Imaginary Quadratic Class Groups and a Survey of Time-Lock Cryptographic Applications*

Aron van Baarsen
aron.van.baarsen@cwi.nl

CWI, Cryptology Group, Amsterdam, The Netherlands

Abstract

Imaginary quadratic class groups have been proposed as one of the main hidden-order group candidates for time-lock cryptographic applications such as verifiable delay functions (VDFs). They have the advantage over RSA groups that they do *not* need a trusted setup. However, they have historically been significantly less studied by the cryptographic research community. This survey provides an introduction to the theory of imaginary quadratic class groups and discusses several considerations that need to be taken into account for practical applications. In particular, we describe the relevant computational problems and the main classical and quantum algorithms that can be used to solve them. From this discussion, it follows that choosing a discriminant $\Delta = -p$ with $p \equiv 3 \mod 4$ prime is one of the most promising ways to pick a class group $\mathrm{Cl}(\Delta)$ without the need for a trusted setup, while simultaneously making sure that there are no easy to find elements of low order in $\mathrm{Cl}(\Delta)$. We provide experimental data on class groups belonging to discriminants of this form, and compare them to the Cohen-Lenstra heuristics which predict the average behaviour of $\mathrm{Cl}(\Delta)$ belonging to a random *fundamental* discriminant. Afterwards, we describe the most prominent constructions of VDFs based on hidden-order groups, and discuss their soundness and sequentiality when implemented in imaginary quadratic class groups. Finally, we briefly touch upon the post-quantum security of VDFs in imaginary quadratic class groups, where the time on can use a fixed group is upper bounded by the runtime of quantum polynomial time order computation algorithms.

*Research partially funded by NWO/TKI Grant 628.009.014.

Contents

Introduction

A time-lock puzzle allows a sender to encrypt a message so that it can only be decrypted after a certain amount of time has passed [RSW96]. More precisely, to solve the "puzzle", the receiver needs to perform a prescribed amount of sequential work in order to retrieve the key that is used to encrypt the message. Time-lock puzzles have applications in, for example, delayed digital payments, sealed bid auctions and key escrow. Rivest et al.'s construction is based on the assumption that repeated squaring in a group of unknown order can not be sped up, even when a large amount of parallel computing power is available [RSW96]. In their construction, the group order can be used as a trapdoor to set up the "puzzle" that conceals the key.

A closely related primitive is that of a verifiable delay function (VDF) introduced by Boneh et al. [BBBF18]. A verifiable delay function is a function that can not be computed faster than a prescribed amount of time, even when a large amount of parallel computing power is available, but whose output can be verified quickly and publicly. VDFs have applications in, for example, public randomness beacons [BBBF18, Rab83, EFKP20, FKPS21], cryptographic timestamping [BBBF18, LSS20] and resource-efficient blockchains [BBBF18, Wes19, Pie19]. The most prominent VDF constructions are those introduced by Wesolowski [Wes19] and Pietrzak [Pie19]. Both constructions are based on the original time-lock puzzle from Rivest et al. [RSW96]. In essence, these VDF constructions consist of computing the repeated squaring of an element in a group of hidden-order and producing a proof which allows any party to efficiently verify the result.

RSA groups and class groups of imaginary quadratic number fields have been proposed as candidate hidden-order groups for these VDF constructions. Class groups have the benefit that they do not require a trusted setup, whereas RSA groups do. That is, one can sample an imaginary quadratic class group $\mathrm{Cl}(\Delta)$ by sampling a random negative fundamental discriminant Δ, which does not reveal any information about the group order except for its parity. On the other hand, generating an RSA modulus by sampling two large primes p, q and setting $N = pq$ reveals the group order $\phi(N) = (p-1)(q-1)$ in the process. This does however make RSA groups suitable for applications where a trapdoor VDF is needed, with the group order serving as a trapdoor to speed up the repeated squaring computation.

In this survey, we focus on imaginary quadratic (IQ) class groups, since they historically received less attention than RSA groups. In Section 1 we provide an introduction to IQ class groups and their connection with equivalence classes of binary quadratic forms. We furthermore introduce the relevant computational problems and cryptographic formalism used throughout the text. In Section 2 we describe several classical as well as quantum algorithms that can be used to solve the relevant computational problems in IQ class groups, and discuss how they influence the choice of discriminant. For example, for applications where it must not be easy to find elements of low order in the group, it is beneficial to pick a discriminant of the form $\Delta = -p$ with $p \equiv 3 \mod 4$ prime. Several other considerations are based on the class group heuristics by Cohen and Lenstra [CL84], which predict the average asymptotic behaviour of IQ class groups over all fundamental discriminants. In Section 3 we present some experimental class group data to verify, for relatively small discriminants, whether these heuristics hold up for the specific form of discriminant $\Delta = -p$ with $p \equiv 3 \mod 4$ prime. Finally, in Section 4 we discuss some time-lock cryptographic applications of IQ class groups, with the focus lying on VDFs.

In this discussion, we restrict ourselves to applications of hidden-order groups that can still have value in a "post-quantum setting". This might seem slightly paradoxical since there exist polynomial time quantum algorithms to compute the order of any finite abelian group [Sho97a, Kit96, ME98]. However, for applications where the group is only used for a short amount of time and a fresh random group can be sampled easily, hidden-order groups can still have value. Choosing this "timing parameter" in such a way that a fresh group is sampled before its order can be computed makes hidden-order groups especially suitable for time-lock cryptographic applications, even in a post-quantum setting. In Section 4.1.4, we give an estimate for the range of timing parameters suitable with a chosen discriminant size. For post-quantum applications where a fixed hidden-order group needs to be used for a longer amount of time, IQ class groups – or any finite abelian group for that matter – will most likely not be useful. The existence of a quantum polynomial time attack leads to an excessively large discriminant size being necessary to preserve security, which makes group operations too expensive for honest parties. For a survey of classical applications of IQ class

groups in cryptography we refer to [BH01, HM00, Ham02]. However, for applications where a fixed group is used for a longer amount of time, Dobson, Galbraith and Smith argue that even some classical attacks are underestimated when it comes to previously argued parameter choices [DGS20].

1 Preliminaries

1.1 Quadratic Number Fields

Let K be a number field and let $\mathcal{O}_K$ be the ring of integers of K. We recall that a *fractional $\mathcal{O}_K$-ideal I* is an $\mathcal{O}_K$-submodule of K such that $\alpha I \subseteq \mathcal{O}_K$ for some $\alpha \in K^*$. Since $\mathcal{O}_K$ is a Dedekind domain, every nonzero fractional ideal is invertible and the nonzero fractional $\mathcal{O}_K$-ideals form a group under ideal multiplication, which we denote by $\mathcal{I}_K$. The principal fractional ideals form a subgroup $\mathcal{P}_K \subseteq \mathcal{I}_K$. The *class group* of K is defined as the quotient $\mathrm{CL}_K = \mathcal{I}_K/\mathcal{P}_K$ and is a finite group whose order we call the *class number* of K and denote by h_K. Let $\mathcal{P}_K^+ \subseteq \mathcal{P}_K$ be the subgroup of principal fractional ideals $a\mathcal{O}_K$ whose generator $a \in K$ has a positive image $\sigma(a)$ under all real embeddings $\sigma : K \to \mathbb{R}$. The *narrow class group* of K is defined as the quotient $\mathrm{Cl}_K^+ = \mathcal{I}_K/\mathcal{P}_K^+$. Note that when K has no real embeddings, we have $\mathrm{Cl}_K^+ = \mathrm{Cl}_K$.

Now let d be a squarefree integer $\neq 1$ and let $K = \mathbb{Q}(\sqrt{d})$ be a quadratic number field. We recall that the ring of integers in K is given by

$$\mathcal{O}_K = \begin{cases} \mathbb{Z}[\frac{1+\sqrt{d}}{2}] & \text{if } d \equiv 1 \mod 4 \\ \mathbb{Z}[\sqrt{d}] & \text{if } d \equiv 2, 3 \mod 4, \end{cases}$$

and that the discriminant of K is given by

$$\Delta_K = \begin{cases} d & \text{if } d \equiv 1 \mod 4 \\ 4d & \text{if } d \equiv 2, 3 \mod 4. \end{cases}$$

Definition 1.1. An integer Δ is called a *fundamental discriminant* if Δ is the discriminant of a quadratic number field K. That is, Δ is unequal to 1 and either $\Delta \equiv 1 \mod 4$ and squarefree, or $\Delta \equiv 0 \mod 4$, $\Delta/4$ is squarefree and $\equiv 2, 3 \mod 4$.

The prime ideals in $\mathcal{O}_K$ satisfy the following property.

Proposition 1.2. *Let K be a quadratic number field of discriminant Δ_K, and let $\alpha \mapsto \alpha'$ be the nontrivial automorphism in $\mathrm{Gal}(K/\mathbb{Q})$. Let p be a prime in $\mathbb{Z}$, and let $\left(\frac{\Delta_K}{p}\right)$ denote the Kronecker symbol.*

(a) *If $\left(\frac{\Delta_K}{p}\right) = 0$ (i.e. $p|\Delta_K$), then $p\mathcal{O}_K = \mathfrak{p}^2$ for some prime ideal $\mathfrak{p}$ of $\mathcal{O}_K$ (i.e. p* ramifies completely *in K).*

(b) *If $\left(\frac{\Delta_K}{p}\right) = 1$, then $p\mathcal{O}_K = \mathfrak{p}\mathfrak{p}'$, where $\mathfrak{p} \neq \mathfrak{p}'$ are prime in $\mathcal{O}_K$ (i.e. p* splits completely *in K).*

(c) *If $\left(\frac{\Delta_K}{p}\right) = -1$, then $p\mathcal{O}_K$ is prime in $\mathcal{O}_K$ (i.e. p is* inert *in K).*

Proof. See [Cox89, Proposition 5.16.]. □

When Δ is a fundamental discriminant, there is a correspondence between the ideal class group of the corresponding quadratic number field K and the form class group of primitive positive definite binary integer-valued quadratic forms of discriminant Δ (see Sections 1.2 and 1.3). For non-fundamental discriminants, we need to introduce the notion of non-maximal orders in a quadratic number field.

An *order* in a quadratic number field K is a subring $\mathcal{O} \subset K$ which is a free $\mathbb{Z}$-module of rank 2. The ring of integers $\mathcal{O}_K$ is the maximal order of K, that is, any order $\mathcal{O}$ is contained in it. The *conductor* of $\mathcal{O}$ is defined as the index $f := [\mathcal{O}_K : \mathcal{O}]$, for which it holds that [Cox89, Lemma 7.2.]

$$\mathcal{O} = \mathbb{Z} + f\mathcal{O}_K.$$

The ring of integers always has conductor 1. If $d \equiv 1 \mod 4$ is negative, then the order $\mathbb{Z}[\sqrt{d}] \subset \mathbb{Z}[(1+\sqrt{d})/2]$ has conductor 2. Let $\{\alpha, \beta\}$ be a $\mathbb{Z}$-basis for $\mathcal{O} \subset K$. Then the *discriminant* of $\mathcal{O}$ is defined as

$$\Delta_{\mathcal{O}} := \left(\det \begin{pmatrix} \alpha & \beta \\ \sigma(\alpha) & \sigma(\beta) \end{pmatrix}\right)^2 \in \mathbb{Z}$$

where σ is the non-trivial element of $\mathrm{Gal}(K/\mathbb{Q})$. We therefore see that

$$\Delta_{\mathcal{O}} = f^2 \Delta_K,$$

Moreover, note that both $K = \mathbb{Q}(\sqrt{\Delta_{\mathcal{O}}})$ and $\mathcal{O} = \mathbb{Z} + \sqrt{\Delta_{\mathcal{O}}/\Delta_K} \cdot \mathcal{O}_K$ are determined uniquely by the discriminant $\Delta_{\mathcal{O}}$.

As opposed to fractional ideals in the full ring of integers $\mathcal{O}_K$, fractional ideals in $\mathcal{O}$ are not always invertible. For example, the ideal $(2, 1+\sqrt{-3}) \subset \mathbb{Z}[\sqrt{-3}]$, which can for example be seen by the Kummer-Dedekind theorem. Recall that a fractional $\mathcal{O}$-ideal $\mathfrak{a}$ is invertible if there there exists a fractional $\mathcal{O}$-ideal $\mathfrak{b}$ such that $\mathfrak{ab} = \mathcal{O}$ (or, equivalently, when it is projective as a $\mathcal{O}$-module). The set $\mathcal{I}_{\mathcal{O}}$ of invertible fractional ideals of $\mathcal{O}$ again forms a group under ideal multiplication. It is clear that any principal fractional $\mathcal{O}$-ideal is invertible, and that these form a subgroup $\mathcal{P}_{\mathcal{O}} \subset \mathcal{I}_{\mathcal{O}}$. Analogous to the definition of the class group of K, the *(ideal) class group* of $\mathcal{O}$ is defined as $\mathrm{Cl}_{\mathcal{O}} = \mathcal{I}_{\mathcal{O}}/\mathcal{P}_{\mathcal{O}}$. Since orders are uniquely determined by their discriminant, we will also denote $\mathrm{Cl}(\Delta)$ for the class group $\mathrm{Cl}_{\mathcal{O}}$ of the order $\mathcal{O}$ of discriminant $\Delta_{\mathcal{O}} = \Delta$.

Recall that the *norm* of an element $\alpha \in K$ is given by the product of the images of α under the distinct embeddings $\sigma : K \hookrightarrow \mathbb{C}$, that is

$$N(\alpha) = \prod_{\sigma \in \mathrm{Gal}(\mathbb{C}/K)} \sigma(\alpha).$$

Hence for $\alpha = a + b\sqrt{d} \in \mathbb{Q}(\sqrt{d})$ an element of a quadratic field, we have

$$N(\alpha) = a^2 - db^2.$$

The norm of an integral ideal $\mathfrak{a} \subseteq \mathcal{O}$ is defined as the cardinality of the quotient $\mathcal{O}/\mathfrak{a}$. That is,

$$N(\mathfrak{a}) = \#(\mathcal{O}/\mathfrak{a}).$$

We can extend this definition to fractional ideals $\mathcal{I}_K$ in the natural way, since any fractional $\mathcal{O}_K$-ideal $\mathfrak{a}$ can be factored uniquely as a product of nonzero prime ideals of $\mathcal{O}_K$. Under this extension, the norm of the principal ideal $\alpha\mathcal{O}_K$ agrees with the norm of α for $\alpha \in K^*$, that is

$$N(\alpha\mathcal{O}_K) = |N(\alpha)|.$$

Let $\mathcal{O}$ be an order of discriminant Δ in a quadratic number field and let $\omega = \frac{\Delta + \sqrt{\Delta}}{2}$. Then it is straightforward to see that $\mathcal{O} = \mathbb{Z} + \omega\mathbb{Z}$. Moreover, any integral ideal $\mathfrak{a}$ of $\mathcal{O}$ can be written as

$$\mathfrak{a} = a\mathbb{Z} + (b + c\omega)\mathbb{Z}$$

with a, b and c integers satisfying $c \mid a$, $c \mid b$ and $0 \leq b < a$, and the norm of $\mathfrak{a}$ is given by $N(\mathfrak{a}) = ac$ [Coh93, Proposition 5.2.1.]. We will also denote such ideals as $(a, b + c\omega)$. An integral ideal of $\mathcal{O}$ is called *primitive* if $c = 1$.

1.2 Binary Quadratic Forms

Definition 1.3. A binary integer-valued quadratic form is a homogeneous degree 2 polynomial

$$Q(X, Y) = aX^2 + bXY + cY^2 \in \mathbb{Z}[X, Y].$$

Q is called *primitive* if $\gcd(a, b, c) = 1$. The *discriminant* of Q is the integer $\Delta(Q) := b^2 - 4ac$. We say that Q is *regular* if $\Delta(Q) \neq 0$, and *singular* otherwise.

Remark. Since we will only be dealing with regular binary integer-valued quadratic forms in this survey, we will simply refer to these as *quadratic forms*.

The matrix $B = \left(\begin{smallmatrix} a & b/2 \\ b/2 & c \end{smallmatrix}\right)$ is called the *Gram matrix* of Q, and we have the relation

$$Q(X,Y) = \begin{pmatrix} X & Y \end{pmatrix} B \begin{pmatrix} X \\ Y \end{pmatrix}.$$

If the entries of B are integers, we say that Q is *matrix-integral* (or, simply, *integral*). Furthermore, note that $\Delta(Q) = -4 \cdot \det(B)$. We say that a quadratic form Q *represents* an integer $m \in \mathbb{Z}$ if there exist $x, y \in \mathbb{Z}$ such that

$$Q(x,y) = m.$$

Definition 1.4. Two quadratic forms Q, Q' are said to be equivalent if there exists an invertible matrix $T = \left(\begin{smallmatrix} p & q \\ r & s \end{smallmatrix}\right) \in \mathrm{GL}_2(\mathbb{Z})$ such that

$$Q(X,Y) = Q'(T(X,Y)) = Q'(pX + qY, rX + sY),$$

and denote this by $Q \sim Q'$. It is easy to see that $\sim$ defines an equivalence relation. We say that the equivalence is *proper* if $\det(T) = +1$, and *improper* if $\det(T) = -1$. Proper equivalence is again an equivalence relation, whereas improper equivalence is not. One can therefore speak of the *proper equivalence class* of a quadratic form. That is, its orbit under the action of $\mathrm{SL}_2(\mathbb{Z})$.

