new algorithms for consideration. On 30 January 2019, the project went into the second evaluation stage, said *Round 2*, with NIST announcing 26 out of 69 original submissions as final competitors. This round may take up to 18 months before completion, after which there may be a third round and only then official standard algorithms will be chosen.
The final algorithms will be rated into a five level list which can be represented as below:

Level	Security	Reference protocol
I	Comparable to or greater than a block cipher with a 128-bit key against an exhaustive key search	AES128
II	Comparable to or greater than a 256-bit hash function against a collision search	SHA256
III	Comparable to or greater than a block cipher with a 192-bit key against an exhaustive key search	AES192
IV	Comparable to or greater than a 384-bit hash function against a collision search	SHA384
V	Comparable to or greater than a block cipher with a 256-bit key against an exhaustive key search	AES256

As a final note, most published post-quantum public-key schemes are focused on the following approaches [1]:

- Hash-based cryptography (e.g. Merkle's hash-tree public-key signature system);

- Multivariate quadratic-equations cryptography (e.g. HFE signature scheme);
- Lattice-based cryptography (e.g. NTRU encryption scheme);
- Code-based cryptography (e.g. McEliece encryption scheme, Niederreiter encryption scheme).

Another possible scheme is the newer "*Supersingular elliptic curve isogeny*"-based cryptography, central point of this thesis.

1.2 This Thesis

In this thesis we want to present a C implementation of a post-quantum protocol introduced in [9] by Vitse. The proposed algorithm exploits the supersingular isogeny problem, which is quantum resistant, for the key generation. Thereafter it exploits a zero knowledge protocol to exchange some data between the parties without revealing unnecessary information.
By implementing this protocol first we show that classical computers are still effective in a post-quantum reality or, viceversa, quantum computers are not compulsory when implementing post-quantum algorithms. Most importantly we adapted an open source library and created something new, useful and effective against the upcoming developments in the cryptography field.

Chapter 2

Elliptic Curve Cryptography

An elliptic curve is a cubic, smooth, projective, algebraic curve of genus one defined over a field $\mathbb{K}$ and has a K-rational point. It is important to note that said K-rational point can be the point at infinity expressed as $\mathcal{O} = [0 : 1 : 0]$. The curve has then a projective form described in equation 2.1 called Tate-Weierstrass extended form:

$$Y^2Z + a_1XYZ + a_3YZ^2 = X^3 + a_2X^2Z + a_4XZ^2 + a_6Z^3 \tag{2.1}$$

Another possible equation for an elliptic curve is the affine form which needs to be complemented by a smoothness condition and the point at infinity $\mathcal{O}$ as expressed in 2.2:

$$\textit{Weierstrass equation:} \begin{cases} y^2 = x^3 + ax + b \\ 4a^3 \neq 27b^2 \end{cases} \bigcup \{\mathcal{O}\} \tag{2.2}$$

We may now briefly list some of the most important properties for an elliptic curve from a cryptography point of view:

- Its point form a group with group law known as Point Addition. It allows to sum two different points $A = (x_A, y_A)$ and $B = (x_B, y_B)$ of the curve obtaining a third point $C = (x_C, y_C) = A + B$ computed as follows:

$$\begin{cases} m = \frac{y_A - y_B}{x_A - x_B} \\ x_C = m^2 - (x_A + x_B) \\ y_C = m(x_A - x_C) - y_A \end{cases}$$

- Whenever trying to sum a point to itself the group law behaves slightly different. In this case it is called Point Doubling and only the slope coefficient m changes to $m = \frac{y_A - y_B}{x_A - x_B}$;
- The curve points form an *abelian group* meaning that the group law is commutative, hence $C = A + B = B + A$;
- By joining both cases of group law it is possible to compute a scalar multiplication by a scalar k and a given point P. This operation can be written as $[k]P$;
- Given a cyclic generator G of an elliptic curve, its order is the minimum number k such that $[k]G = \mathcal{O}$. The curve generated by G is denoted as $E : \langle G \rangle$ and has cardinality equal to G's order;
- The discrete logarithm problem on an elliptic curve is then defined as: "*given the points G and $P = [k]G$, find the scalar k*";
- Finally an elliptic curve, defined in a finite field $\mathbb{F}_p$ with p prime greater than 3, having cardinality $p + 1$, is said *Supersingular* otherwise is said Ordinary.

In figure 2.1 we can see two common affine representation of an elliptic curve: on the left $a = -3$, $b = 1$, on the right $a = -2$, $b = 2$.

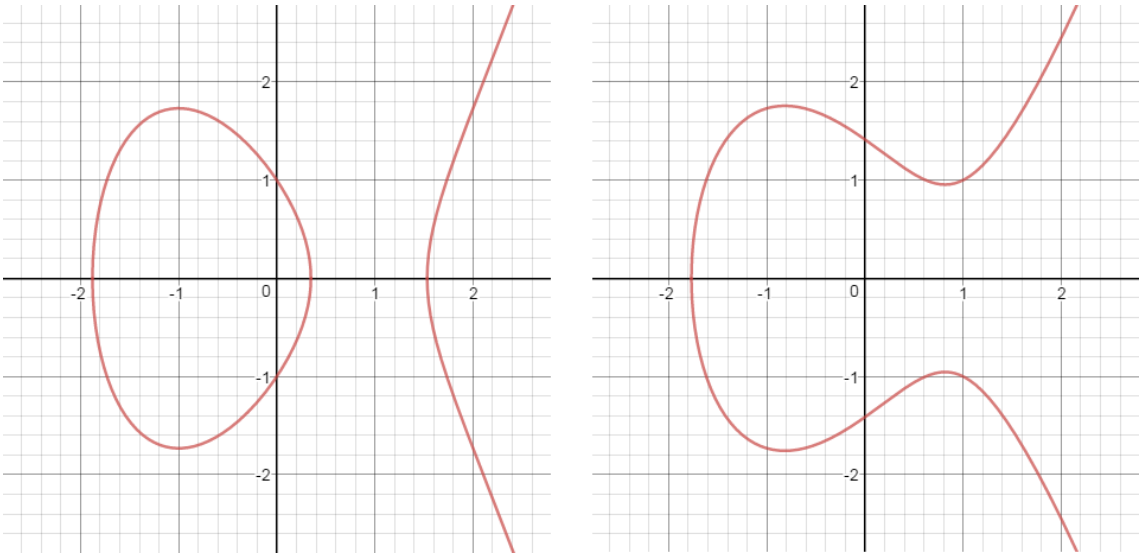

Figure 2.1: Two affine elliptic curves

2.1 Important concepts

In this section are listed many fundamental mathematical concepts that will be used in the next chapters.

2.1.1 m-torsion group

Given an elliptic curve E defined over a field K with characteristic p, an integer $m \neq 0$, the m-torsion group $E[m]$ is defined as follows:

$$E[m] \simeq (\mathbb{Z}/m\mathbb{Z})^2 \ \textit{if}\ p \nmid m$$

Otherwise we have $m = p^i$ hence:

$$E[p^i] \simeq \begin{cases} \mathbb{Z}/p^i\mathbb{Z} & \forall i \geq 0 \text{ if E is Ordinary} \\ \{\mathcal{O}\} & \forall i \geq 0 \text{ if E is Supersingular} \end{cases}$$

As consequence of above, a supersingular curve has no points of order p.

2.1.2 Isomorphic curves

Given two groups $(G, \circ)$, $(H, *)$, the function $\phi : G \to H$ is said homomorphism if holds:

$$\phi(g \circ h) = \phi(g) * \phi(h) \quad \forall g, h \in G$$

If the homomorphism is also bijective then it is called *Isomorphism*, it is represented as $G \simeq H$ and the two groups have the same algebraic structure.

j-invariant

Given the elliptic curve $E : y^2 = x^3 + ax + b$ its j-invariant is defines as follows:

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

Two elliptic curves are said *Isomorphic* over an algebraic closure $\overline{\mathbb{K}}$ if and only if they have the same j-invariant.

2.1.3 Isogenies

An isogeny is a surjective group morphism between two elliptic curves. Moreover, given two elliptic curves E, E' and the map $\phi : E \to E'$ acting on the two curves, the following hold:

- ϕ is a surjective group morphism said *isogeny*
- ϕ is a group morphism with finite kernel
- ϕ is an algebraic, non constant map of projective varieties which maps the point at infinity $\mathcal{O}$ of E on the point at infinity $\mathcal{O}'$ of E'.

Two elliptic curves E, E' are thus said *Isogenous* if there exists an isogeny between them. Moreover, the isogeny preserves the cardinality of each curve hence we can say that two elliptic curves are isogenous if and only if they have the same cardinality $\#E = \#E'$.

The isogeny equation can be computed via Velù's formulae: let $E : y^2 = x^3 + ax + b$ be an elliptic curve, P a generic point of E, $\langle K \rangle$ a cyclic group generated by K; let $E' \simeq E/\langle K \rangle$ is the quotient elliptic curve such that $\phi : E \to E'$ is the isogeny between the two curves, with kernel $\langle K \rangle$. The isogeny has equation:

$$\phi(P) = \left(x(P) + \sum_{Q \in \langle K \rangle \setminus \{\mathcal{O}\}} \Big(x(P+Q) - x(Q) \Big),\ y(P) + \sum_{Q \in \langle K \rangle \setminus \{\mathcal{O}\}} \Big(y(P+Q) - y(Q) \Big) \right) \tag{2.3}$$