Let B' be the Gram matrix of Q'. We then see that $Q \sim Q'$ if and only if

$$B = {}^tTB'T$$

for some $T \in \mathrm{GL}_2(\mathbb{Z})$. From this it is straightforward to check that $\Delta(Q) = \Delta(Q')$, that Q and Q' represent the same integers, and that Q is primitive if and only if Q' is primitive. Furthermore, note that when $a \neq 0$, we can write

$$4aQ(X,Y) = (2aX + bY)^2 - \Delta(Q)Y^2,$$

from which we can distinguish three cases:

(1) $\Delta(Q) < 0$ and $a > 0$. Then Q only represents positive integers non-trivially and we say that Q is *positive definite*.

(2) $\Delta(Q) < 0$ and $a < 0$. Then Q only represents negative integers non-trivially and we say that Q is *negative definite*.

(3) $\Delta(Q) > 0$. Then Q represents both positive and negative integers, in which case we say that Q is *indefinite*.

These properties are again preserved under equivalence. Moreover, we note that if Q is positive (resp. negative) definite, then $-Q$ is negative (resp. positive) definite; hence the study of positive definite quadratic forms is equivalent to the study of negative definite quadratic forms.

In the continuation of this survey, we will mainly focus on primitive positive definite quadratic forms (PPD forms). Since (properly) equivalent quadratic forms represent the same integers, it would be useful if we were able to write down a specific representative of an equivalence class of forms. It turns out that there are only finitely many equivalence classes of quadratic forms of a given discriminant, and we will now work towards a method to list all PPD forms of a given discriminant. We mainly follow [Cox89].

Definition 1.5. A PPD form $Q(X,Y) = aX^2 + bXY + cY^2$ is callled *reduced* if

$$|b| \leq a \leq c \quad \text{and} \quad b \geq 0 \text{ if either } |b| = a \text{ or } a = c.$$

(Note that a, $c > 0$ since Q only represents positive integers.)

Theorem 1.6. *Every PPD form is properly equivalent to a unique reduced form*

Proof. See [Cox89, Theorem 2.8.]. □

Example 1.7. The forms $3X^2+2XY+5Y^2$ and $3X^2-2XY+5Y^2$ are clearly equivalent through the transformation $\left(\begin{smallmatrix}-1 & 0\\ 0 & 1\end{smallmatrix}\right)$. However, Theorem 1.6 implies that they are not properly equivalent.

Suppose that $Q(X)=aX^2+bXY+cY^2$ is a reduced form of discriminant $\Delta<0$. Then $b^2\leq a^2$ and $a\leq c$, and so

$$-\Delta=4ac-b^2\geq 4a^2-a^2=3a^2;$$

hence $b^2\leq a^2\leq -\Delta/3$, which implies that for a fixed Δ, there are only finitely many choices for a and b. Moreover, since $b^2-4ac=\Delta$, the same holds for c, and so there are only a finite number of reduced forms of discriminant Δ. We say that two PPD forms are in the same *class* if they are properly equivalent. We will denote $C(\Delta)$ for the set of classes of determinant Δ and refer to it as the *form class group* (of PPD forms of discriminant Δ). The order of $C(\Delta)$ is denoted as $h(\Delta)$ and is called the *class number* (of PPD forms of discriminant Δ). Theorem 1.6 now implies the following result.

Theorem 1.8. *Let $\Delta<0$ be fixed. Then the number $h(\Delta)$ of proper equivalence classes of PPD forms of discriminant Δ is equal to the number of reduced forms of discriminant Δ. In particular, $h(\Delta)$ is finite.*

Note that the above discussion gives rise to an algorithm for computing reduced forms and class numbers, albeit not a very efficient one.

Example 1.9. Given a negative integer $\Delta\equiv 0,\ 1 \mod 4$, define the form

$$Q_0(X,Y):=\begin{cases}X^2-\frac{\Delta}{4}Y^2 & \text{if } \Delta\equiv 0 \mod 4,\\ X^2+XY+\frac{1-\Delta}{4}Y^2 & \text{if } \Delta\equiv 1 \mod 4.\end{cases}$$

It is easy to check that Q_0 is reduced, and has discriminant Δ. The form Q_0 is called the *principal form* of discriminant Δ.

Definition 1.10. [Cas78, p.332] A PPD form is called *ambiguous* if it is improperly equivalent to itself. Similarly, a class of forms is called ambiguous if the forms in it are ambiguous.

Example 1.11. Two special examples of ambiguous forms are forms of the shape

$$aX^2+cY^2,\quad aX^2+aXY+cY^2$$

which are sometimes referred to as *special ambiguous* forms. It can be shown that every ambiguous class in $C(\Delta)$ where $\Delta<0$ is a fundamental discriminant, contains exactly 2 special ambiguous forms [She13, Proposition 3.3.19.]. Furthermore, it can be shown that for a fundamental discriminant $\Delta<0$ the number of ambiguous classes in $C(\Delta)$ is exactly 2^{r-1} where r is the number of prime divisors of the fundamental discriminant Δ [She13, Theorem 3.3.20.]. These ambiguous classes are closely connected to elements of order ≤ 2 in $C(\Delta)$ (see Proposition 1.19).

It is possible to impose a group structure on $C(\Delta)$, called *Dirichlet composition* (or simply *composition*), making it into a finite group – as the name form class group already suggested. To define this group structure, we need the following result.

Lemma 1.12. *Assume that $Q(X,Y)=aX^2+bXY+cY^2$ and $Q'(X,Y)=a'X^2+b'XY+c'Y^2$ both have discriminant Δ and satisfy $\gcd(a,a',(b+b')/2)=1$ (since the forms have the same discriminant, $b\equiv b' \mod 2$ and $(b+b')/2$ is an integer). Then there exists a unique integer B modulo $2aa'$ such that*

$$\begin{aligned}B&\equiv b \mod 2a,\\ B&\equiv b' \mod 2a',\\ B^2&\equiv D \mod 4aa'.\end{aligned}$$

Proof. See [Cox89, Lemma 3.2.] □

Definition 1.13. Let $Q(X,Y) = aX^2 + bXY + cY^2$ and $Q'(X,Y) = a'X^2 + b'XY + c'Y^2$ both be PPD forms of discriminant $\Delta < 0$ which satisfy $\gcd(a, a', (b+b')/2) = 1$. Then the *Dirichlet composition* of $Q(X,Y)$ and $Q'(X,Y)$ is defined as the form

$$S(X,Y) = aa'X^2 + BXY + \frac{B^2 - \Delta}{4aa'}Y^2,$$

where B is determined as in Lemma 1.12.

It is relatively straightforward to see that S is primitive, positive definite, and has discriminant Δ. Composing a PPD form $Q(X,Y) = aX^2 + bXY + cY^2$ of negative discriminant $\Delta \equiv 0, 1 \mod 4$ with the principal form $Q_0(X,Y)$ of discriminant Δ from Example 1.9, we see that $B = b$ satisfies the conditions of Lemma 1.12 and that the composition of Q and Q_0 is Q. Similarly, note that the form $Q_-(X,Y) := aX^2 - bXY + cY^2$ is properly equivalent to $Q_-(-Y,X) = cX^2 + bXY + Y^2$. Composing the latter with Q (again taking $B = b$) yields the form $S(X,Y) = acX^2 + bXY + Y^2$, which is properly equivalent to

$$\begin{cases} S(-Y, X + \frac{b}{2}Y) = X^2 - \frac{\Delta}{4}Y^2 & \text{if } \Delta \equiv 0 \mod 4, \\ S(-Y, X + \frac{b+1}{2}Y) = X^2 + XY + \frac{1-\Delta}{4}Y^2 & \text{if } \Delta \equiv 1 \mod 4. \end{cases}$$

This gives some intuition as to why Dirichlet composition imposes a group structure on the set $C(\Delta)$ of proper equivalence classes of PPD forms of discriminant Δ. We omit the other details, but refer the interested reader to [Cox89, Theorem 3.9.].

Theorem 1.14. *Let $\Delta \equiv 0, 1 \mod 4$ be negative and let $C(\Delta)$ be the form class group of PPD forms of discriminant Δ. Then Dirichlet composition gives $C(\Delta)$ the structure of a finite abelian group whose order is the class number $h(\Delta)$.*

Furthermore, the identity element of $C(\Delta)$ is the class containing the principal form

$$Q_0(X,Y) := \begin{cases} X^2 - \frac{\Delta}{4}Y^2 & \text{if } \Delta \equiv 0 \mod 4, \\ X^2 + XY + \frac{1-\Delta}{4}Y^2 & \text{if } \Delta \equiv 1 \mod 4, \end{cases}$$

and the inverse of the class containing the form $Q(X,Y) = aX^2 + bXY + cY^2$ is the class containing the form $Q_-(X,Y) = aX^2 - bXY + cY^2$.

The class containing the principal form Q_0 is called the *principal class* and the form Q_- is called the *opposite form* of Q.

1.3 The Class Group Correspondence

It turns out that for a real quadratic field $K = \mathbb{Q}(\sqrt{d})$ with discriminant $\Delta_K > 0$ there is a bijection between the narrow class group Cl_K^+ and the form class group of primitive quadratic forms of discriminant Δ_K (see e.g. [Coh93, Theorem 5.2.9]). See [BW89, BBT94, BMM00] for some examples of cryptographic constructions based on real quadratic number fields.

For the cryptographic constructions discussed in this survey, we will however focus on imaginary quadratic number fields $K = \mathbb{Q}(\sqrt{d})$ with $\Delta_K < 0$ and their orders $\mathcal{O} \subset K$. In this case there is a bijection between the class group $\mathrm{Cl}_\mathcal{O}$ and the form class group $C(\Delta_\mathcal{O})$ of PPD forms of discriminant $\Delta_\mathcal{O}$. The choice for imaginary quadratic class groups is partly motivated by the following theorem.

Theorem 1.15 (Stark-Heegner [Sta67])**.** *Let $K = \mathbb{Q}(\sqrt{d})$ be an imaginary quadratic number field (i.e. $d < 0$). Then the class number h_K is equal to 1 if and only if*

$$d = -1,\ -2,\ -3,\ -7,\ -11,\ -19,\ -43,\ -67,\ -163.$$

This implies that as long as we sample a fundamental discriminant $\Delta < -163$, the class group $\mathrm{Cl}(\Delta)$ is never going to be trivial. In the case where $K = \mathbb{Q}(\sqrt{d})$ is a real quadratic number field (i.e. $d > 0$), it is unknown whether there are infinitely many fields $\mathbb{Q}(\sqrt{d})$ with class number 1.

We will now proceed to discuss the bijection between these two notions of class groups in the case where the discriminant $\Delta_\mathcal{O}$ of $\mathcal{O} \subset K = \mathbb{Q}(\sqrt{d})$ is negative. This bijection turns out to be an isomorphism with respect to the group structure imposed by Dirichlet composition on the form class group, as introduced in Theorem 1.14.

Theorem 1.16. *Let $\mathcal{O}$ be the order of discriminant $\Delta < 0$ in an imaginary quadratic field K.*

(1) *If $Q(X,Y) = aX^2 + bXY + cY^2$ is a PPD form of discriminant Δ, then*

$$\mathfrak{a}_Q := a\mathbb{Z} + \frac{1}{2}\left(\sqrt{\Delta} - b\right)\mathbb{Z} \subset \mathcal{O}$$

is an invertible $\mathcal{O}$-ideal.

(2) *The map $Q \mapsto \mathfrak{a}_Q$ induces an isomorphism between the form class group $C(\Delta)$ and the ideal class group $\mathrm{Cl}_{\mathcal{O}}$. In particular the order of $\mathrm{Cl}_{\mathcal{O}}$ is equal to the class number $h(\Delta)$.*

(3) *A positive integer m is represented by a form $Q(X,Y)$ if and only if there exists an invertible ideal $\mathfrak{a} \subset \mathcal{O}$ with $\mathrm{Nm}(\mathfrak{a}) = m$, and $[\mathfrak{a}] = [\mathfrak{a}_Q]$ in $\mathrm{Cl}_{\mathcal{O}}$ (recall that $\mathrm{Nm}(\mathfrak{a}) = |\mathcal{O}/\mathfrak{a}|$)*

Proof. See [Cox89, Theorem 7.7] □

Remark (1). The map $Q \mapsto \mathfrak{a}_Q$ from Theorem 1.16 has a natural inverse which is defined as follows. If $\mathfrak{a} = \mathbb{Z}\alpha + \mathbb{Z}\beta$ is an invertible $\mathcal{O}$-ideal with $\mathrm{Im}(\beta/\alpha) > 0$ (without loss of generality, since we can always switch α and β), then

$$Q(X,Y) := \frac{N(\alpha X + \beta Y)}{N(\mathfrak{a})}$$

is a PPD form of discriminant Δ. This map induces the inverse to the map of Theorem 1.16 on the level of classes.

Remark (2). Combining Theorem 1.16 with Theorem 1.6, we can describe classes of ideals in terms of tuples of integers (a,b). By Theorem 1.16 the class group $\mathrm{Cl}_{\mathcal{O}}$ of an order $\mathcal{O}$ of discriminant Δ in an imaginary quadratic field K consists of the classes of ideals of the form

$$a\mathbb{Z} + \frac{1}{2}\left(\sqrt{\Delta} - b\right)\mathbb{Z}, \quad a, b \in \mathbb{Z}.$$

Let $c := (b^2 - \Delta)/(4a)$. Then by Theorem 1.6 and the fact that $aX^2 + bXY + cY^2$ needs to be a PPD form, a, b and c must satisfy:

(1) $4a \mid b^2 - \Delta$,

(2) $\gcd(a,b,c) = 1$,

(3) $|b| \leq a \leq c \quad$ and $\quad b \geq 0$ if either $|b| = a$ or $a = c$.

Similar to the proof of Theorem 1.8, these conditions can be turned into an algorithm for computing representatives of the classes of $\mathrm{Cl}_{\mathcal{O}}$ (note that by Theorem 1.6 these classes are completely determined by the tuples (a,b)).

Remark. From now on we will use the notation $C(\Delta)$ and $\mathrm{Cl}(\Delta)$ interchangeably to denote the class group of discriminant Δ. We will mainly use the notation $C(\Delta)$ when working in the form class group of PPD forms of discriminant Δ and the notation $\mathrm{Cl}(\Delta)$ when working in the ideal class group $\mathrm{Cl}_{\mathcal{O}}$ of the order $\mathcal{O}$ of discriminant $\Delta_{\mathcal{O}} = \Delta$

Example 1.17. Let $K = \mathbb{Q}(\sqrt{-3})$. Then $\mathcal{O}_K = \mathbb{Z}[\frac{1+\sqrt{-3}}{2}]$ of discriminant $\Delta_K = -3$, and the order $\mathcal{O} = \mathbb{Z}[\sqrt{-3}]$ has conductor 2 and discriminant $\Delta_{\mathcal{O}} = -12$. Using the proof of Theorem 1.8, we see that $C(-3)$ only contains the class of the principal form $X^2 + XY + Y^2$ and $C(-12)$ only contains the class of the principal form $X^2 + 3Y^2$. The correspondence of Theorem 1.16 gives

$$X^2 + XY + Y^2 \mapsto \mathbb{Z} + \frac{1}{2}(\sqrt{-3} - 1)\mathbb{Z} = \mathcal{O}_K$$
$$X^2 + 3Y^2 \mapsto \mathbb{Z} + \sqrt{-3}\mathbb{Z} = \mathcal{O};$$

hence $\mathrm{Cl}_{\mathcal{O}_K}$ and $\mathrm{Cl}_{\mathcal{O}}$ are both trivial. Note that this does not imply that $\mathcal{O}$ is a principal ideal domain, since it is not a Dedekind domain. In fact $\mathcal{O}$ is not a principal ideal domain since it is not a unique factorization domain, because

$$(1 + \sqrt{-3}) \cdot (1 - \sqrt{-3}) = 4 = 2 \cdot 2.$$

Example 1.18. Let $K = \mathbb{Q}(\sqrt{-5})$. Then $\mathcal{O}_K = \mathbb{Z}[\sqrt{-5}]$ of discriminant $\Delta_K = -20$, and K has no other orders. Again using the proof of Theorem 1.8, we see that $C(-20)$ consists of the classes of the reduced forms

$$X^2 + 5Y^2, \quad 2X^2 + 2XY + 3Y^2.$$

The corresponding ideals under the isomorphism from Theorem 1.16 are respectively

$$\mathbb{Z} + \sqrt{-5}\mathbb{Z} = \mathcal{O}_K, \qquad 2\mathbb{Z} + (\sqrt{-5} - 1)\mathbb{Z} = (2, 1 + \sqrt{-5});$$

hence the class group $\mathrm{Cl}_{\mathcal{O}_K}$ has order 2 and is generated by the class of $\mathfrak{p} = (2, 1 + \sqrt{-5})$ (i.e. the unique prime lying over 2).

From the correspondence from Theorem 1.16, it is possible to show the following result, which will be useful later on when for cryptographic applications we want to pick the discriminant in such a way that there are no easy to find low order torsion elements in the class group.

Proposition 1.19. *Let $\Delta < 0$ be a fundamental discriminant (see Definition 1.1), and let r be the number of distinct primes dividing Δ. Then the number of elements of order ≤ 2 in $\mathrm{Cl}(\Delta)$ is exactly 2^{r-1}.*

Proof. This is a special case of [Cox89, Proposition 3.11.]. Alternatively, one can easily show that a class in $C(\Delta)$ is ambiguous if and only if it has order ≤ 2 in $C(\Delta)$, and subsequently apply the results discussed in Example 1.11. □

1.4 Security Games and Adversaries

Definition 1.20. A security game G is defined with respect to a set of parameters par (defining the group family) and an adversary $\mathcal{A}$ that plays the game. A game consists of a main procedure that receives as input a security parameter $\kappa \in \mathbb{Z}_{\geq 1}$ and at the end outputs a single bit 0 ($\mathcal{A}$ loses) or 1 ($\mathcal{A}$ wins). We denote the output of a game G executed with parameters par and adversary $\mathcal{A}$ as $\mathsf{G}^{\mathcal{A}}_{\mathrm{par}}(\kappa)$. We define the *advantage* of $\mathcal{A}$ in G as

$$\mathsf{Adv}^{\mathsf{G}}_{\mathrm{par},\mathcal{A}}(\kappa) := \Pr[\mathsf{G}^{\mathcal{A}}_{\mathrm{par}}(\kappa) = 1].$$

We denote the *(expected) running time* of $\mathsf{G}^{\mathcal{A}}_{\mathrm{par}}(\kappa)$ by $\mathsf{Time}^{\mathsf{G}}_{\mathrm{par},\mathcal{A}}(\kappa)$. We extend this notation to be able to denote the advantage conditional on an event E in G:

$$\mathsf{Adv}^{\mathsf{G}}_{\mathrm{par},\mathcal{A}}|_E(\kappa) := \Pr[\mathsf{G}^{\mathcal{A}}_{\mathrm{par}}(\kappa) = 1 \mid E].$$

Definition 1.21. Let G, H be security games. We write $\mathsf{H} \xrightarrow{(\Delta_\varepsilon, \Delta_t)} \mathsf{G}$ if there exists an algorithm $\mathcal{R}$ (called a $(\Delta_\varepsilon, \Delta_t)$-reduction) such that for all algorithms $\mathcal{A}$ playing game G, the algorithm $\mathcal{B} := \mathcal{R}^{\mathcal{A}}$ playing game H satisfies

$$\mathsf{Adv}^{\mathsf{H}}_{\mathrm{par},\mathcal{B}}(\kappa) \geq \Delta_\varepsilon \cdot \mathsf{Adv}^{\mathsf{G}}_{\mathrm{par},\mathcal{A}}(\kappa) - \mathsf{negl}(\kappa), \quad \mathsf{Time}^{\mathsf{H}}_{\mathrm{par},\mathcal{B}}(\kappa) \leq \Delta_t \cdot \mathsf{Time}^{\mathsf{G}}_{\mathrm{par},\mathcal{A}}(\kappa) + T(\kappa),$$

where $T(\kappa)$ is an insignificant overhead, i.e., $\lim_{\kappa\to\infty} {}^{T(\kappa)}\!/_{\mathsf{Time}^{\mathsf{G}}_{\mathrm{par},\mathcal{A}}(\kappa)} \to 0$.

1.5 Abelian Group Families

The definitions in this section follow Section 4 of [vBS21].

Definition 1.22. An *abelian group family* $\mathcal{G} = (\mathcal{G}_\kappa)_{\kappa=1}^{\infty}$ is a family of probability distributions over finite abelian groups defined with:

1. An efficient sampling algorithm GGen that, on input 1^κ, samples uniformly at random a group $\mathbb{G} \in \mathcal{G}_\kappa$ and outputs a group description of $\mathbb{G}$ and $1_{\mathbb{G}}$.
2. An efficient sampling algorithm $\mathsf{GSample}$ which, given a group description of $\mathbb{G}$, outputs a group element $x \in \mathbb{G}$ sampled uniformly at random.

3. Efficient algorithms GMul and GInv that, respectively, multiplies two group elements, and inverts a group element.

4. A group order upper bound $U(\kappa)$: $\forall\kappa\forall\mathbb{G} \in \mathcal{G}_\kappa : U(\kappa) \geq |\mathbb{G}|$, such that $\log U(\kappa) \in \mathsf{poly}(\kappa)$ and $1/U(\kappa) \in \mathsf{negl}(\kappa)$.

5. A random group generator count $n(\kappa) \in \mathbb{Z}_{>0}$ and $n(\kappa) \in \mathsf{poly}(\kappa)$ such that

$$\Pr[\langle \boldsymbol{g} \rangle \neq \mathbb{G} \mid \mathbb{G} \xleftarrow{\$} \mathcal{G}_\kappa, \boldsymbol{g} \xleftarrow{\$} \mathbb{G}^{n(\kappa)}] \in \mathsf{negl}(\kappa).$$

When the security parameter κ is clear from the context, we will usually omit κ and simply denote U and n instead of $U(\kappa)$ and $n(\kappa)$, respectively.

Remark. Note that the bit size of the representations of the group elements of all $\mathbb{G} \in \mathcal{G}_\kappa$ should be polynomial in the security parameter κ, since otherwise it would not be possible to construct efficient algorithms on $\mathbb{G}$. If we assume that an upper bound $p(\kappa)$ on the bit size of the representations is known, this automatically gives an upper bound $U_\kappa = 2^{p(\kappa)}$ on the order of $\mathbb{G}$, for which $\log(U_\kappa)$ is polynomial in κ.

The following Lemma also provides a candidate group generator count $n(\kappa)$.

Lemma 1.23. *For any abelian group $\mathbb{G}$ and $U \geq |\mathbb{G}|$, let $n := \lceil \log_2 U \rceil$ then there exist $g_1, \ldots, g_n$ such that $\langle g_1, \ldots, g_n \rangle = \mathbb{G}$. Moreover, $2n$ random elements fail to generate the full group with exponentially small probability in n, i.e.,*

$$\Pr[\langle \boldsymbol{g} \rangle \neq \mathbb{G} \mid \boldsymbol{g} \xleftarrow{\$} \mathbb{G}^{2n}] \leq 2^{-n} (\leq 1/U).$$

Proof. The first part of the lemma follows directly from the observation that if $g_{i+1} \notin \langle g_1, \ldots, g_i \rangle$ then we have that $|\langle g_1, \ldots, g_{i+1} \rangle| = k \cdot |\langle g_1, \ldots, g_i \rangle|$ with $k \in \mathbb{Z}_{\geq 2}$. The second part of the lemma follows from two observations. First, that for all $g_1, \ldots, g_i$ that generate a strict subgroup $\mathbb{G}' := \langle g_1, \ldots, g_i \rangle \neq \mathbb{G}$ the probability that a randomly sampled element lies in the subgroup is bounded as $\Pr[x \in \mathbb{G}' \mid x \xleftarrow{\$} \mathbb{G}] \leq 1/2$. Second, for $\boldsymbol{g} \in \mathbb{G}^{2n}$ with $\langle \boldsymbol{g} \rangle \neq \mathbb{G}$, it follows that for at least n indices i it holds that $g_{i+1} \in \langle g_1, \ldots, g_i \rangle \neq \mathbb{G}$. □

We will end this section with some examples on how we can define a group family for class groups of imaginary quadratic number fields.

Example 1.24. Let $l : \mathbb{Z}_{\geq 1} \to \mathbb{Z}_{\geq 1}$ be some positive non-constant polynomial. A first way is to, for fixed κ, uniformly sample a negative integer $\Delta \equiv 0, 1 \mod 4$ of length $l(\kappa)$, and consider the class group $\mathrm{Cl}_\mathcal{O}$ of the order $\mathcal{O} := \mathbb{Z} + \sqrt{\Delta/\Delta_K} \cdot \mathcal{O}_K$ in $K := \mathbb{Q}(\sqrt{\Delta})$. If we write $\Delta = d^2\Delta'$ such that Δ' is squarefree, then $K = \mathbb{Q}(\sqrt{\Delta'})$ has discriminant $\Delta_K = \Delta'$ if $\Delta' \equiv 1 \mod 4$ and $\Delta_K = 4\Delta'$ if $\Delta' \equiv 2, 3 \mod 4$. So letting $f = d$ if $\Delta' \equiv 1 \mod 4$ and $f = d/2$ if $\Delta' \equiv 2, 3 \mod 4$, then $\Delta = f^2\Delta_K$ and $\mathcal{O}$ is the order of index f in $\mathcal{O}_K$. (Note that $d \equiv 0 \mod 2$ in the second case since otherwise $d^2 \equiv 1 \mod 4$ which would imply that $\Delta \equiv 2, 3 \mod 4$.) Alternatively, one could sample a prime $p \equiv 3 \mod 4$ of length $l(\kappa)$ and consider the class group of the maximal order $\mathbb{Z}[\frac{1+\sqrt{-p}}{2}]$ of discriminant $\Delta = -p$, or the class group of the non-maximal order $\mathbb{Z}[\sqrt{-p}]$ of discriminant 4Δ. In the same manner one could sample a prime $p \equiv 1 \mod 4$ of length $l(\kappa)$ and consider the class group of the maximal order $\mathbb{Z}[\sqrt{-p}]$ of discriminant $\Delta = -4p$.

Moreover, we will later see that that the order of $\mathrm{Cl}(\Delta)$ is approximately $\sqrt{|\Delta|}$ (see Section 2.1.3); hence we can define an upper bound $U(\kappa) \propto 2^{l(\kappa)/2}$ (note that the constant might in fact depend (sub)polynomially on κ, but we ignore this for now). By Lemma 1.23 we therefore know that $n(\kappa) \approx l(\kappa)$ random elements generate the class group with all but exponentially small probability. Multiplication and inversion in the class group can be done efficiently through the correspondence with quadratic forms from Section 1.3. See [Coh93, Section 5.4.2] for algorithms for the composition and reduction of quadratic forms. Note that by Theorem 1.14, we can compute the inverse class a form $aX^2 + bXY + cY^2$ by simply reducing the form $aX^2 - bXY + cY^2$. The question remains whether we can efficiently sample elements of the class group sufficiently close to uniformly random. If a system of generators is known, it is straightforward to sample elements close to uniform by raising the generators to random integer exponents, sampled from a sufficiently large interval, and

multiplying them [vBS21, Lemma 4.8]. However, computing the structure of the class group is intractable for large discriminants. As we will see in Section 2.1.2, the fastest known classical algorithm has a heuristic running time of $L_{|\Delta|}[1/2, 1]$. Hence it is intractable to compute a system of generators and sample elements close to uniform in the way described above.

Another possible approach is the following. Under the assumption of GRH, the class group $\mathrm{Cl}(\Delta)$ is generated by the classes of prime ideals of norm at most $12\log(|\Delta|)^2$ (see [Bac90, Theorem 4]). Here one can limit oneself to primes $\mathfrak{p}$ lying over primes p for which $\left(\frac{\Delta}{p}\right) = 1$ (i.e. which split completely) and $p \leq 12\log(|\Delta|)^2$, since the prime ideals $\mathfrak{p}$, $\mathfrak{p}'$ lying over a split prime p have norm equal to p (see [Coh93, Section 5.5]). For discriminants of length $l(\kappa)$ this means we have to consider primes $p \leq 12l(\kappa)^2$, which is polynomial by construction. Similarly this would let us sample elements close to uniform by raising the primes $\mathfrak{p}$ lying over these p to some random exponents and multiplying them together.

In light of recent cryptographic constructions based on the action of the class group of an imaginary quadratic order on an appropriate set of supersingular elliptic curves, such as CSIDH [CLM+18], various heuristics of smaller sets of prime ideals generating the class group were proposed. When p is a prime of the form $p = 4\ell_1 \cdots \ell_n - 1$, where the ℓ_i are small distinct odd primes, and we consider the class group of the order $\mathcal{O} = \mathbb{Z}[\sqrt{-p}]$, Castryck et al. argue that a large cyclic subgroup of the class group is generated by the classes of the primes $\mathfrak{l}_i := (\ell_i, \sqrt{-p} - 1)$ lying over ℓ_i [CLM+18]. Moreover, they argue that, by the Cohen Lenstra heuristics [CL84] and the Gaussian heuristic [NV10, Chapter 2, Definition 8], one can uniformly sample elements in the (large cyclic subgroup of the) class group by sampling $e_1, \ldots, e_n \xleftarrow{\$} \{-m, \ldots, m\}$ and considering $[\mathfrak{l}_1^{e_1} \cdots \mathfrak{l}_n^{e_n}]$, where $2m + 1 \geq \sqrt[n]{\#\,\mathrm{Cl}_{\mathcal{O}}}$. Furthermore, Biasse, Iezzi and Jacobson argue that, for $c > 0$, the class group $\mathrm{Cl}_{\mathcal{O}}$ of an imaginary quadratic order $\mathcal{O}$ of discriminant Δ is generated by the classes of $k = \log^{2/3}(|\Delta|)$ split prime ideals $(\mathfrak{p}_i)_{i \leq k}$ of norm at most $\log^c(|\Delta|)$, and that each class of $\mathrm{Cl}_{\mathcal{O}}$ has a representative of the form $\prod_i \mathfrak{p}_i^{e_i}$ with $e_i \leq e^{\log^{1/3}(|\Delta|)}$.

We will later discuss appropriate choices of defining a group family which are believed to make the relevant computational problems hard.

1.6 Computational Problems

We will define the relevant computational problems with respect to some abstract finite abelian group family $\mathcal{G} = (\mathcal{G}_\kappa)_{\kappa \in \mathbb{Z}_{\geq 1}}$ as in Definition 1.22, which in practice will for instance be one of the group families from Example 1.24. We separate the definitions for cyclic groups and finite abelian (not necessarily cyclic) groups since the former provide a more natural framework for discrete logarithm and Diffie-Hellman problems as we know them.

Name	Game $((\mathbb{G}, g) \xleftarrow{\$} \mathcal{G}_\kappa)$	Outcome
$\mathsf{MO}_{\mathcal{C}}$	$N \xleftarrow{\$} \mathcal{A}(g)$	$N \equiv 0 \pmod{\lvert\mathbb{G}\rvert}$
$\mathsf{HO}_{\mathcal{C}}$	$N \xleftarrow{\$} \mathcal{A}(g)$	$N = \lvert\mathbb{G}\rvert$
$\mathsf{DLog}/\mathsf{DLog}_1$	$X \xleftarrow{\$} \mathbb{G},\ e \xleftarrow{\$} \mathcal{A}(g, X)$	$g^e = X$
DLog_2	$X \xleftarrow{\$} \mathbb{G},\ Y \xleftarrow{\$} \langle X \rangle,\ e \xleftarrow{\$} \mathcal{A}(g, X, Y)$	$X^e = Y$
$\mathsf{CDH}/\mathsf{CDH}_1$	$a, b \xleftarrow{\$} \mathcal{U}_{\lvert\mathbb{G}\rvert},\ Y \xleftarrow{\$} \mathcal{A}(g, g^a, g^b)$	$Y = g^{ab}$
CDH_2	$X \xleftarrow{\$} \mathbb{G},\ a, b \xleftarrow{\$} \mathcal{U}_{\lvert\langle X\rangle\rvert},\ Y \xleftarrow{\$} \mathcal{A}(g, X, X^a, X^b)$	$Y = X^{ab}$

Here $\mathcal{G} = (\mathcal{G}_\kappa)_{\kappa=1}^{\infty}$ is a cyclic group family with security parameter κ, and $\mathcal{A}$ is an adversary playing the game. Each game starts by sampling $(\mathbb{G}, g)$.