Note that with $x(P)$ and $y(P)$ we define respectively the x and y coordinate of P. The image curve $E' \simeq E/\langle K \rangle : y^2 = x^3 + a'x + b'$ has parameters a', b' defined as follows:

$$a' = a - 5 \cdot \sum_{Q \in \langle K \rangle \setminus \{\mathcal{O}\}} \Big(3x(Q)^2 + a \Big)$$
$$b' = b - 7 \cdot \sum_{Q \in \langle K \rangle \setminus \{\mathcal{O}\}} \Big(5x(Q)^3 + 3ax(Q) + b \Big)$$

An example of isogeny is shown in figure 2.2 where the left curve $E/\mathbb{F}_{11} : y^2 = x^3 + x$ is mapped on the right one $E'/\mathbb{F}_{11} : y^2 = x^3 - 4x$. The kernel is the point $(0, 0)$ on E and the isogeny has parameters:

$$\phi(x, y) = \left(\frac{x^2 + 1}{x}, y \frac{x^2 - 1}{x^2} \right).$$

Degree

Following [6], let $\phi : E \to E'$ be an isogeny defined over a field $\mathcal{K}$, and let $\mathcal{K}(E)$, $\mathcal{K}(E')$ be the function fields of E, E'. By composing ϕ with the functions of $\mathcal{K}(E')$ we obtain a subfield of $\mathcal{K}(E)$ that we denote by $\phi * \mathcal{K}(E')$. The isogeny ϕ is said to be separable if the extension of function fields is. In this case its degree is equal to its kernel's: $deg\ \phi = \#ker\ \phi$.
Let E be an elliptic curve and G a finite subgroup of E then there is a *unique* elliptic curve $E' : E/G$ and a unique separable isogeny ϕ such that $ker\ \phi = G$ and $\phi : E \to E'$.

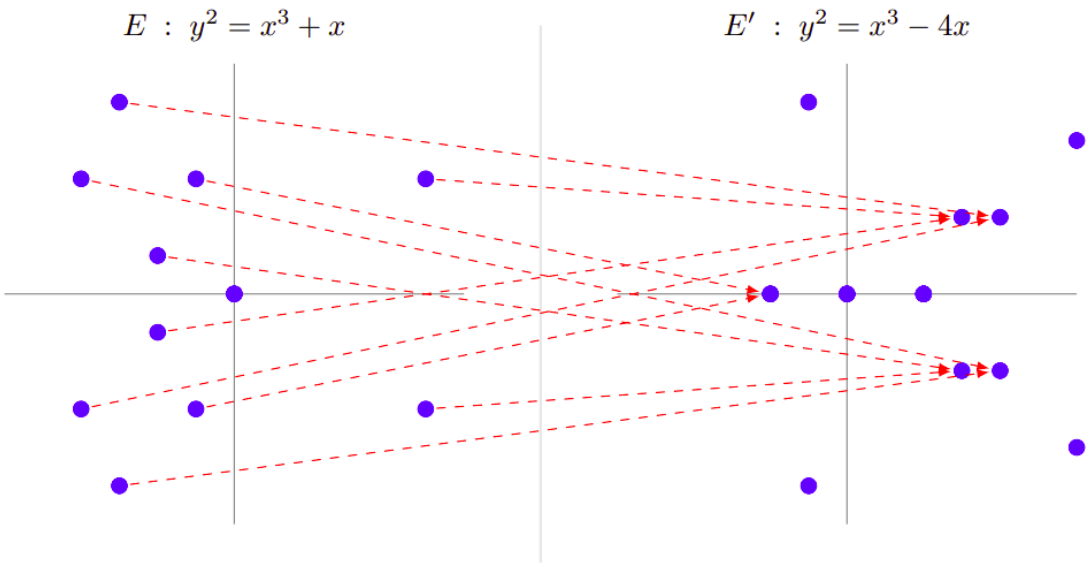

Figure 2.2: Isogeny mapping between E and E'

2.1.4 Endomorphism

Isogenies from a curve to itself are called *endomorphisms*. Given an elliptic curve E, a point P of E and an integer m, the scalar multiplication-by-m defined by $[m] : P \to [m]P$ is the endomorphism of E. Its kernel is exactly the m-th torsion subgroup $E[m]$.

Frobenius Endomorphism

Let E be an elliptic curve defined over a field $\mathcal{K}$ with q elements, its Frobenius endomorphism is the map $\pi : (X : Y : Z) \to (X^q : Y^q : Z^q)$.
It also holds:

- $ker\ \pi = \{\mathcal{O}\}$
- $ker(\pi - 1) = E(\mathcal{K})$

2.1.5 Bases and Weil Pairing

Given an elliptic curve E defined over a finite prime field $\mathbb{F}_p$, it is possible to find a pair (P, Q) of independent points of E such that they form a basis of said curve. If both points have the same order k then the set of all their linear combinations form an elliptic curve of order k denoted as $E[k] = \langle P, Q \rangle$.
It is possible to check for linear independence of a pair (P, Q) via Weil Pairing. This particular pairing is a map defined as $e : E[k] \times E[k] \to \mu_k$ where μ_k is the group of k-th root of unity. The Weil pairing returns one root of unity μ; the order k of μ is also the order of the pairing, consequently the input pair generates a cyclic group of order k.