Table 1: Overview of the relevant computational games in cyclic groups

It is not too hard to show that if one can solve the DLog problem in cyclic groups, one can obtain a multiple of the order of the group order, and thus solve the $\mathsf{MO}_{\mathcal{C}}$ problem, with high probability. We have in fact shown that it is possible to obtain the *exact* order from multiple independent multiples of the order, one can obtain the *exact* order with high probability. Furthermore, we have shown that $\mathsf{MO} \Rightarrow \mathsf{DLog}_1$ for finite abelian (not necessarily cyclic) groups. The details of these reductions are out of scope for this review, but we refer the interested reader to [vBS21]. Since solving DLog in groups of *known* order such as multiplicative groups of finite fields or elliptic curve groups still appears to be hard, it is believed that there does not exist an efficient reduction in the

Name	Game ($\mathbb{G} \xleftarrow{\$} \mathcal{G}_\kappa,\ \boldsymbol{g} := (g_1, \ldots, g_n) \xleftarrow{\$} \mathbb{G}^n$)	$\mathcal{A}$ wins if
MO	$N \xleftarrow{\$} \mathcal{A}(\boldsymbol{g})$	$N \equiv 0 \pmod{\lvert\mathbb{G}\rvert} \wedge N \neq 0$
HO	$N \xleftarrow{\$} \mathcal{A}(\boldsymbol{g})$	$N = \lvert\mathbb{G}\rvert$
LO	$(X, d) \xleftarrow{\$} \mathcal{A}(\boldsymbol{g})$	$X \neq 1_\mathbb{G} \wedge 1 < d \ll 2^\kappa \wedge X^d = 1_\mathbb{G}$
EO	$X \xleftarrow{\$} \mathbb{G},\ d \xleftarrow{\$} \mathcal{A}(\boldsymbol{g}, X)$	$X^d = 1_\mathbb{G}$
DLog_1	$X \xleftarrow{\$} \mathbb{G},\ \boldsymbol{e} \xleftarrow{\$} \mathcal{A}(\boldsymbol{g}, X)$	$\boldsymbol{g}^{\boldsymbol{e}} = X$
DLog_2	$X \xleftarrow{\$} \mathbb{G},\ Y \xleftarrow{\$} \langle X \rangle,\ a \xleftarrow{\$} \mathcal{A}(\boldsymbol{g}, X, Y)$	$X^a = Y$
CDH_2	$X \xleftarrow{\$} \mathbb{G},\ \ a, b \xleftarrow{\$} \mathcal{U}_{\lvert\langle X\rangle\rvert},\ Y \xleftarrow{\$} \mathcal{A}(\boldsymbol{g}, X, X^a, X^b)$	$Y = X^{ab}$
e-RT	$X \xleftarrow{\$} \mathbb{G},\ Y \xleftarrow{\$} \mathcal{A}(\boldsymbol{g}, X^e)$	$Y^e = X \ \wedge\ e > 1$
StRoot	$X \xleftarrow{\$} \mathbb{G},\ (Y, e) \xleftarrow{\$} \mathcal{A}(\boldsymbol{g}, X)$	$Y^e = X \ \wedge\ e > 1$
ARoot	$X \xleftarrow{\$} \mathcal{A}(\boldsymbol{g}),\ \ell \xleftarrow{\$} \mathrm{Primes}(2\kappa),\ Y \xleftarrow{\$} \mathcal{A}(X, \ell)$	$X \neq 1_\mathbb{G} \ \wedge\ Y^\ell = X$

Here $\mathcal{G} = (\mathcal{G}_\kappa)_{\kappa=1}^\infty$ is a group family with security parameter κ, and $\mathcal{A}$ is an adversary playing the game. Each game starts by sampling $\mathbb{G}$, $g_1, \ldots, g_n$.

Table 2: Overview of the relevant computational games in finite abelian groups

opposite direction. In particular there does not exist an efficient *generic* reduction by the lower bounds on the complexity of solving DLog in the *generic group model* established by Shoup [Sho97b, Theorem 1]. Similar considerations hold for the CDH problem by [Sho97b, Theorem 3].

Given a multiple N of the group order $\mathbb{G}$ and a positive integer e with $\gcd(e, |\mathbb{G}|) = 1$, it is possible to compute e-th roots of any element $X \in \mathbb{G}$. One can define $N' := N/\gcd(e^{\lfloor \log_e(N) \rfloor}, N)$. The resulting value N' will still be a multiple of $|\mathbb{G}|$, and will be coprime to e. An e-th root of X is now given by $Y := X^d$ where $ed \equiv 1 \mod N'$. From this the reductions StRoot $\Rightarrow$ MO, ARoot $\Rightarrow$ MO and e-RT $\Rightarrow$ MO for $\gcd(e, |\mathbb{G}|) = 1$ follow.

For the case where $\gcd(e, |\mathbb{G}|) > 1$, the situation is less straightforward. For cyclic groups and some more general forms of finite abelian groups, Shank's algorithm can be extended to compute e-th roots [Lin97, Chapter 3] when the *exact* group order is known. More specifically, this covers the cases where $\mathbb{G}$ can be written as a product $\mathbb{G} = S \times H$ where S is a cyclic group of order e^n and H is a finite abelian group of order coprime to e [Lin97, Theorem 11.]. In these particular cases one can easily decide whether an e-th root of an element $X \in \mathbb{G}$ exists by checking whether $X^{|\mathbb{G}|/e} \stackrel{?}{=} 1_\mathbb{G}$ [Lin97, Lemma 10.].

Note that this condition is not sufficient for more general forms of finite abelian groups. For example, when $\mathbb{G} = (\mathbb{Z}/2\mathbb{Z}) \times (\mathbb{Z}/2\mathbb{Z})$, we see that $X^2 = 1$ for all $X \in \mathbb{G}$ while the only square is the identity. The group $\mathbb{G} = \mathbb{Z}/4\mathbb{Z}$ has the same order and does permit this condition; hence solely knowing the group order is not sufficient to determine which elements are squares. Similarly, in $\mathbb{G} = (\mathbb{Z}/4\mathbb{Z}) \times (\mathbb{Z}/2\mathbb{Z})$ the elements $(2, 0)$ and $(2, 1)$ both have order 2, but only $(2, 0)$ is a square; hence additionally knowing the order of each element does not let us decide whether an element is a square. This leads us to believe that in these cases more information is needed about the structure of the group, such as the decomposition of $\mathbb{G}$ in terms of cyclic groups of prime power. The problem of determining such a decomposition is sometimes referred to as the *structure problem* SP. The order problem HO and the discrete logarithm problem DLog can both be seen as special cases of the structure problem SP [BV07, Section 9.7]. Determining whether an element $X \in \mathbb{G}$ is an e-th power now comes down to determining whether all the cyclic components of X are e-th powers in their respective cyclic subgroups. However to do so one would need to be able to compute the isomorphism of a group $\mathbb{G}$ to its decomposition in terms of cyclic groups. If one were able to do so efficiently, essentially all computational problems in the group $\mathbb{G}$ would be trivial.

It remains an open question whether it is possible to compute e-th roots (where $\gcd(e, |\mathbb{G}|) > 1$) given only a multiple of the order or even without knowing the order. Furthermore, it is even an open question whether there exist efficient algorithms to compute e-th roots (where $\gcd(e, |\mathbb{G}|) > 1$) in arbitrary finite abelian groups of *known* order.

In RSA groups it is a well known result that computing square roots is equivalent to finding a multiple of the group order, which in turn is equivalent to factoring (see e.g. [KL14, Theorem 13.36]). However, in the class group $\mathrm{Cl}(\Delta)$ of an imaginary quadratic number field of discriminant $\Delta < 0$, it is possible to efficiently compute square roots (resp. decide if they exist) if the factorization of the discriminant Δ is known [Lag80, Theorem 6.10 (resp. Corollary 6.4)]. The factorization of

the discriminant does in general not reveal anything about the group order $h(\Delta)$ except for its parity [HMT98]. On the other hand, if the factorization of Δ is kept secret, computing square roots in $\mathrm{Cl}(\Delta)$ is standard model equivalent to factoring Δ [HMT98, Theorem 12]. It is an open question whether e-RT is equivalent to HO or MO for any value of $e > 1$ in general finite abelian groups. In the abelian hidden order algebraic group model it has been shown that e-RT is equivalent to MO if $\gcd(e, |\mathbb{G}|) = 1$ [vBS21, Theorem 7.4]

Note that similar considerations as the above hold for computing roots of an element $X \in \mathbb{G}$ if (a multiple of) its order $|\langle X \rangle|$ is known, in contrast to (a multiple of) the order of the group being known in the above discussion. Therefore, in order for e-RT, StRoot and ARoot to be difficult, it must be difficult to compute the order of random elements $X \xleftarrow{\$} \mathbb{G}$. We call this the element order problem EO, and it is easy to see that we have reductions $\{e\text{-RT}, \mathsf{StRoot}, \mathsf{ARoot}\} \Rightarrow \mathsf{EO}$ (for $\gcd(e, |\mathbb{G}|) = 1$). In the other direction we have the reductions $\mathsf{EO} \Rightarrow \{\mathsf{MO}, \mathsf{HO}, \mathsf{DLog}_1, \mathsf{DLog}_2\}$. These are all trivial except for $\mathsf{EO} \Rightarrow \mathsf{DLog}_1$ which goes via the reduction $\mathsf{MO} \Rightarrow \mathsf{DLog}_1$ [vBS21, Theorem 6.5], and $\mathsf{EO} \Rightarrow \mathsf{DLog}_2$ which can be shown almost identical to the reduction $\mathsf{MO}_{\mathcal{C}} \Rightarrow \mathsf{DLog}$ [vBS21, Lemma 3.1]. Moreover, using similar methods to [vBS21, Section 7] one can show a reduction $\mathsf{MO} \Rightarrow \mathsf{EO}$.

A necessary condition for the EO problem to be hard is that the order of a random element must be large with high probability. It follows from the following lemma that this is true if $|\mathbb{G}|$ has a sufficiently large (e.g., superpolynomial) prime divisor.

Lemma 1.25. *Let $|\mathbb{G}| = \prod_p p^{e(p)}$ be the prime factorization of $|\mathbb{G}|$. Then the probability for a prime $p \mid |\mathbb{G}|$ to divide the order of a uniformly random element of $\mathbb{G}$ is $1 - 1/p^{e(p)}$.*

Proof. By the fundamental theorem of finite abelian groups we can write $\mathbb{G} \cong \bigoplus_{i=1}^{t} (\mathbb{Z}/p_i^{e_i}\mathbb{Z})$, where $p_1, \dots p_t$ are (not necessarily distinct) prime numbers. The order of an element $X \in \mathbb{G}$ is *not* divisible by a prime $p \mid |\mathbb{G}|$ if and only if X has trivial components in all subgroups corresponding to $(\mathbb{Z}/p_i^{e_i}\mathbb{Z})$ with $p_i = p$. There are exactly $\prod_{p_i \neq p} p_i^{e_i} = |\mathbb{G}|/p^{e(p)}$ such elements. □

1.7 Computing Class Group Torsion

Given an abelian group $\mathbb{G}$, we denote $\mathbb{G}[\ell]$ for the *subgroup of ℓ-torsion elements*. That is, the $g \in \mathbb{G}$ such that $g^\ell = 1$ (where we write the group operation multiplicatively). Further recall that for p prime, the p-rank of $\mathbb{G}$, denoted $r_p(\mathbb{G})$, is the largest rank of an elementary abelian p-subgroup of $\mathbb{G}$. That is, $r_p(\mathbb{G})$ is equal the largest non-negative integer r such that $(\mathbb{Z}/p\mathbb{Z})^r$ is a subgroup of $\mathbb{G}$. Note that the number of elements of order p in $\mathbb{G}$ is completely characterized by the p-rank of $\mathbb{G}$ as

$$\#\mathbb{G}[p] = p^{r_p(\mathbb{G})},$$

and that the number of elements of order *exactly* p is therefore equal to $p^{r_p(\mathbb{G})} - 1$. In this section we discuss bounds on the size of the torsion subgroups of the class group of an imaginary quadratic number field, and explore possible ways to compute low order torsion elements. That is, to explore the difficulty of the low order problem LO in IQ class groups.

The *exponent* of a finite abelian group $\mathbb{G}$ is defined to be the smallest positive integer m such that $g^m = 1$ for every $g \in \mathbb{G}$, i.e. such that $\mathbb{G}[m] = \mathbb{G}$. Writing $m(\Delta)$ for the exponent of $\mathrm{Cl}(\Delta)$, Boyd and Kisilevsky [BK72] showed that, under the assumption of the generalized Riemann hypothesis (GRH), $m(\Delta) \to \infty$ as $|\Delta| \to \infty$. They furthermore showed unconditionally that there are only finitely many $\Delta < 0$ for which $m(\Delta) = 3$, i.e. for which $\mathrm{Cl}(\Delta)$ is an elementary abelian 3-group. Earlier Chowla [Cho34] showed that there are only finitely many $\Delta < 0$ for which $m(\Delta) \leq 2$.

Ellenberg and Venkatesh [EV07] showed that, under the assumption of GRH, the cardinality of $\mathrm{Cl}(\Delta)[\ell]$ is $\ll_\epsilon |\Delta|^{1/2 - \frac{1}{2\ell} + \epsilon}$. They also showed unconditionally that the 3-torsion part of $\mathrm{Cl}(\Delta)$ is $\ll_\epsilon |\Delta|^{1/3+\epsilon}$. Recently, Heath-Brown and Pierce [HBP17] showed unconditionally that for any prime $\ell \geq 5$

$$\sum_{\substack{0 < d < X \\ -d \text{ fundamental}}} |\mathrm{Cl}(-d)[\ell]| \ll_\epsilon X^{\frac{3}{2} - \frac{3}{2\ell+2} + \epsilon}.$$

It was earlier shown by Davenport and Heilbronn [DH71] that for the case $\ell = 3$

$$\sum_{\substack{0<d<X \\ -d \text{ fundamental}}} |\operatorname{Cl}(-d)[3]| \sim 2 \sum_{\substack{0<d<X \\ -d \text{ fundamental}}} 1 \sim \frac{6}{\pi^2} X \quad \text{as } X \to \infty,$$

which implies that the average number of 3-torsion elements goes to two as $X \to \infty$. The average number of elements that have order *exactly* three therefore goes to one. This result was further explored by Bhargava and Varma [BV16]. They showed that if one allows non maximal orders (in imaginary quadratic number fields), this average becomes $1 + \zeta(2)/\zeta(3)$. Note that $\zeta(2)/\zeta(3) \approx 1.36843 > 1$; hence if one effectively wants to limit the number of low-order torsion elements in the class group – as it turns out we want for time-lock cryptographic applications – it makes sense to only consider maximal orders. It was furthermore conjectured by Cohen and Lenstra that for any odd prime p, the average number of p-torsion elements goes to 2 [CL84, (C 6)]. Therefore, on average the class group contains one element of order *exactly* p. Naturally this never occurs for a particular class group since if a group contains one element of order p it immediately contains $p - 1$ elements of order exactly p.

When $\mathfrak{a}$ is an ideal of order ℓ in the class group of $K = \mathbb{Q}(\sqrt{d})$ for $d < 0$, this means that $\mathfrak{a}^\ell$ is principal. That is, there are $a, b \in \mathbb{Z}$ such that

$$\mathfrak{a}^\ell = \begin{cases} (a + b\frac{1+\sqrt{d}}{2})\mathbb{Z}[\frac{1+\sqrt{d}}{2}] & \text{if } d \equiv 1 \mod 4, \\ (a + b\sqrt{d})\mathbb{Z}[\sqrt{d}] & \text{if } d \equiv 2, 3 \mod 4. \end{cases}$$

Taking the ideal norm on both sides and putting $c := N(\mathfrak{a})$, we obtain the equations

$$c^\ell = \begin{cases} a^2 + ab + b^2\left(\frac{1-d}{4}\right) & \text{if } d \equiv 1 \mod 4, \\ a^2 - db^2 & \text{if } d \equiv 2, 3 \mod 4. \end{cases} \tag{1}$$

We can write

$$\mathfrak{a} := \alpha\mathbb{Z} + \frac{1}{2}\left(\sqrt{\Delta_K} - \beta\right)\mathbb{Z} \subset \mathcal{O}_K, \tag{2}$$

for some $\alpha, \beta \in \mathbb{Z}$, that is, coming from a reduced PPD form under the correspondence from Theorem 1.16. Taking this form to be reduced, we can assume that $\beta^2 \leq \alpha^2 \leq -\Delta_K/3$. From equation (2), we see that $c = N(\mathfrak{a}) = \alpha$ (cf. [Coh93, Proposition 5.2.1.]), and so combining this with equations (1) we obtain the bounds

$$\begin{cases} |c| \leq (\frac{-d}{3})^{1/2}, \quad |a| \leq (\frac{-d}{3})^{\ell/4}, \quad |b| \leq (\frac{4}{1-d})^{1/2}(\frac{-d}{3})^{\ell/4} & \text{if } d \equiv 1 \mod 4, \\ |c| \leq (\frac{-4d}{3})^{1/2}, \quad |a| \leq (\frac{-4d}{3})^{\ell/4}, \quad |b| \leq (\frac{1}{-d})^{1/2}(\frac{-4d}{3})^{\ell/4} & \text{if } d \equiv 2, 3 \mod 4. \end{cases}$$

Now assume we can find an integer solution (a, b, c) to equation (1). Then we want to find a corresponding ideal $\mathfrak{a}$ with norm c. One could simply try to put

$$\mathfrak{a} = c\mathbb{Z} + \frac{1}{2}\left(\sqrt{\Delta_k} - \gamma\right)\mathbb{Z}$$

for some $\gamma \in \mathbb{Z}$. This is an $\mathcal{O}_K$ ideal if and only if $4c|(\gamma^2 - \Delta_K)$ (cf. [Coh93, Theorem 5.2.4.]), and has $N(\mathfrak{a}) = c$. Moreover, since we can pick $\mathfrak{a}$ reduced, we can try to find integers $\delta \in \mathbb{Z}_{\geq 0}$, $\gamma \in \mathbb{Z}$ such that

(1) $\gamma^2 = 4c\delta + \Delta_K$, or equivalently $\delta = \frac{\gamma^2 - \Delta_K}{4c}$;

(2) $\gcd(c, \gamma, \delta) = 1$;

(3) $|\gamma| \leq c \leq \delta$ and $\gamma \geq 0$ if either $|\gamma| = c$ or $c = \delta$.

Example 1.26. Let $K = \mathbb{Q}(\sqrt{-p})$ where $p \equiv 3 \mod 4$ is a prime, and $\ell = 3$. Then equation (1) becomes

$$c^3 = a^2 + ab + b^2\left(\frac{1+p}{4}\right), \tag{3}$$

where a, b and c are integers satisfying the bounds

$$|c| \leq \left(\frac{p}{3}\right)^{1/2}, \quad |a| \leq \left(\frac{p}{3}\right)^{3/4}, \quad |b| \leq \left(\frac{4}{1+p}\right)^{1/2} \left(\frac{p}{3}\right)^{3/4}. \tag{4}$$

If one can successfully find a solution to this equation, a corresponding ideal $\mathfrak{a}$ can be recovered by finding integers $\delta \in \mathbb{Z}_{\geq 0}$, $\gamma \in \mathbb{Z}$ such that (1), (2) and (3) are satisfied, and subsequently putting

$$\mathfrak{a} = c\mathbb{Z} + \frac{1}{2}\left(\sqrt{-p} - \gamma\right)\mathbb{Z}.$$

For example taking $p = 23$, equation (3) has nontrivial solutions

$$(a, b, c) \in \{(\pm 1, 0, 1), (\pm 1, \pm 1, 2), (\mp 2, \pm 1, 2)\}$$

within the bounds (4). It is easy to see that the solutions with $c = 1$ corresponds to the trivial ideal $\mathfrak{a} = \mathcal{O}_K$. For $c = 2$, we obtain $(c, \gamma, \delta) \in \{(2, \pm 1, 3)\}$ satisfying (1), (2) and (3). The corresponding ideals are given by

$$\mathfrak{p}_{2\pm} = 2\mathbb{Z} + \frac{1}{2}(\sqrt{-23} \pm 1)\mathbb{Z},$$

i.e., the prime ideals of $\mathcal{O}_K = \mathbb{Z}[\frac{1+\sqrt{-23}}{2}]$ lying over 2, which indeed have order 3 in Cl_K.

Example 1.27. Let $K = \mathbb{Q}(\sqrt{-p})$ where $p \equiv 1 \mod 4$ is a prime, and again $\ell = 3$. Equation (1) now becomes

$$c^3 = a^2 + pb^2,$$

where a, b and c are integers satisfying the bounds

$$|c| \leq \left(\frac{4p}{3}\right)^{1/2}, \quad |a| \leq \left(\frac{4p}{3}\right)^{3/4}, \quad |b| \leq \left(\frac{-1}{p}\right)^{1/2} \left(\frac{4p}{3}\right)^{3/4}. \tag{5}$$

For example, taking $p = 29$, equation (5) has nontrivial solutions

$$(a, b, c) \in \{(\pm 1, 0, 1), (\pm 2, \pm 5, 9), (\pm 3, \pm 2, 5), (\pm 8, 0, 4), (\pm 10, \pm 2, 6)\}.$$

The solutions with $c = 1$ correspond to the trivial ideal $\mathcal{O}_K$, and the solutions with $c \in \{4, 9\}$ lead us to principal ideals; so it remains to consider the remaining solutions. For $c = 5$, we obtain $(c, \gamma.\delta) \in \{(5, \pm 2, 10)\}$ satisfying (1), (2) and (3). The corresponding ideals are given by

$$\mathfrak{p}_{5\pm} = 5\mathbb{Z} + (\sqrt{-29} \pm 1)\mathbb{Z},$$

i.e., the primes of $\mathcal{O}_K[\sqrt{-29}]$ lying over 5, which indeed have order 3 in the class group Cl_K. Checking $c = 6$ for completeness' sake, we obtain $(c, \gamma, \delta) \in \{(6, \pm 2, 5)\}$, which does not correspond to a reduced ideal. Indeed, it corresponds to the ideals $6\mathbb{Z}+(\sqrt{-29}\pm 1)\mathbb{Z}$, which are equivalent to the ideals $5\mathbb{Z}+(\sqrt{-29}\mp 1)\mathbb{Z}$, respectively (through multiplication by the principal ideals $(1/6\mp\sqrt{-29}/6)$, respectively).

Coppersmith's Method

Coppersmith's Method [Cop97] works to find sufficiently small integer roots of a modular polynomial in one variable or to find integer roots of a polynomial in two variables over the integers (provided there exists a small enough solution in both cases). It was already originally suggested to also work in some cases for polynomials in three variables over the integers, and this was explored further in [BJ07]. It is however claimed in [BBF18] that the bounds (4) for equation (1) are out of reach for Coppersmith's method, but that it might be possible to tune the method specifically for this family of surfaces.

LO in Class Groups

A recent work by Belabas, Kleinjung, Sanso and Wesolowski [BKSW20] explores the low order assumption LO (see Table 2) in class groups of imaginary quadratic number fields. It is noted that the class group $\mathrm{Cl}(-\Delta)$, where Δ is a Mersenne prime (i.e. $\Delta = 2^p - 1$ with p prime), satisfies $h(-\Delta) \equiv 0 \mod p - 2$. Moreover, they show that the corresponding element of low order is the class of the ideal $\mathfrak{p}_2 := 2 \cdot \mathbb{Z} + (\sqrt{-\Delta} - 1)/2 \cdot \mathbb{Z}$. That is, the class of the prime lying over 2 has order dividing $p - 2$ in $\mathrm{Cl}(-\Delta)$. The low order problem is thus easy to solve in such a class group, even though it is in general believed to be hard for large primes $\Delta \equiv 3 \mod 4$. Moreover, they show that if $D = 4u^3 - 1$ with $u \in \mathbb{Z}_{>0}$, then the low order assumption does not hold in $\mathrm{Cl}(-D)$ since the class of the ideal $I = u \cdot \mathbb{Z} + (1 + \sqrt{-D})/2 \cdot \mathbb{Z}$ in $\mathbb{Z}[(1 + \sqrt{-D})/2]$ has order 3 in $\mathrm{Cl}(-D)$. Hence one also has to be careful of discriminants of this form. Finally, they quote the following theorem by Mollin [Mol96], which provides us with another class of discriminants to avoid.

Theorem 1.28 ([Mol96, Theorem 3.1.])**.** *Let $D > 0$ be a squarefree integer and let $\sigma = 2$ if $D \equiv 3 \mod 4$, and $\sigma = 1$ otherwise. Furthermore, assume that $D = \sigma^2 m^t - b^2$ where $t > 1$, $m > 1$ and $b > 0$ are integers such that $b \neq 2m^{t/2} - 1$ if t is even and $b \neq \lfloor \sigma m^{t/2} \rfloor$ if t is odd. Then the ideal $m \cdot \mathbb{Z} + (b + \sqrt{-D})/\sigma \cdot \mathbb{Z}$ has order t in the class group $\mathrm{Cl}(-4D/\sigma^2)$ of $\mathbb{Q}(\sqrt{-D})$.*

2 Choosing Cryptographic Parameters

Since the class group of an order in an imaginary quadratic number field depends only on the discriminant, the discriminant is the main cryptographic parameter. In this section we discuss how we can pick the discriminant to make some of the computational problems discussed in Section 1.6 hard. To stress the difference with the problems in arbitrary finite abelian groups, we write a prefix IQ-G to mean that we consider the problem G specifically in class groups of imaginary quadratic orders. We separately discuss the classical setting in Section 2.1 and the post-quantum setting in Section 2.2, and discuss the most important algorithms for the relevant problems in both settings. This will lead to different parameter choices for both settings and will effectively limit the suitable applications for imaginary quadratic class groups in the post-quantum setting.

Throughout this section, the notation $f(x) \sim g(x)$ is used to indicate that two functions f and g are *asymptotically equivalent*, that is $\lim_{x \to \infty} f(x)/g(x) = 1$. The function $L_n[e, c]$ is used to describe the asymptotic running time of subexponential algorithms. Let n, e, $c \in \mathbb{R}$ with $0 \leq e \leq 1$ and $c > 0$. Then

$$L_n[e, c] = \exp\left((c + o(1)) \cdot (\log |n|)^e \cdot (\log \log |n|)^{1-e}\right).$$

2.1 Classical Setting

Hamdy and Möller analyze the most relevant classical algorithms for computing discrete logarithms in class groups [HM00]. We discuss these methods in the following subsections. One thing to note is that the definition of the discrete logarithm problem differs between various sources. Sometimes, a fixed or random element $g \in \mathrm{Cl}(\Delta)$ is chosen first, and the problem is to find the discrete logarithm of an element $X \in \mathrm{Cl}(\Delta)$ with respect to g, or to show that it does not exist [HT99, Vol00]. A different definition of the problem is that of, given a fixed or randomly sampled set of generators $g_1, \ldots, g_n \in \mathrm{Cl}(\Delta)$ and an element $x \in \mathrm{Cl}(\Delta)$, to compute a representation $(e_1, \ldots, e_n) \in \mathbb{Z}^n$ such that $g_1^{e_1} \cdots g_n^{e_n} = X$. Many discrete logarithm and group order computing algorithms actually compute representations as in the latter problem in order to solve the former problem [Coh93, Vol00]. To avoid overly cluttering this section, we will refer to all variants of the DLog problem in imaginary quadratic class groups as IQ-DLog. Since there exist efficient reductions from the multiple order problem MO to both variants of the discrete logarithm problem (see [vBS21, Corollary 6.4, Theorem 6.5]), and since order computation algorithms are in practice very similar to discrete logarithm computation algorithms for abelian groups (see [Coh93, Chapter 5]), we will assume the same asymptotic runtime for solving IQ-MO/HO as we do for solving IQ-DLog.

2.1.1 Reduction to Other Problems

Let $\Delta < 0$ be a fundamental discriminant and let f be a positive integer. Then if $\Delta \neq -3, -4$,

$$h(f^2\Delta) = h(\Delta) f \prod_{p|f} \left(1 - \left(\frac{\Delta}{p}\right)\frac{1}{p}\right),$$

where $\left(\frac{\Delta}{p}\right)$ denotes the Kronecker symbol [Cox89, Theorem 7.24.]. For example, $h(-8) = 1$, since $X^2 + 2Y^2$ is the only reduced form of discriminant -8, and so $h(-8p^2) = p - \left(\frac{-8}{p}\right)$. This would seem like a convenient way to avoid the computation of class numbers of large fundamental discriminants, which in general is intractable as we will see later on, yet still obtain the class number of large discriminants.

However, the IQ-DLog problem in $\mathrm{Cl}(-8p^2)$, for p an odd prime, can be reduced in polynomial time to the DLog problem in $\mathbb{F}_p^*$ (if $p \equiv 1, 3 \mod 8$) or $\mathbb{F}_{p^2}^*$ (if $p \equiv 5, 7 \mod 8$). The latter can both be solved by a variant of the general number field sieve, which has complexity $L_\Delta[\frac{1}{3}, c]$ [HT99]. In general, the IQ-DLog problem in $\mathrm{Cl}(p^2\Delta_1)$, where $\Delta_1 < 0$ is such that $h(\Delta_1) = 1$ and p is an odd prime, can be reduced in polynomial time to the DLog problem in $\mathbb{F}_p^*$ (if $\left(\frac{\Delta_1}{p}\right) = 1$) or $\mathbb{F}_{p^2}^*$ (if $\left(\frac{\Delta_1}{p}\right) = -1$) [HT99, Theorem 1]. On the contrary, the best known algorithm for computing discrete logarithms in class groups $\mathrm{Cl}(\Delta)$ of arbitrary discriminant Δ has complexity $L_\Delta[\frac{1}{2}, 1]$. This motivates us to restrict ourselves to class groups of *maximal orders*, and therefore to pick a *fundamental discriminant* for cryptographic applications.

In order to check whether an arbitrary discriminant Δ is fundamental, it must be checked whether Δ (if $\Delta \equiv 1 \mod 4$) respectively $\Delta/4$ (if $\Delta \equiv 0 \mod 4$) is squarefree. Obviously this can be done by factoring the discriminant, but this is infeasible if the discriminant under consideration is large. A simpler method is to construct D from distinct prime factors, and set $\Delta = -D$ if $D \equiv 3 \mod 4$ and $\Delta = -4D$ otherwise. Some examples are:

(a) $\Delta = -p$ where $p \equiv 3 \mod 4$ is prime;

(b) $\Delta = -pq$ where p and q are primes s.t. $pq \equiv 3 \mod 4$ and $\left(\frac{p}{q}\right) = -1$.

(c) $\Delta = -8pq$ where p and q are primes s.t. $p \equiv 1 \mod 8$, $p + q \equiv 8 \mod 16$, and $\left(\frac{p}{q}\right) = -1$.

Note that these are all fundamental by construction. The additional advantage of selecting discriminants like this is that the two-part of $h(\Delta)$ is known to be small: in case (a), $h(\Delta)$ is odd by Proposition 1.19; in case (b), the even part of $h(\Delta)$ is exactly 2 [Réd28]; in case (c), the even part of $h(\Delta)$ is exactly 8 (see [Kap74, Proposition B_9']).

Furthermore, choosing $\Delta = -8pq$ might be attractive from a complexity theoretic point of view. In this case $C(\Delta)$ contains exactly 4 ambiguous classes by Example 1.11. More precisely, by [She13, Proposition 3.3.14], each class contains a PPD form of the shape

$$aX^2 + cY^2$$

with $ac = 2pq$. For $(a, c) \in \{(2p, q), (2q, p)\}$, knowing a and c immediately leads to a factorization of Δ. A similar argument holds up for case (b). Moreover, these ambiguous elements can be obtained by computing discrete logarithms in $\mathrm{Cl}(\Delta)$. More precisely, by finding the smallest k such that $H^{2^k} = 1$ for an appropriately constructed $H \in \mathrm{Cl}(\Delta)$, so that $H^{2^{k-1}}$ is ambiguous [SL84]. Therefore, in these cases FACT $\Rightarrow$ IQ-DLog, where we denote FACT for the integer factorization problem. This means that if Δ is chosen like this and p and q are kept secret, then solving the IQ-DLog problem for Δ is at least as hard as factoring the RSA modulus $N = pq$. Note that however, when implemented in a cryptographic setting, this requires a trusted setup, whereas simply picking a prime discriminant as in (a) does not. This motivates us to focus on discriminants of the form $\Delta = -p$ with $p \equiv 3 \mod 4$ prime for our computational experiments in section 3.

2.1.2 Index-Calculus Techniques

To get a grasp on the difficulty of the IQ-DLog and IQ-HO problem, we compare the computational work needed to solve these problems to the computational work needed to solve the integer

factorization problem. We assume the expected running time of factoring an integer n by the general number field sieve (GNFS) to be proportional to $L_n[\frac{1}{3}, \sqrt[3]{\frac{64}{9}}]$ (see for example [Ste08]). For the IQ-DLog, index-calculus algorithms with an expected running time proportional to $L_{|\Delta|}[\frac{1}{2}, \frac{3}{4}\sqrt{2}]$ (under the assumption of GRH) have been presented in [Vol00]. Moreover, Jacobson [Jac99] showed that one can use a variant of multiple polynomial quadratic sieve (MPQS) for discrete logarithm computations in $\mathrm{Cl}(\Delta)$. Improvements to this algorithm were made by [Bia10]. The MPQS factoring algorithm has a conjectured expected running time proportional to $L_n[\frac{1}{2}, 1]$ (see for example [Lan01]), but the CL-MPQS discrete logarithm computation has not been analyzed yet. Empirical data suggests an expected running time of $L_{|\Delta|}[\frac{1}{2}, 1]$. In [HM00] this running time is used to compare the estimated expected work of the GNFS for factoring integers and the CL-MPQS for computing discrete logarithms in class groups (see [HM00, Table 4] and [Ham02, Section 5.2.2]).

To determine the bit-length κ_Δ of the discriminant associated to a security parameter k, we consider the equality

$$2^\kappa c_\Delta = L_{2^{\kappa_\Delta}}[\tfrac{1}{2}, 1],$$

where the constant c_Δ can be extrapolated from actual running times of the MPQS algorithm. Based on [LV01, Table 1], a value $c_\Delta \approx 33.52686$ is suggested in [Ham02, Section 5.4]. Based on this constant, an algorithm is given to calculate κ_Δ in [Ham02, Algorithmus 5.3]. For security parameters $\kappa = 72, 80, 88$ this yields bit lengths $\kappa_\Delta = 671, 795, 929$, respectively [Ham02, Tabelle 5.11], which corresponds to factoring an RSA modulus of length $\kappa_{\mathrm{RSA}} = 1028, 1464, 2054$, respectively [LV01, Table 1]. More recent estimates propose that a bit-length of $\kappa_\Delta = 798, 1348, 1827$ corresponds to factoring an RSA modulus of length $\kappa_{\mathrm{RSA}} = 1024, 2048, 3072$, respectively [BJS10]. These moduli sizes correspond to security levels $\kappa = 80, 112, 128$, respectively, by the NIST recommendations [NIST20].

2.1.3 Pollard's ρ and λ Methods

We consider Pollard's ρ and λ methods for computing discrete logarithms, orders of group elements and hence roots of groups elements in *cyclic groups*. From [vOW99] it is known that the unparallelized version of Pollard's ρ algorithm takes $\sqrt{\pi|\mathbb{G}|/2}$ group operations (ignoring lower order terms) for cyclic groups $\mathbb{G}$. However, to use Pollard's ρ algorithm the group order must be known, so it is intractable to use in class groups. The runtime for Pollard's λ method is $2\sqrt{b}$ where $b \leq |\mathbb{G}|$ and we look for the discrete logarithm in the interval $[0, b)$. We will therefore only need an estimate of the group order to use it. Pollard's λ method becomes faster when $b < 0.39|\mathbb{G}|$. So if we have reason to believe that the discrete logarithm lies in a small interval, this might be faster than Pollard's ρ method [vOW99]. Moreover, r-fold parallelization speeds up the λ-method by a factor r.