2.2 Various optimisations

This section aims to sum up all the many optimisations applied.

2.2.1 Montgomery Curves

A Montgomery curve over $\mathbb{F}_q$ is a special form of an elliptic curve with affine equation:

$$M_{(A,B)} : By^2 = x(x^2 + Ax + 1) \tag{2.4}$$

where the parameters A, B are in $\mathbb{F}_q$ and satisfy $B \neq 0$, $A^2 \neq 4$. In projective coordinates $(X : Y : Z)$ with $x = X/Z$, $y = Y/Z$ the equation becomes:

$$M_{(A,B)} : BY^2Z = X(X^2 + AXZ + Z) \tag{2.5}$$

The latter has a unique point at infinity $\mathcal{O} = (0 : 1 : 0)$ and it is the **only** point where $Z = 0$.

There are many improvements on choosing a Montgomery curve over a simple Weierstrass elliptic curve. For a complete discussion about these special curves it is recommended to check [8]; for this thesis the most relevant speed up are:

- reduced number of operations for the group law;

- only x-coordinates operations are possible thus halving the overall operations when computing the scalar multiplication;
- the Point Addition can be further improved: when adding the points P, Q, a third point should be provided namely $Q - P$. In this case we talk about Differential Addition.

Other important improvements rely on a different curve equation and precomputing some constants used in scalar multiplication and j-invariant calculation [10]. First of all it is important to remember that the notation $(X_p : Z_p)$ represents a projective tuple in $\mathbb{P}^1$ over $\mathbb{F}_{p^2}$ and also the affine coordinate $x_p = X_p/Z_p$. Now, starting from the affine curve $E_a/\mathbb{F}_{p^2} : y^2 = x^3 + ax^2 + x$, we can write it into projective form and finally apply the equivalence $(A : C) \sim (a : 1)$ which transforms the curve into:

$$E : CBy^2 = Cx^3 + Ax^2 + Cx \tag{2.6}$$

This new equation allows to reduce the number of inversions, the most costly operation, with the little disadvantage of slightly increasing the total operations needed. The pair $(A_{24} : C_{24})$ denotes the projective form of the constant $(a - 2)/4$ in $\mathbb{P}^1$ which can be expressed as:

$$(A_{24} : C_{24}) \sim (a - 2 : 4)$$

Lastly the constants used in practice are:

$$\begin{aligned}(A_{24}^{+} : C_{24}) &\sim (A + 2C : 4C)\\ (A_{24}^{+} : A_{24}^{-}) &\sim (A + 2C : A - 2C)\\ (a_{24}^{+} : 1) \quad &\sim (A + 2C : 4C)\end{aligned}$$

j-invariant

The j-invariant of a standard projective Montgomery curve depends only on the parameter A and is computed as follows:

$$j(M_{(A,B)}) = \frac{256(A^2 - 3)^3}{A^2 - 4}$$

In implementation scenarios we use the equation 2.6 which requires the computation of $(A : C)$ to compute the new j-invariant:

$$j(M_{(A,B)}) = \frac{256(A^2 - 3C^2)}{C^4(A^2 - 4C^2)}.$$

2.2.2 Weil Pairing

The base algorithm for the Weil Pairing requires, roughly, $\mathcal{O}(n)$ operations which is greatly inefficient for large order bases; as an example, typical orders are 2^{250} and 3^{159}. We now show a faster way to run the algorithm as described in [7].
Let P, Q be two points both of order $m \cdot n$ so that they are in $E[mn]$; the n-th power of the pairing $e_{mn}(P, Q)$ can be computed as follows:

$$e_{mn}(P, Q)^n = e_m([n]P, [n]Q)$$

We now consider an example: let be $m = 4$, $n = 2^{370}$ such that $mn = 2^{372}$, it is easy to show that

$$e_{mn}(P, Q)^n = e_{4 \cdot 2^{370}}(P, Q)^{2^{370}} = e_4([2^{370}]P, [2^{370}]Q)$$

If P and Q have order 2^{372} then it is also true that $P' = [2^{370}]P$ and $Q' = [2^{370}]Q$ both have order 4.
Another speed up: all the involved checks are possible considering the only x coordinates of input basis.
After computing $x(P')$ and $x(Q')$ it is mandatory to check that $x(P') \neq x(Q')$ via a projective cross-multiplication. The last two steps consist in checking:

- Said $(X : Z) = x([2]P')$, then it must hold true that $Z \neq 0$;
- Said $(\overline{X} : \overline{Z}) = x([4]P')$, then it must hold true that $\overline{Z} = 0$.

Same checks have to be performed on Q'.
The pair (P, Q) is then a basis of order mn if it passes all checks.

2.2.3 Isogenies

We have seen in section 2.1.3 that, said $\langle K \rangle$ a cyclic group, the worst drawback of Velù's formulae is that they require $\mathcal{O}(\#\langle K \rangle)$ operations hence making their implementation intractable. A great improvement is possible when the order of K is smooth, meaning it is a power of a prime number, via composition of small degree isogenies [7, 9] obtaining a cost drop to $\mathcal{O}\Big(\log(\#\langle K \rangle) \cdot \log(\log(\#\langle K \rangle))\Big)$. Since typical applications use kernels of order 2^n or 3^m, said speed up is achievable.

Given an elliptic curve E, a cyclic subgroup $\langle R \rangle \subseteq E[l^e]$ of order l^e, with l prime, there is a unique isogeny $\phi_R : E \to E/\langle R \rangle$ of degree l^e defined over $\mathbb{F}_{p^2}$ with kernel $\langle R \rangle$. Such isogeny can be computed by composing e l-isogenies, in this way we iteratively compute $E_{i+1} \simeq E_i/\langle [l^{e-i-1}]R_i \rangle$ for $0 \leq i < e$. Note that the point R_i is an l^{e-i}-torsion point and the point $\langle [l^{e-i-1}]R_i \rangle$ has order l. The resulting isogeny is then $\phi = \phi_{e-1} \circ \ldots \circ \phi_0$ having degree l^e as required.
By doing so, the resulting structure is called *isogeny graph*, a (multi)-graph whose vertices are the j-invariants of isogenous curves and whose edges are the isogenies between them. In order to compute a large degree isogeny it is fundamental to know how to "navigate" the graph. Two concepts are required at this point: an isogeny walk and a traversing strategy.
The isogeny walk, as shown in figure 2.3, starts with the isogeny kernel R, of order l^e, at the root and descends via isogenies (right edges) and scalar multiplications (left edges). The figure also shows the equivalence between ϕ_i and $[l^{e-i}]R_i$.

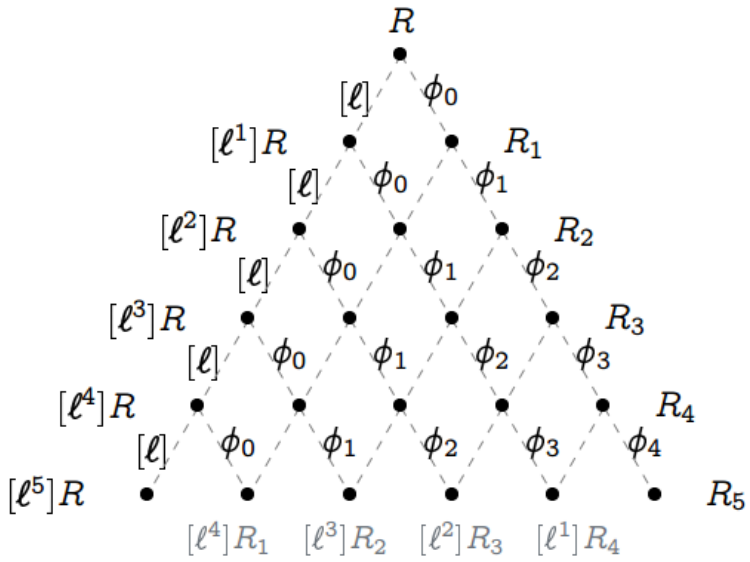

Figure 2.3: l-isogeny walks up to degree l^5

A traversing strategy is a non-cyclic path from the root to its leaves at the bottom of the walk. For the given walk there are seven well formed strategies as shown in figure 2.4.

Figure 2.4: Well formed strategies for $e = 4$

Among all strategies, there are optimal ones which minimise the number of operations and these can be precomputed offline. As result of all these improvements, large smooth degree isogeny computations, say l^e-isogeny, are done with complexity $\mathcal{O}(e \log e)$.

Small degree isogenies

In this section we show the alternative formulae to Velù's used in real applications. Since we are talking about 2-parties protocols, only two different isogenies should be studied: 2- and 3-isogenies. In the former case, a further speed up is possible if we consider 4-isogenies thus halving the total operations needed [11].