By the Cohen-Lenstra heuristics [CL84, (C 1)], the probability that $\mathrm{Cl}_{\mathrm{odd}}(\Delta)$ (i.e. the odd part of the class group) is cyclic is approximately 0.977575. Moreover, it can be deduced from the heuristics that if $\mathrm{Cl}_{\mathrm{odd}}(\Delta)$ is not cyclic, then with high probability $\mathrm{Cl}_{\mathrm{odd}}(\Delta)$ has a cyclic subgroup $\mathbb{G}_{\mathrm{cyc}}$ such that $|\mathbb{G}_{\mathrm{cyc}}|$ is of nearly the same order of magnitude as $h_{\mathrm{odd}}(\Delta)$ [Ham02, Vermutung 5.1.11]. If we select Δ as in Section 2, the even part is 1, 2 or 8, and thus $|\mathbb{G}_{\mathrm{cyc}}|$ and $h(\Delta)$ have nearly the same order of magnitude. The above probabilities are all to be interpreted for randomly picking a negative fundamental discriminant Δ with $|\Delta| \leq D$ in the limit where D goes to infinity.

In order to provide a lower bound for the size of Δ we need an (asymptotic) lower bound for $h(\Delta)$ that only depends on Δ. However, the best proven explicit lower bound is too weak [HM00]. It follows from the Brauer-Siegel Theorem [Lan94, Theorem 4. on p.260] that for a fundamental discriminant Δ, we have (since the regulator of an imaginary quadratic field is 1 by definition)

$$\ln h(\Delta) \sim \ln \sqrt{|\Delta|} \quad \text{as } \Delta \to -\infty.$$

That is, for any $\varepsilon > 0$ and Δ sufficiently negative, one has

$$\sqrt{|\Delta|}^{1-\varepsilon} \leq h(\Delta) \leq \sqrt{|\Delta|}^{1+\varepsilon}.$$

This shows that the class group is large if the discriminant is large. However, no explicit constants are known to make this bound effective [HM00].

Assuming the generalized Riemann hypothesis, it can be shown that for $\Delta \neq -3, -4$:

$$\frac{1+o(1)}{c_2 \log\log|\Delta|}\sqrt{|\Delta|} < h(\Delta) < (1+o(1))c_3\sqrt{|\Delta|}\log\log|\Delta|, \tag{6}$$

where $c_2 = 12e^\gamma/\pi \approx 6.803$ and $c_3 = 2e^\gamma/\pi \approx 1.134$ [Lit28]. Here $\gamma \approx 0.577$ is Euler's constant. Since $\log\log|\Delta| < 8$ for all discriminants $|\Delta| \leq 2^{4300}$, this implies that for $c_4 = 8c_2 \approx 54.426$ we have [Ham02, Section 5.1.1]:

$$h(\Delta) > \frac{\sqrt{|\Delta|}}{c_4} > \frac{\sqrt{|\Delta|}}{2^6}. \tag{7}$$

Moreover, it can be shown that [Coh93, Section 5.10.1]:

$$\sum_{|\Delta|\leq D} h(\Delta) \sim \frac{D^{3/2}}{3\pi}C,$$

where the sum ranges over negative fundamental discriminants, with

$$C = \prod_{p \text{ prime}} \left(1 - \frac{1}{p^2(p+1)}\right) \approx 0.881538397$$

The number of fundamental discriminants asymptotically behaves as [Coh93, Section 5.10.1]:

$$|\{\Delta : |\Delta| \leq D, \quad \Delta < 0 \text{ fundamental}\}| \sim \frac{3}{\pi^2}D.$$

We can therefore conclude that on average the class number asymptotically behaves as

$$\overline{h(D)} := \frac{\sum_{|\Delta|\leq D} h(\Delta)}{|\{\Delta : |\Delta| \leq D, \quad \Delta < 0 \text{ fundamental}\}|} \sim c_1\sqrt{D},$$

where $c_1 = C\pi/9 \approx 0.30771495$. Under the assumption that this average is not affected by restricting Δ to a specific form, the expected running time of Pollard's λ for various orders of magnitude of $|\Delta|$ is given in [HM00]. In Section 3 we investigate whether this assumption holds up for some small discriminants of the form $\Delta = -p$ with $p \equiv 3 \mod 4$ prime.

If we were to apply the non-parallelized version of the λ-method over the whole interval $[0, |\mathbb{G}|)$ (note that we can not apply the ρ-method since it is intractable to compute the group order). Then for the class group of discriminant $\Delta < 0$ with $|\Delta| \leq 2^{4300}$, we would by equation (7) need at least $|\Delta|^{1/4}/4$ group operations. For a single group operation, we need time $O(\mu(\log|\Delta|)\cdot\log\log|\Delta|)$ using Schönhage's algorithm [Sch91], where $\mu(n)$ is the complexity of multiplying two n-bit integers [Ham02, Section 7.1].

2.1.4 The Pohlig-Hellman Algorithm and Smoothness Probabilities

The Pohlig-Hellman algorithm utilizes the prime factorization of the group order in order to simplify discrete logarithm computations. However, as we mentioned in Section 2.1.2, the best known *classical* algorithm for computing the class number is a variant of MPQS for discrete logarithm computations in class groups, which heuristically has the same expected asymptotic running time as the original MPQS (i.e. $L_{|\Delta|}[\frac{1}{2}, 1]$). Thus, if $|\Delta|$ is large, it is infeasible to compute $h(\Delta)$ or multiples or odd factors of $h(\Delta)$. In particular, it is infeasible to compute the smooth part of $h(\Delta)$. Moreover, there is no efficient method known that checks whether a particular odd prime divides $h(\Delta)$. Consequently, the Pohlig-Hellman algorithm is in general not applicable to class groups.

We therefore consider the special case where $h(\Delta)$ is smooth. Recall that an integer x is called *y-smooth* if all of the primes p dividing x are less or equal to y. Later we will see that in fact the probability that $h(\Delta)$ of a randomly chosen fundamental discriminant is smooth is very small. If the class number is smooth, then it is possible to compute the order of an arbitrary element by a method similar to the $(p-1)$-factoring algorithm. That is, given $\gamma \in \mathrm{Cl}(\Delta)$, recursively compute

$$\alpha_0 = \gamma, \quad \alpha_i = \alpha_{i-1}^{p_i^{e(p_i,B)}}$$

for all $p_i \leq B$, where p_i is the i-th prime, B is a smoothness bound, and $e(p_i, B)$ depends only on p_i and B. For instance, if $e(p_i, B) = \lfloor \log_{p_i} B \rfloor$ for each p_i, then the algorithm will cover each possible prime power below the smoothness bound. A similar method is used in the factoring algorithm of Schnorr and Lenstra [SL84].

If $h(\Delta)$ is B-smooth, then this computation may yield 1 at some point. If this happens, then there is an i such that $\alpha_i = 1$ but $\alpha_{i-1} \neq 1$, and we immediately know that p_i is the largest prime dividing $|\langle\gamma\rangle|$. If we subsequently set $\gamma' = \gamma^{p_i^{e(p_i)}}$, where $e(p_i)$ is the smallest positive integer such that $\alpha_{i-1}^{p_i^{e(p_i)}} = 1$, and repeat the whole procedure with γ'. Then we obtain the second largest prime dividing $|\langle\gamma\rangle|$, and eventually we get the complete prime factorization of $|\langle\gamma\rangle|$. Finally we are able to compute roots as well as discrete logarithms in $\langle\gamma\rangle$ by applying the Pohlig-Hellman algorithm.

Assume that the $(p-1)$-like method described above succeeds for an element γ and bound B, and let q denote the largest prime dividing $|\langle\gamma\rangle|$. If we use a fast exponentiation method, then we have to perform at least $\sum_{p<q} e(p, B) \cdot \lfloor \log_2 p \rfloor$ group operations to find q. In order to find a smoothness bound, we first consider the easiest case, i.e. $e(p_i, B) = 1$ for all p_i. Now

$$\sum_{p<q} \lfloor \log_2 p \rfloor \leq \theta(q)/\log 2,$$

where θ is the Chebyshev θ-function. Rosser and Schoenfeld showed that [RS75, Thm. 6]:

$$0.998684x < \theta(x) < 1.001102x \quad \text{for all } x \geq 1319007.$$

Under the assumption of the Riemann hypothesis, Schoenfeld showed that [Sch76, Thm. 10]:

$$|\theta(x) - x| < \frac{1}{8\pi}\sqrt{x}\log^2(x) \quad \text{for all } x \geq 599.$$

Therefore, to find q we have to perform about $q/\log 2$ group operations. Note that we would have got the same result even in the worst case where $e(p, q) = \lfloor \log_p q \rfloor$ for all $p < q$, since

$$\begin{aligned}\sum_{p<q} \lfloor \log_2 p \rfloor \cdot \lfloor \log_p q \rfloor &\leq \pi(q)\log_2 q \\ &\sim q/\log 2 \quad (\text{as } q \to \infty),\end{aligned} \tag{8}$$

where $\pi(q)$ denotes the number of primes less than or equal to q. We will now continue to discuss the smoothness probability of the class number $h(\Delta)$ and use these results to determine lower bounds on the required size of Δ.

Smoothness Probability of Class Numbers

Hamdy and Saidak stated a special case of the Cohen-Lenstra heuristics [CL84] specifically for imaginary quadratic class groups, and slightly extended them to arrive at the folowing heuristics.

Conjecture 2.1 ([HS06, Conjecture 1.1])**.** *Let p be an odd prime and $\Delta < 0$.*

(1) (Divisibility). *The probability $\mathfrak{V}(p)$ that p divides $h(\Delta)$ is equal to*

$$\mathfrak{V}(p) = 1 - \prod_{n=1}^{\infty}\left(1 - \frac{1}{p^n}\right) = \frac{1}{p} + \frac{1}{p^2} - \frac{1}{p^5} - \frac{1}{p^7} + \ldots.$$

(2) (Uniformity). *The property* (1) *can be refined as follows: As $N \to \infty$,*

$$\#\{|\Delta| \leq N \,:\, h(\Delta) \equiv 0 \mod p\} \sim N\mathfrak{V}(p),$$

where p is fixed, or - more generally - a prime that satisfies $p = o(N^\alpha)$ for all $\alpha > 0$.

(3) (Independence). *If p and q are fixed odd primes, and $\mathfrak{V}(pq)$ denotes the probability that both p and q divide $h(\Delta)$, then*

$$\mathfrak{V}(pq) = \mathfrak{V}(p)\mathfrak{V}(q).$$

More generally, the density statement remains true as $N \to \infty$, as long as $p, q = o(N^\alpha)$ for all $\alpha > 0$.

(4) (Equidistribution). *For a given prime p and an integer $0 \leq a < p$, define $\mho_a(p)$ to be the probability that $h(\Delta) \equiv a \mod p$. Then all moduli a are equidistributed, i.e.*

$$\mho_a(p) = \frac{1}{p-1} \cdot \mho(p) = \frac{1}{p-1}\left(\frac{1}{p} + \frac{1}{p^2} - \frac{1}{p^5} - \frac{1}{p^7} + \dots\right)$$

Here part (1), (2) and (3) follow implicitly from ideas behind the Cohen-Lenstra Heuristics [CL84], and (4) is a new addition from [HS06] which the authors claim to be supported by abundant numeral evidence. Under the assumption of these conjectures, the following result is shown.

Theorem 2.2 ([HS06, Theorem 3.1])**.** *Let D be a positive integer, H an upper bound for the class numbers of fundamental discriminants $\Delta < 0$ with $|\Delta| \leq D$. Let $u > 1$ be a fixed real number, and let $B := H^{1/u}$ be a smoothness bound. Let $\Delta < 0$ be a randomly chosen fundamental discriminant such that $|\Delta| \leq D$, and let N be a randomly chosen integer such that $1 \leq N \leq H$. Then under the assumption of Conjecture 2.1, we have*

$$\lim_{D\to\infty} \frac{\Pr[h(\Delta) \textit{ is } B\textit{-smooth}]}{\Pr[N \textit{ is } B\textit{-smooth}]} = 1.$$

Remark. By the inclusion-exclusion principle, we have

$$\Pr[N \text{ is not B-smooth}] = \sum_{i>0} (-1)^{i-1} S_i \tag{9}$$

with

$$\lim_{D\to\infty} S_i = \sum_{p_1,\dots,p_i} \prod_{p_j} \frac{1}{p_j} \tag{10}$$

such that each product of primes appears only once. Here $\sum_{p_1,\dots,p_i}$ denotes the sum over all primes $B < p_1 \leq \dots \leq p_i$ such that $p_1 \cdots p_i \leq H$, and each S_i is the sum of products that involve exactly i primes (note that $H \to \infty$ as $D \to \infty$). Moreover, note that we have $S_i = 0$ for all $i > u$.

Similar expressions to (9) and (10) can be found for class numbers if one replaces $1/p$ by $\mho(p)$ in equation (10). This gives the expression

$$\Pr[h(\Delta) \text{ is not B-smooth}] = \sum_{i>0} (-1)^{i-1} T_i$$

with

$$\lim_{D\to\infty} T_i = \sum_{p_1,\dots,p_i} \prod_{p_j} \mho(p_j).$$

Again, $T_i = 0$ for all $i > u$. The proof of the theorem follows from $\lim_{D\to\infty} T_i/S_i = 1$ for all $i \leq u$.

Proof. See [HS06, Theorem 3.2] for a formal proof. □

If we again let $B := H^{1/u}$ with the same language as in Theorem 2.2, then the probability that a random positive integer $1 \leq N \leq H$ is B-smooth is approximately

$$\rho(u)\left(1 + O_\epsilon\left(\frac{\log(u+1)}{\log B}\right)\right) \tag{11}$$

for $B \geq 2$, $1 \leq u \leq \exp((\log(B)^{3/5-\epsilon})$ and $\epsilon > 0$, and where ρ is Dickman's ρ-function [HS97]. By Theorem 2.2 we therefore arrive at the the same probability (11) for the class number $h(\Delta)$ of a randomly chosen fundamental discriminant $|\Delta| \leq D$ being B-smooth.

Assume that an attacker applies the $(p-1)$-like method as described in Section 2.1.4 to class groups of random discriminants of a certain length. We want to determine the size of D such that the attacker can not with high probability find the biggest prime factor of the class number $h(\Delta)$ belonging to a randomly chosen fundamental discriminant Δ with $|\Delta| \leq D$. As in [Ham02, Section 5.2.4], let $W_{\max}$ be the maximal amount of computational work the attacker is willing to do for a randomly chosen discriminant Δ and a randomly chosen $\gamma \xleftarrow{\$} \mathrm{Cl}(\Delta)$. When p is the greatest prime

divisor of $h(\Delta)$, the probability that p does *not* divide the order of γ is $\leq 1/p$ by Lemma 1.25. Hence this probability becomes negligible when $h(\Delta)$ has a superpolynomially large prime divisor. Assuming that the largest prime factor p of $h(\Delta)$ divides $|\langle\gamma\rangle|$, implies that $|\langle\gamma\rangle|$ is B-smooth if and only if $h(\Delta)$ is B-smooth. In this case the attacker can expect to succeed once for an investment of computational work

$$W_{\text{total}} = \frac{W_{\max}}{\Pr[h(\Delta) \text{ is B-smooth}]}.$$

The probability $\Pr_{\Delta,B} := \Pr[h(\Delta)$ is B-smooth] that the class number $h(\Delta)$ of a randomly chosen fundamental discriminant Δ, with $|\Delta| \leq D$, is B-smooth is approximately $\rho(u)$ as seen above. If we put $H = \sqrt{D}$ as an upper bound for the class numbers of fundamental discriminants Δ with $|\Delta| \leq D$, then $B = D^{1/2u}$. This allows us to derive a bound on the minimum size of $|\Delta|$ for a given amount of total computational work W_{total} and a given smoothness probability $\Pr_{\Delta,B}$.

Example 2.3. Assume one desires a smoothness probability of $\Pr_{\Delta,B} = 2^{-32}$. Then we can determine the smoothness bound B and the size of $|\Delta|$ for which a total amount of work $W_{\text{total}} = 2^{64}$ is needed to achieve one case of success with the $(p-1)$-like method described in Section 2.1.4. This gives $W_{\max} = 2^{32}$, and from equation (8) we obtain the smoothness bound $B = 2^{32}$ (ignoring the term $1/\log(2) \approx 1.443$). For a randomly chosen fundamental discriminant Δ with $|\Delta| \geq B^{2u}$, with smoothness bound $B = 2^{32}$, we desire a smoothness probability $\Pr_{\Delta,B} \leq 2^{-32}$. Taking $\Pr_{\Delta,B} = \rho(u)$, we obtain that $u \approx 9.42$ and therefore that $|\Delta| \geq 2^{603}$. In [Ham02, Tabelle 5.8] the outcomes of these calculations for several values of W_{total} and $\Pr_{\Delta,B}$ can be found.