2-isogenies

Let $E_{A,B}$, $E_{A',B'}$ be two elliptic curves, $(x_2, y_2) \in E_{A,B}$ be a point of order 2 having $x_2 \neq 0$. Let $\phi_2 : E_{A,B} \to E_{A',B'}$ be the (unique) 2-isogeny with kernel $\langle (x_2, y_2) \rangle$. The image curve can be computed as follows:

$$E_{A',B'} : \left(A', B'\right) = \left(2 \cdot \left(1 - 2x_2^2\right), \quad Bx_2\right)$$

For any point $P = (x_P, y_P) \notin \langle (x_2, y_2) \rangle$ of $E_{A,B}$, its image $\phi_2 : (x_P, y_P) \mapsto (x_{\phi_2}, y_{\phi_2})$ can be computed as:

$$x_{\phi_2} = \frac{x_P^2 x_2 - x_P}{x_P - x_2}$$
$$y_{\phi_2} = y_P \cdot \frac{x_P^2 x_2 - 2x_P x_2^2 + x_2}{\left(x_P - x_2\right)^2}$$

4-isogenies

Let $E_{A,B}$, $E_{A',B'}$ be two elliptic curves, $(x_4, y_4) \in E_{A,B}$ be a point of order 4 having $x_4 \neq \pm 1$. Let $\phi_4 : E_{A,B} \to E_{A',B'}$ be the (unique) 4-isogeny with kernel $\langle (x_4, y_4) \rangle$. The image curve can be computed as follows:

$$E_{A',B'} : \left(A', B'\right) = \left(4x_4^4 - 2, \quad -x_4(x_4^2 + 1) \cdot B/2\right)$$

For any point $P = (x_P, y_P) \notin \langle (x_4, y_4) \rangle$ of $E_{A,B}$, its image $\phi_4 : (x_P, y_P) \mapsto (x_{\phi_4}, y_{\phi_4})$ can be computed as:

$$x_{\phi_4} = \frac{-x_P(x_P x_4^2 + x_P - 2x_4) \cdot (x_P x_4 - 1)^2}{(x_P - x_4)^2 \cdot (2x_P x_4 - x_4^2 - 1)}$$
$$y_{\phi_4} = y_P \cdot \frac{-2x_4^2(x_P x_4 - 1)\left(x_P^4(x_4^2 + 1) - 4x_P^3(x_4^3 + x_4) + 2x_P^2(x_4^4 + 5x_4^2) - 4x_P(x_4^3 + x_4) + x_4^2 + 1\right)}{(x_P - x_4)^3(2x_P x_4 - x_4^2 - 1)^2}$$

3-isogenies

Let $E_{A,B}$, $E_{A',B'}$ be two elliptic curves, $(x_3, y_3) \in E_{A,B}$ be a point of order 3. Let $\phi_3 : E_{A,B} \to E_{A',B'}$ be the (unique) 3-isogeny with kernel $\langle (x_3, y_3) \rangle$. The image curve can be computed as follows:

$$E_{A',B'} : \left(A', B'\right) = \Big(\left(Ax_3 - 6x_3^2 + 6\right) x_3, \quad Bx_3^2 \Big)$$

For any point $P = (x_P, y_P) \notin \langle (x_3, y_3) \rangle$ of $E_{A,B}$, its image $\phi_3 : (x_P, y_P) \mapsto (x_{\phi_3}, y_{\phi_3})$ can be computed as:

$$x_{\phi_3} = \frac{x_P(x_P x_3 - 1)^2}{(x_P - x_3)^2}$$
$$y_{\phi_3} = y_P \cdot \frac{(x_P x_3 - 1)(x_P^2 x_3 - 3x_P x_3^2 + x_P + x_3)}{(x_P - x_3)^3}$$

2.3 Isogeny-based Cryptography

Traditional Elliptic Curve Cryptography (ECC) relies its security on the Discrete Logarithm Problem (DLP) which can be stated as: "*Let E be an elliptic curve defined over $\mathbb{F}_p$ with p prime greater than 3. Given two points P, Q of said curve, find an integer k such that $Q = [k]P$*".
That said, Shor's algorithm presents a threat to DLP based cryptography since it can compute discrete logarithms in polynomial time. A quantum resistant solution involves the use of the *computational supersingular isogeny problem* stated as follows: "*given two supersingular elliptic curves E and E', find an isogeny ϕ such that $\phi : E \to E'$*". While for ordinary elliptic curves there exists a sub-exponential quantum attack, for the given problem does not exist thus making it post-quantum resistant.

Chapter 3

SIDH-based OT

The protocol studied and implemented in this thesis relies on an adaptation of Supersingular Isogeny Diffie-Hellman (SIDH) as a base structure for an Oblivious Transfer protocol. By joining these two protocols together it is possible to construct a post-quantum multi-party scheme.
First things first, we explain how SIDH and OT work, then how it is possible to merge them into a new procotol.

3.1 SIDH Key Exchange

The Supersingular Isogeny Diffie-Hellman is a protocol developed by Jao and De Feo in 2011 which offers a great ratio efficiency-security having keys smaller than other post-quantum protocols (lattice-based and code-based), moreover they are smaller than traditional Diffie-Hellman public keys.
The scheme begins with a shared supersingular elliptic curve and two parties, Alice and Bob, who get assigned two different torsion groups. After few isogenies and data exchanges both parties have two curves with the same j-invariant. The latter is then used a shared key hence used to encrypt and decrypt data between the parties. Let's now describe the protocol deeper in details.

All the supersingular elliptic curves used in SIDH are defined over $\mathbb{F}_{p^2}$ where $p = l_A^{e_A} l_B^{e_B} \pm 1$ prime, l_A and l_B small primes, e_A and e_B integers such that $l_A^{e_A} \approx l_B^{e_B}$.

Typical choices for e_A fall within the range $[100, 500]$ according to the security level desired. All SIDH implementations consider $e_A = 2$, $e_B = 3$ and so we will henceforth. In order to simplify the notation we will be using $e_A = n$ and $e_B = m$ and assign $l_A^{e_A} = 2^n$ to Alice, $l_B^{e_B} = 3^m$ to Bob.
Initial public parameters:

- A supersingular elliptic curve $E \subset \mathbb{F}_{p^2}$ with $p = 2^n 3^m \pm 1$ prime;
- A bases $(U, V) \subset E[2^n]$ in $\mathbb{F}_{p^2}$;
- A bases $(P, Q) \subset E[3^n]$ in $\mathbb{F}_{p^2}$.

The protocol proceeds as follows:

Alice	**Bob**
Chooses $x_A, y_A \in \mathbb{Z}/2^n\mathbb{Z}$ randomly, at least one of them coprime to 2	Chooses $x_B, y_B \in \mathbb{Z}/3^m\mathbb{Z}$ randomly, at least one of them coprime to 3
Computes $R_A = x_A \cdot U + y_A \cdot V$	Computes $R_B = x_B \cdot P + y_B \cdot Q$
Computes the curve $E_A \simeq E/\langle R_A \rangle$	Computes the curve $E_B \simeq E/\langle R_B \rangle$
Computes the isogeny $\phi_A : E \to E_A$	Computes the isogeny $\phi_B : E \to E_B$
$\longleftarrow$ Exchange their new curves E_A, E_B $\longrightarrow$	
Computes $P' = \phi_A(P)$, $Q' = \phi_A(Q)$	Computes $U' = \phi_B(U)$, $V' = \phi_B(V)$
$\longleftarrow$ Exchange their new points U', V', P', Q' $\longrightarrow$	
Computes $E_{BA} \simeq E_B/\langle x_A P' + y_a Q' \rangle$	Computes $E_{AB} \simeq E_A/\langle x_B U' + y_B V' \rangle$

At the end of the protocol both parties have $E_{AB} \simeq E_{BA}$ having the same j-invariant which can be used as shared secret.
Correctness:

3.2 Oblivious Transfer

The *oblivious transfer*, or **OT**, is a multi-party cryptography scheme in which two or more parties are involved. A typical OT application is the secure function eval-

uation where every party holds an input for a given function. In this scenario, the output should be computed in a way such that no party has to reveal unnecessary information about their input. Correctness of the protocol is usually proved with a zero knowledge proof.
The oblivious transfer has many different implementations each of them achieving different yet similar goals. The base idea is to send one of many pieces of information to a second party while the sender has no knowledge of which piece has been sent. A classical implementation is the *Rabin OT* in which Alice, with a probability of 1/2, sends a simple bit to Bob. This scheme leaves Alice "oblivious" of whether Bob has received it or not.

There exists another variation proposed by Shimon Even, Oded Goldreich and Abraham Lempel called "*1 out of 2 Oblivious Transfer*", often written as $\binom{1}{2}$-OT, which can easily be generalised to "*1 out of n OT*". In this variation, a party say B, receives one out of two (alternatively n) piece of information but the sender party, say A, does not know which piece B has received.
As shown in figure 3.1, A sends the pieces b_0, b_1 to B which, in turn, chooses a random integer c. At the end of the transfer B will have knowledge of the c-th piece b_c of A.

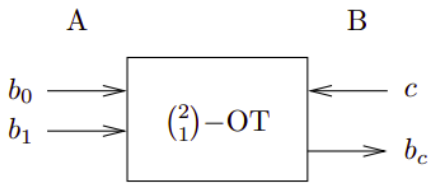

Figure 3.1: Simple $\binom{1}{2}$-OT

Chapter 4

Implementation details

Chapter 5

LATEX COMMANDS

1. Backslash ‘, alt+’
2. *corsivo*, ctrl+I
3. **grassetto**, ctrl+B
4. $$\begin{cases} c = E(m, K) & \text{Cifratura di } m \text{ in } c \\ m = D(c, K) & \text{Decifratura di } c \text{ in } m \end{cases}$$
5. Testo centrato:

$$S_d(C_A,\, K^S) = S_d[S_c(m_A,\, K^S),\, K^S] = m_A$$

6. Nota a piè di pagina[I]
7. Riferimento ad un capitolo/section
   labelNome Chapter/Section
8. Cita il riferimento
   label -¿
   refNome Chapter/Section

[I] Nota

9. Cita bibliografia
   citeNome bibitem

10. "*Questa roba scritta così*"