We want to stress that the above example only accredits the amount of work that is needed to be done for an attacker to retrieve the largest prime divisor of $h(\Delta)$ (in the setting we described) and does not take into account the additional amount of work that needs to be done when subsequently applying the Pohlig-Hellman algorithm to compute discrete logarithms.

For the cryptographic applications discussed in this survey, it is important to study whether heuristic assumptions such as those given in Conjecture 2.1 still hold for our restricted choice of discriminants discussed in Section 2.1.1. Some data supporting this assumption is given in [Ham02, Annahme 5.2.4]. In Section 3 we calculated some empirical smoothness probabilities for class numbers belonging to discriminants $\Delta = -p$ with $p \equiv 3 \mod 4$ and prime.

2.1.5 Sutherland's Algorithm and Semismoothness Probabilities

Dobson, Galbraith and Smith [DGS20] suggest that previously proposed discriminant sizes for class groups of imaginary quadratic number fields are not sufficient to reach the security level they claim to as an instantiation of a group of unknown order. They claim that the main reason for this is that the impact of Sutherland's generic group order algorithm [Sut07] has not been recognized in applications where a fixed group is used for many users and a long period of time [DGS20]. This is because there is some small probability a "bad" group has been sampled for which Sutherland's algorithm can compute its order relatively quickly. For this context they propose the following security definition.

Definition 2.4 ([DGS20, Definition 1])**.** Let Gen be an algorithm outputting groups. We say that Gen reaches security level (λ, ρ) if with probability $1 - 1/2^{\rho}$ over the outputs of Gen, any algorithm $\mathcal{A}$ given an output $\mathbb{G}$ of Gen requires at least 2^{λ} bit operations to succeed in computing $|\mathbb{G}|$ with probability close to 1.

Remark. For the constructions of verifiable delay functions (VDFs) discussed in Section 4.1, when implemented in IQ class groups, it is easy to sample a fresh group $\mathbb{G} = \mathrm{Cl}(\Delta)$ for each VDF instance by letting Gen sample a random discriminant Δ (e.g., of the form $\Delta = -p$ with $p \equiv 3 \mod 4$ and prime). In this context it is more natural to consider the average complexity of $\mathcal{A}$ on input $\mathbb{G} \xleftarrow{\$} \mathsf{Gen}$, which will be at least $2^{\lambda+\rho}$ bit operations. We discuss this setting in more detail in Section 4.1.4 when we discuss the security of VDFs based on hidden-order groups in a post-quantum setting.

In this section we discuss Sutherland's group order algorithm and how this affect the discriminant size for imaginary quadratic class groups.

Sutherland's algorithm provides a way of computing the order of an element in a generic group (e.g., an abelian group of unknown order). It is inspired by Shank's baby-step giant-steps method and Pollard's rho algorithm, and is also referred to as the primorial-steps algorithm [Sut07]. Sutherland's algorithm is a probabilistic algorithm, whose success probability highly depends on the semismoothness probability of the group order $|\mathbb{G}|$.

Semismoothness Probabilities

Definition 2.5. Let n denote a positive integer and let x, y and z be real numbers. Then n is *semismooth* with respect to y and z if all of its prime factors are bounded by y and all but at most one are bounded by z. The function $\psi(x, y, z)$ counts the positive integers up to x which are semismooth with respect to y and z.

0 Note that a number that is semismooth with respect to y and z is y-smooth, and is at most one prime factor away from being z-smooth. Typically, one is interested in $\psi(x, x^s, x^r)$ for some $r < s$, most commonly $r = 1/u$ and $s = 2/u$. The function $G(r, s)$ estimates the asymptotic probability that an integer in $[1, x]$ is semismooth with respect to x^s and x^r.

Definition 2.6. For $0 < r < s < 1$, the *semismooth probability function* $G(r, s)$ is defined by

$$G(r, s) = \lim_{x \to \infty} \psi(x, x^s, x^r)/x.$$

Note that the order of r and s is reversed in the definition of $G(r, s)$, so that the smallest number comes first. See [Ber02, BS06, BS13] for ways to accurately compute semismoothness probabilities. Also see [BP96] for more on semismoothness probabilities.

Fast Order Algorithms

Fast order algorithms are algorithms that can compute the order of an element $\alpha \in \mathbb{G}$ efficiently in the case where we are given either a set containing the prime factors of $|\langle\alpha\rangle|$, or an integer N such that $\alpha^N = 1_{\mathbb{G}}$, along with its prime factorization. The first case can be seen as a special case of the second. Since if S contains the prime factors of $|\langle\alpha\rangle|$, the implicit bound given by the size of the identifiers implies that $N = \prod_{p \in S} p^{\lfloor \log_p 2^\ell \rfloor}$ is a multiple of $|\langle\alpha\rangle|$.

Definition 2.7. An integer N such that $\alpha^N = 1_{\mathbb{G}}$, together with a factorization $N = n_1^{h_1} \cdots n_w^{h_w}$ satisfying $\gcd(n_i, n_j) = 1$ for $i \neq j$, is called a *factored exponent* of α. An integer M of the form $M = n_1^{e_1} \cdots n_w^{e_w}$ which divides N is called a *compatible divisor* of N. The *pseudo-order* of α relative to the factored exponent N is the least compatible divisor M for which $\alpha^M = 1_{\mathbb{G}}$, and is denoted $|\langle\alpha\rangle|_N$.

Note that when the notation $|\langle\alpha\rangle|_N$ is used, a specific factorization of N is assumed to have been given. Moreover, note that the set of compatible divisors of N, as well as the set set of exponents of α, are both closed under taking gcd's. The integer $|\langle\alpha\rangle|_N$ is the unique minimal member of the set of compatible divisors of N.

Definition 2.8. A *fast order algorithm* $\mathcal{A}(\alpha, N)$ is a generic group algorithm which computes $|\langle\alpha\rangle|_N$, given a factored exponent N of α.

When the *prime* factorization of N is given, $|\langle\alpha\rangle|_N$ will be equal to $|\langle\alpha\rangle|$. More general, if $|\langle\alpha\rangle|$ is a compatible divisor of N, then $|\langle\alpha\rangle|_N = |\langle\alpha\rangle|$, but in general they are not guaranteed to be equal. If the prime factorization of N is not available, computing $|\langle\alpha\rangle|_N$ is often just as good as computing $|\langle\alpha\rangle|$ [Sut07, p.109].

Sutherland gives a construction of a fast order algorithm which uses $(1 + o(1))n$ group operations and space polylogarithmic in n, where $n = \log_2 N$ [Sut07, Algorithm 7.1].

The Primorial Steps Algorithm

Sutherland's algorithm for order computations in black-box groups is also referred to as the primorial-steps algorithm. It can be implemented in a bounded and unbounded way [Sut07, Algorithm 4.1 and Algorithm 4.2, respectively]. We state the bounded version of the algorithm in Algorithm 2.10 since it is a bit more concise. As the name suggests, the algorithm makes use of primorials, which are defined as follows.

Definition 2.9. The w-th *primorial* P_w is the product of the first w primes, that is $P_w = \prod_{p \leq p_w} p$, with the convention that $P_0 = 1$. Note that the Euler phi of a primorial P_w is given by

$$\phi(P_w) = \prod_{p \leq p_w} (p-1).$$

Algorithm 2.10. *[Sut07, Algorithm 4.1] Given $\alpha \in \mathbb{G}$, $M \geq |\langle\alpha\rangle|$, $M' \leq M$, and a fast order algorithm $\mathcal{A}(\alpha, E)$ (see Definition 2.8). The algorithm computes $|\langle\alpha\rangle|$ as follows.*

1. **Compute β:** Maximize P_w such that $P_w \leq \sqrt{M'}$. Then maximize m such that $m^2 P_w \phi(P_w) \leq M'$. Set $b \leftarrow mP_w$, $b' := m\phi(P_w)$ $E \leftarrow \prod_{p \leq p_w} p^{\lfloor \log_p M \rfloor}$, and compute $\beta = \alpha^E$.
2. **Baby Steps:** For $j = 1, \ldots, b$, if $\gcd(j, P_w) = 1$, compute β^j and store (β^j, j) in a table. If any $\beta^j = 1_{\mathbb{G}}$, set $N' \leftarrow j$ and go to step 4.
3. **Giant Steps:** For $i = 2b, 3b, \ldots, b'b$, compute β^i and do a table lookup. When $\beta^i = \beta^j$, set $N' \leftarrow i - j$ and go to step 4.
4. **Compute $|\langle\alpha\rangle|$:** Compute $h \leftarrow \mathcal{A}(\alpha^{N'}, E)$ and return hN'.

The bounded version performs best on average when $M' := (\zeta(2)/2) \cdot M$.

In the unbounded version of the algorithm, the dependency of the choice of P_w on M is removed. The dependency on M of the prime powers in E is changed to depend on the group element identifier length $\ell = O(\log_2 |\mathbb{G}|)$. In this case a growth function $d(k)$ is used to determine the number of steps to take in each iteration, and a bound L on the prime factors in the exponent E used to compute $\beta = \alpha^E$. See [Sut07, Section 4.2] for more details on this version of the algorithm. The main result concerning the complexity of this algorithm is the following.

Theorem 2.11. *[Sut07, Proposition 4.7.] Let $T(N)$ be the number of group operations used by the unbounded primorial steps algorithm on input α with $|\langle\alpha\rangle| = N$. If $|\langle\alpha\rangle|$ is distributed uniformly over $[1, M]$, $\ell = O(\log_2(M))$, and $L = M^{1/u}$. Then there is a constant c such that*

$$\Pr[T(|\langle\alpha\rangle|) \leq cM^{1/u}] \geq G(1/u, 2/u), \tag{12}$$

for all sufficiently large M, where $G(r, s)$ is the semismooth probability function (see Def. 2.6).

It is possible to alter the algorithm to limit the space used by the algorithm and prevent the use of a prime bound L by using a multi-stage sieve. See [Sut07, Chapter 5].

For the parameters achieving the complexity stated in Theorem 2.11, the first step in the primorial steps algorithm [Sut07, Algorithm 4.2] requires an exponentiation of the element α to the power $E := \prod_{p \leq M^{1/u}} p^{\lfloor \log_p 2^\ell \rfloor}$, which takes roughly

$$\log_2 E = \sum_{p \leq M^{1/u}} \log_2 p \cdot \lfloor \log_p 2^\ell \rfloor \leq \sum_{p \leq M^{1/u}} \ell \sim \frac{\ell u \cdot M^{1/u}}{\log M} = \frac{cu \cdot M^{1/u}}{\log 2}$$

sequential multiplications in the group $\mathbb{G}$, where M is an upper bound on the order of the element α and we put $\ell = c \cdot \log_2 M$. In Section 4.1 we will introduce delay functions, which rely on the hardness of computing a large exponentiation of a random element $\alpha \xleftarrow{\$} \mathbb{G}$ in less than T sequential multiplications. The primorial steps algorithm will therefore only help to speed up this computation as long as $\log_2 E \ll T$, which approximately happens when $M^{1/u} \ll T$. Since M needs to be an upper bound on the order of a random elements, it needs to be an upper bound on the order of the group $\mathbb{G}$. Furthermore, the semismoothness probability $G(1/u, 2/u)$ from equation (12)

decreases quickly as u increases. For example, $u = 2.1$ gives ≈ 0.9488 whereas $u = 10$ already gives $\approx 5.382 \cdot 10^{-9}$. We therefore judge that the primorial steps algorithm does not pose a large threat for the delay functions discussed in Section 4.1 as long as the delay parameter $T < |\mathbb{G}|^{1/u}$, which still allows for an exponentially large range of parameters. Here we assume that a fresh random group of unknown order is sampled for each instance of the delay function, which can be done relatively easy and without a trusted setup for imaginary quadratic class groups (in one of the ways discussed in Section 2.1.1).

For applications where a fixed group is used for a longer period of time one needs to consider a more conservative security parameter than we discussed in Section 2.1.2 and Section 2.1.3. Let Gen be an algorithm outputting groups, and assume that with probability at least $1-1/2^\rho$ over $\mathbb{G} \leftarrow \mathsf{Gen}$, any algorithm $\mathcal{A}$, on input $\mathbb{G}$, requires at least 2^λ bit operations to succeed in computing $|\mathbb{G}|$ with probability close to 1. For applications where $\mathbb{G} \leftarrow \mathsf{Gen}$ is only used for a single instantiation of some computational problem, one would argue that this leads to a security level of $\kappa := \lambda + \rho$ (that is, any adversary needs at least $2^{\lambda+\rho}$ bit operations). However, if a single $\mathbb{G} \leftarrow \mathsf{Gen}$ is used for a longer period of time and by many users, we need to consider that with probability $1/2^\rho$ a "bad" group might have been sampled, for which the problem is significantly easier to solve than 2^λ operations. A key problem here is that for proposed constructions of hidden order groups that do not need a trusted setup, there is no efficient way to check whether a randomly sampled group is "bad". In a setting where there is a trusted setup, such as the RSA setting, the group order is known to the group generator, who can make sure that a "good" group is chosen. For the trustless setting Dobson, Galbraith and Smith argue to consider a security level $\kappa := (\lambda, \rho)$ [DGS20]. For example, based on the complexity of Sutherland's algorithm (see Section 2.1.5), the authors argue that security level $(\lambda, \rho) = (128, 55)$ requires ≈ 3840-bit discriminants, which is more than double the bit-length $\kappa_\Delta = 1827$ suggested in the previous paragraph for a $\kappa = 128$ security level. An even more conservative choice $(\lambda, \rho) = (128, 128)$ would require 6784-bit discriminants. See [DGS20, Table 2] for discriminant sizes required for various other choices of security level (λ, ρ).

2.2 Post-Quantum Setting

In the post-quantum setting, the overview is quite simple. There is essentially one quantum algorithm which can compute the order of any finite abelian group and can compute discrete logarithms in finite abelian groups, both in polynomial time. This algorithm was discovered by Shor [Sho97a] and is referred to as Shor's algorithm. This does however not mean that hidden-order groups such as imaginary quadratic class groups are broken for every application. We can still use a hidden-order group for an instance of a verifiable delay function (VDF, see Section 4.1) as long as the timing parameter of the VDF is set below the runtime of the polynomial time order computation algorithm and a fresh random group is sampled for every VDF instance.

2.2.1 Abelian Hidden Subgroup Algorithms

Providing all the details of the quantum circuit computed in Shor's algorithm is out of scope for this review. We rather describe the problem it is able to solve efficiently and show how the discrete logarithm problem and the problem of computing the order of an element in an abelian group can be seen as a special case. We moreover discuss the gate complexity and circuit depth of the main components of the algorithm. Shor's algorithm [Sho97a] was originally discovered as an efficient quantum algorithm for solving the *hidden shift* problem, of which the integer factorization problem and the problem of computing the order of an element in an abelian group can be seen as a special case. Later works generalized the algorithm to be able to efficiently solve the *hidden subgroup* problem in abelian groups [Kit96, ME98].

Definition 2.12 (Abelian hidden subgroup problem)**.** Let G be an abelian group and let $f : G \to S$ be a function to some finite set S. Suppose that f has the property that there exists a subgroup $H \subset G$ such that f is constant within each coset, and distinct on different cosets, that is:

$$f(g) = f(g') \iff gH = g'H.$$

Then the *hidden subgroup problem* for f is the problem of finding this subgroup H given black-box access to the function f.

The discrete logarithm problem can be seen as a special case of the hidden subgroup problem as follows. We show this for the DLog_2 problem for brevity. Say we want to compute the discrete logarithm of an element $Y := X^d$ with respect to $X \in \mathbb{G}$. Let $N := |\langle X \rangle|$ and consider the function

$$f : \mathbb{Z}_N \times \mathbb{Z} \to \langle X \rangle, \quad (a, b) \mapsto X^a Y^{-b}.$$

Then for elements $(a_1, b_1), (a_2, b_2) \in \mathbb{Z}_N \times \mathbb{Z}$ (considered as an additive group) we have

$$X^{a_1} Y^{-b_1} = X^{a_2} Y^{-b_2} \iff a_1 - a_2 \equiv d(b_1 - b_2) \mod N \iff (a_1, b_1) - (a_2, b_2) \in \langle (d, 1) \rangle.$$

So we see that finding the subgroup $H := \langle (d, 1) \rangle$ of $\mathbb{Z}_N \times \mathbb{Z}$ gives us the discrete logarithm d of Y with respect to X. Here we assume that the order $N = |\langle X \rangle|$ is known, but in fact the order of any element $X \in \mathbb{G}$ can be readily computed by applying Shor's original period-finding algorithm [Sho97a] to the map

$$f : \mathbb{Z} \to \mathbb{G}, \quad a \to X^a,$$

for which $f(a) = f(b) \iff a \equiv b \mod |\langle X \rangle|$.