11. Aggiungi una figura:
    beginfigure[H]
    includegraphics[width=
    textwidth,center]NOME IMMAGINE
    captionCAPTION SOTTO L'IMMAGINE
    begincenter Testo aggiuntivo
    endcenter
    labelLABEL DI RIFERIMENTO DA PRENDERE CON
    ref
    endfigure

12. Tabella

Passo i	m_{i-1}	Operazione	m_i
1	1	Double	2
2	2	Add	3
3	3	Double	6
4	6	Double	12
5	12	Double	24
6	24	Add	25
7	25	Double	50
8	50	Double	100

13. Tabella 2

	Operazioni			Anni MIPS	
L	RSA	ECC	ECC/RSA	RSA	ECC
80	$3.8 \cdot 10^{13}$	$1.5 \cdot 10^{24}$	$3.9 \cdot 10^{10}$	1.2	$4.7 \cdot 10^{10}$
112	$1.9 \cdot 10^{18}$	$6.5 \cdot 10^{33}$	$3.4 \cdot 10^{15}$	$6.0 \cdot 10^{4}$	$2.0 \cdot 10^{20}$
128	$5.2 \cdot 10^{21}$	$4.3 \cdot 10^{38}$	$8.2 \cdot 10^{16}$	$1.6 \cdot 10^{8}$	$1.3 \cdot 10^{25}$
192	$6.0 \cdot 10^{31}$	$7.9 \cdot 10^{57}$	$1.3 \cdot 10^{26}$	$1.9 \cdot 10^{18}$	$2.5 \cdot 10^{44}$
256	$1.4 \cdot 10^{42}$	$1.5 \cdot 10^{77}$	$1.0 \cdot 10^{35}$	$4.4 \cdot 10^{28}$	$4.7 \cdot 10^{63}$

14. Algoritmo

Algorithm 1 Montgomery Ladder

for $i = j - 1$ to 0 **do**
  if $d_i = 1$ **then**
    $P_1 = P_1 + P_2$
    $P_2 = 2P_2$
  else
    $P_2 = P_1 + P_2$
    $P_1 = 2P_1$
  end if
end for
return P_1

15. Testo allineato

$$
\begin{aligned}
m_A &= (c_A)^{d_B} \bmod(n) && \text{Sostituiamo } c_A = (m_A)^{e_B} \bmod(n)\\
m_A &= [(m_A)^{e_B}]^{d_B} \bmod(n) && \text{Raccogliamo l'esponente di } m_A\\
m_A &= m_A^{e_B \cdot d_B} \bmod(n) && \text{Applichiamo: } e_B d_B = 1 + h\theta(n)\\
m_A &= m_A^{1+h\theta(n)} \bmod(n) && \text{Semplifichiamo l'esponente}\\
m_A &= m_A(m_A^{\theta(n)})^h \bmod(n) && \text{Applichiamo il teorema di Eulero}\\
m_A &= m_A(1)^h \bmod(n) && \text{Dato che } 1^h = 1 \text{ scriviamo}\\
m_A &= m_A \bmod(n) && \text{c.v.d.}
\end{aligned}
$$

16. Cambia item symbol in itemize

- Nuovo simbolo dell'itemize

Bibliography

[1] *Post Quantum Cryptography: Implementing Alternative Public Key Schemes on Embedded Devices*, by Stefan Heyse, October 2013.

[2] *Mathematical and Provable Security Aspects of Post-Quantum Cryptography*, by Alan Szepieniec, December 2018

[3] *Cybersecurity in an era with quantum computers: will we be ready?*, by Mosca M., Cryptology ePrint Archive, Report 2015/1075, 2015.

[4] *On the Development and Standardisation of Post-Quantum Cryptography. A Synopsis of the NIST Post-Quantum Cryptography Standardisation Process, its Incentives, and Submissions*, by Maja Worren Legernæs, Norwegian University of Science and Technology, Master of Science in Communication Technology, June 2018.

[5] *Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*, by Peter W. Shor, `https://arxiv.org/pdf/quant-ph/9508027.pdf`.

[6] *Mathematics of Isogeny Based Cryptography*, by Luca De Feo, École mathématique africaine, May 2017, Thiès, Senegal.

[7] *Efficient algorithms for supersingular isogeny Diffie-Hellman*, by Craig Costello, Patrick Longa, and Michael Naehrig, Microsoft Research, USA, 2016.

[8] *Montgomery curves and their arithmetic, The case of large characteristic fields*, by Craig Costello, Benjamin Smith, 2017.

[9] *Simple oblivious transfer protocols compatible with Kummer and supersingular isogenies*, by Vanessa Vitse, January 2019.

[10] *Isogeny based crypto: what's under the hood?*, by Luca De Feo, November 2018.

[11] *Supersingular Isogeny Key Encapsulation*, by David Jao, April 2019