The above can be generalized to the multiple element representation case as in DLog_1 by considering the map

$$f : L(\boldsymbol{g}) \times \mathbb{Z} \to \mathbb{G}, \quad (a_1, \ldots, a_n, b) \mapsto g_1^{a_1} \cdots g_n^{a_n} X^{-b},$$

where $L(\boldsymbol{g})$ is the relation lattice of $\boldsymbol{g} := (g_1, \ldots, g_n)$. That is, the lattice consisting of all integer vectors $\boldsymbol{e} := (e_1, \ldots, e_n) \in \mathbb{Z}^n$ for which $g_1^{e_1} \cdots g_n^{e_n} = 1_{\mathbb{G}}$. The structure of the relationship lattice for $\boldsymbol{g} \in \mathbb{G}^n$ can be computed by considering the hidden subgroup problem for the map

$$f : \mathbb{Z}^n \to \mathbb{G}, \quad (a_1, \ldots, a_n) \to g_1^{a_1} \cdots g_n^{a_n},$$

for which $f(a_1, \ldots, a_n) = f(b_1, \ldots, b_n) \iff (a_1, \ldots, a_n) - (b_1, \ldots, b_n) \in L(\boldsymbol{g})$.

First we discuss the number of qubits, number of elementary quantum gates and circuit depth, together with some possible trade-offs, when implementing Shor's algorithm to factor RSA moduli. Afterwards we will make a rough translation of this discussion to the setting of implementing a generalization of Shor's to solve the hidden subgroup problem in imaginary quadratic class groups. We will consider the cost of computing the order of a single (random) element $X \xleftarrow{\$} \mathrm{Cl}(\Delta)$ as a lower bound for the cost of computing the order $h(\Delta) = |\mathrm{Cl}(\Delta)|$.

Integer Factorization

Implementing Shor's algorithm for the factorization of an n-bit integer (e.g., an RSA modulus $N = pq$ of length $n = \lceil \log_2 N \rceil$) can in theory be done as follows: A register consisting of $2n$ qubits and a register consisting of n qubits are both initialized to the all 0's state. First $2n$ Hadamard gates are applied to the first register to obtain a uniform superposition over exponents $a = 0, \ldots, 2^{2n} - 1$. Then, for a random $x \xleftarrow{\$} \mathbb{Z}_N^*$, the function $a \mapsto x^a$ is evaluated in the second register over the superposition obtained in the previous step. This takes roughly $2n \cdot n \log n = 2n^2 \log n$ elementary qubit-operations, assuming an asymptotically optimal multiplication algorithm such as Harvey-van der Hoeven [HvdH21] can be implemented in an asymptotically equivalent number of elementary qubit gates. Moreover, the quantum Fourier transform used in the algorithm can be implemented using $O(n^2)$ elementary gates and all steps in the algorithm need to be repeated an expected $\log n$ times. Ultimately this adds up to $O(n^2 \log^2 n)$ elementary gates in theory.

However, in practice the number of elementary quantum gates needed to implement fast multiplication algorithms such as Schönhage-Strassen [SS71], Karatsuba [KO62] or Harvey-van der Hoeven [HvdH21] might be higher than their classical complexity [GE21]. This is mainly because quantum computations need to be *reversible*, and translating these algorithms to a reversible form might introduce additional overhead in the number of gates or the number of qubits needed. Recall that we can measure the gate complexity by the number of Toffoli gates needed, since these are complete for classical circuits. A second reason to use an integer multiplication algorithm with $O(n^2)$ complexity is because the constants hidden behind the $O(n^{\log 3})$ [KO62], $O(n \log n \log \log n)$ [SS71] and $O(n \log n)$ [HvdH21] of asymptotically fast multiplication algorithms will usually result in a higher concrete complexity for the sizes of numbers we actually need to multiply in practice.

For example, for $n = 2048$ the Toffoli count of a $O(n^2)$ multiplication algorithm is still lower than that of asymptotically faster multiplication algorithms. This leads to a complexity of $O(n^3 \log n)$ elementary qubit gates.

The number of qubits can be brought down to $2n + 3$ while increasing the number of elementary quantum gates to $O(n^3 \log n)$ and the circuit depth to $O(n^3)$ [Bea03]. This increase in circuit complexity is because modular addition and multiplication are trickier to realize with a lower number of qubits and since an $O(n^2)$ integer multiplication algorithm is concretely more efficient as discussed above. Moreover, a lower number of control qubits leads to a larger number of operations needing to be done sequential, which could otherwise be parallelized. More recent works achieve a slightly better trade-off between the number of qubits, the number of gates and the circuit depth, see [GE21, Table 1] for an overview. Gidney and Ekerå manage to achieve a $O(n^2 \log n)$ circuit depth using optimized modular additions, with $3n + 0.002n \log n$ qubits and $O(n^3 \log n)$ elementary quantum gates. We want to note that all the above estimates are in the *abstract circuit model* and consider logical qubits. For estimates considering the number of physical qubits and the expected runtime in a more realistic practical setting where there are overheads from distillation, routing and error correction, we refer to [GE21, Table 2].

Class Group Order Computation

Implementing an abelian hidden subgroup algorithm for the computation of the order of an element $X \in \mathrm{Cl}(\Delta)$ with $n = \lceil \log_2 |\Delta| \rceil$ can in theory be done similarly as for factoring. Evaluating the function $a \mapsto X^a$ for exponents of length around $n \approx \log_2 |\mathrm{Cl}(\Delta)|^2$ bits takes $O(n)$ multiplications in $\mathrm{Cl}(\Delta)$. Each multiplication takes $O(\mu(n) \cdot \log n)$ bit operations using Schönhage's algorithm [Sch91], where $\mu(n)$ is the complexity of multiplying two n-bit integers [Ham02, Section 7.1]. Again assuming an asymptotically optimal quantum implementation of an asymptotically optimal integer multiplication algorithm, this leads to $O(n \log^2 n)$ elementary qubit operations. Similarly we assume that there exists a circuit evaluating f on a superposition of inputs using $O(n^2 \log^2 n)$ elementary qubit gates. Again the quantum Fourier transform can be implemented using $O(n^2)$ elementary gates and all steps in the algorithm need to be repeated an expected $\log n$ times. Ultimately this leads to the assumption that it is possible to compute the order of X using $O(n^2 \log^3 n)$ elementary gates in theory. Again the number of necessary qubits would lie around $3n$, assuming that we can represent the elements of $\mathrm{Cl}(\Delta)$ using n qubits and that all operations can be implemented on these representations without incurring any additional overhead.

However, similar to before, it might be more efficient in practice to use an $O(n^2)$ integer multiplication algorithm. Assuming that the additional steps in Schönhage's quadratic form composition algorithm [Sch91] incur no additional overhead when implemented in a quantum circuit, we arrive at an estimated $O(n^3 \log^2 n)$ elementary qubit gates to compute the order of $X \in \mathrm{Cl}(\Delta)$. Additionally Jacobson, Sawilla and Williams [JSW06a] argue that in practice Shanks' NUCOMP algorithm (in the formulation from [JSW06b]) is more efficient than Schönhage's algorithm [Sch91] for performing multiplications in imaginary quadratic class groups. NUCOMP has an asymptotic complexity of $O(\mu(n) \cdot n)$, where $\mu(n)$ is the complexity of n-bit integer multiplication [JSW06b]. Combined with an $O(n^2)$ integer multiplication algorithm this leads to an estimated $O(n^4 \log n)$ elementary qubit gates to compute the order of $X \in \mathrm{Cl}(\Delta)$. We are not aware of any works discussing the implementation of the NUCOMP algorithm in an abstract quantum circuit.

As far as we know there are no works discussing more practical challenges one might face when implementing the order computation algorithm for imaginary quadratic class groups in a quantum circuit, such as the works [Bea03] and [GE21] do for integer factorization. It is left to future work to give a more accurate estimate for the complexity of order computations in imaginary quadratic class groups in the abstract circuit model as well as in a more realistic practical setting.

In a post-quantum setting where a fixed group is used for a longer amount of time, imaginary quadratic class groups will most likely not be suitable. For such applications, due to the polynomial quantum order computation methods described above, the discriminant would need to be chosen so large that group operations will become too inefficient to compute for honest parties. For example, for RSA it has been suggested that using a 2^{43} bit (around 1 terabyte) modulus offers around 100 bits of post-quantum security [BHLV17]. However, even assuming an optimal integer multiplication algorithm, a similarly sized discriminant would lead to class group multiplications costing roughly

2^{53} bit operations using Schönhage's algorithm [Sch91] (ignoring the constants hidden in the big-O notation, which would only lead to a higher concrete complexity).

In a post-quantum setting where a security guarantee only needs to hold for a relatively short amount of time, one can consider a more fine-grained notion of security. If the security guarantee for example relies on the hardness of the multiple order problem MO, we can regularly sample a fresh group to make sure security holds up. Class groups might still have value in this case, since one can easily sample a fresh group by sampling a new discriminant, for example of the form $\Delta = -p$ with p a large random prime $p \equiv 3 \mod 4$. One example of this setting is that of time-lock cryptographic applications, such as verifiable delay functions (VDF), implemented in class groups. We discuss some estimates for appropriate parameter sizes in Section 4.1.4.

3 Experimental Class Group Data

In Section 2, we discussed various algorithms and observations which influence the choice of discriminant for cryptographic applications of class groups. In particular, a number of these observations, such as the average size of the class group or the Cohen-Lenstra heuristics [CL84], provide asymptotic statements about imaginary quadratic class groups $\mathrm{Cl}(\Delta)$, where Δ is a uniformly random fundamental discriminant with $|\Delta| < D$ as $D \to \infty$. For secure implementation of cryptographic applications, such as verifiable delay functions (Section 4.1), it might be beneficial to pick a discriminant of the specific form $\Delta = -p$ where $p \equiv 3 \mod 4$ is a prime. In this section we experimentally test the hypothesis to what extent the statements from Section 2 about the average behaviour of the class group hold up for this specific choice of discriminant. We calculated the class number $h(\Delta)$ and the structure of $\mathrm{Cl}(\Delta)$ as a product of cyclic groups, for all primes $p \equiv 3 \mod 4$ of bit-length 24 through 35.

3.1 Class Group Order

In this section we check whether the asymptotic statements from Section 2.1.3 regarding the order of the class group belonging to a *fundamental* discriminant give good predictions for the order of the class group with discriminant $-p$ where $p \equiv 3 \mod 4$ is a prime.

To obtain an initial overview of the distribution of the class numbers, we constructed histograms of the class numbers and fitted a gamma distribution to them, as it was suggested by Buell [Bue84] that the class numbers seem to obey some sort of gamma distribution. These plots are arranged in Table 3.

Table 3: Histograms of class numbers for fundamental discriminants $-p$, where $p \equiv 3 \mod 4$ is a prime of bit-length $k = 24, \ldots, 35$, fitted with a gamma distribution with shape parameter k, scale parameter θ and location parameter loc.

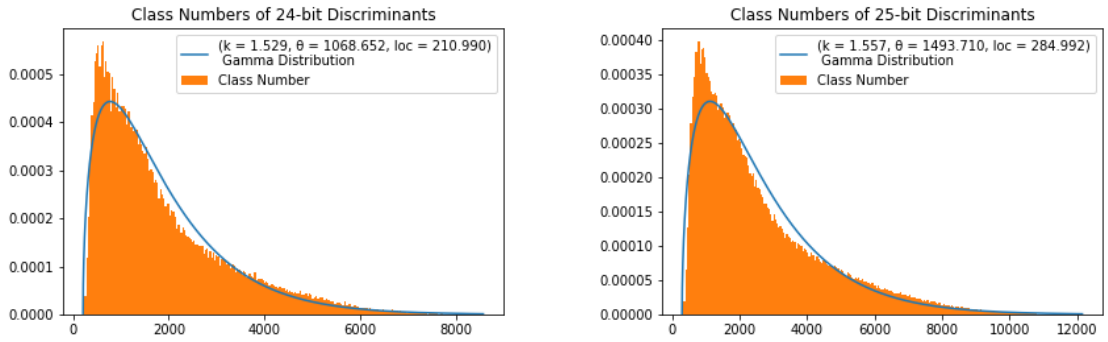


Table 3: Histograms of class numbers for fundamental discriminants $-p$, where $p \equiv 3 \mod 4$ is a prime of bit-length $k = 24, \ldots, 35$, fitted with a gamma distribution with shape parameter k, scale parameter θ and location parameter loc. (Continued.)

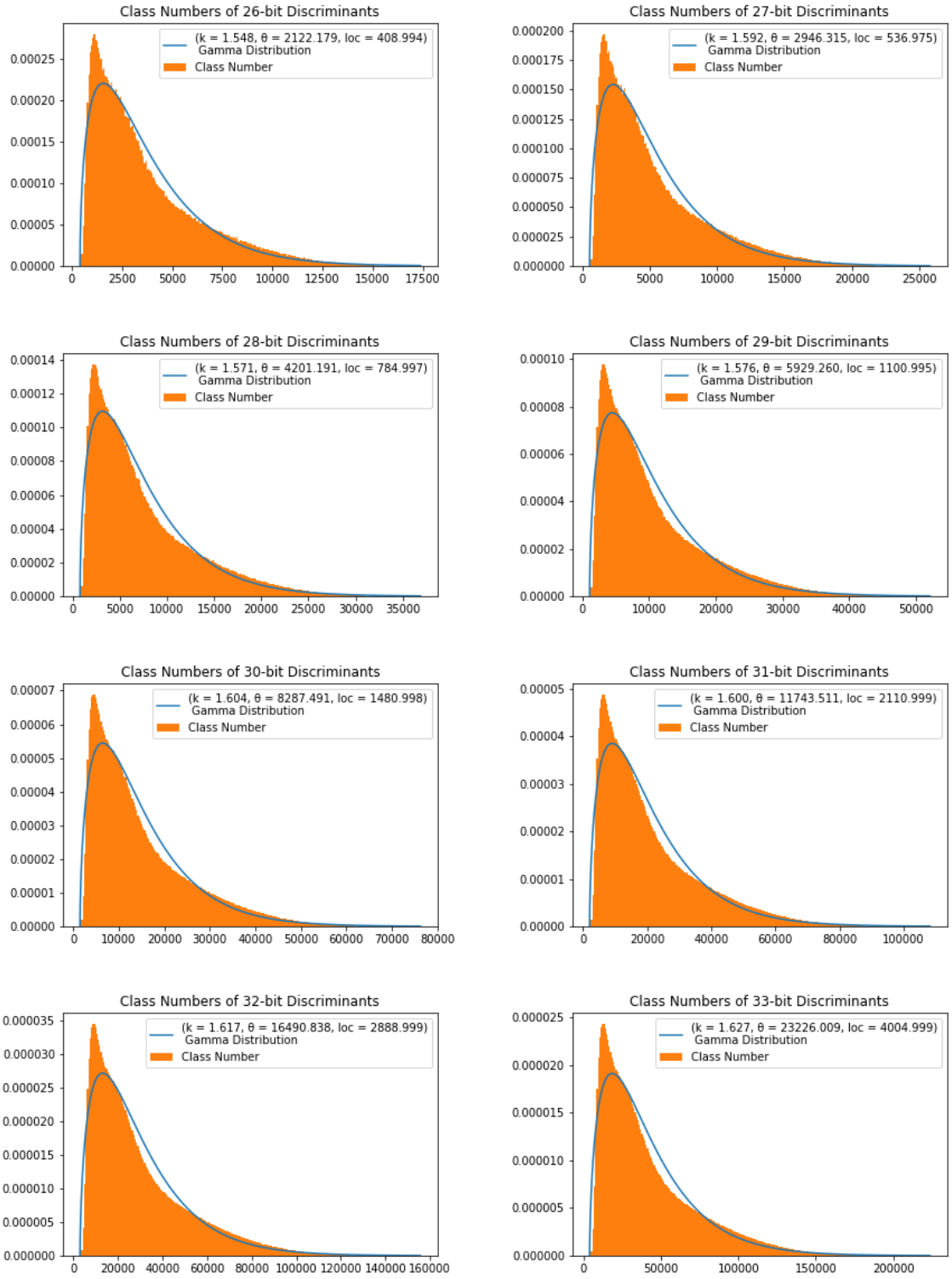
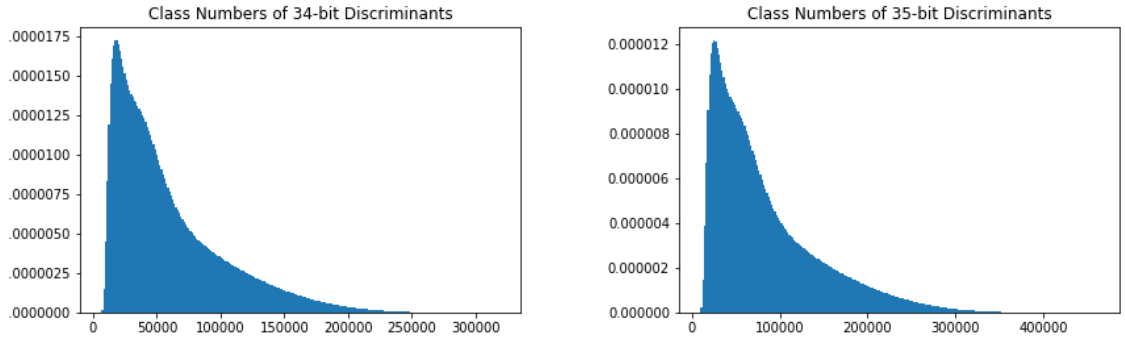


Table 3: Histograms of class numbers for fundamental discriminants $-p$, where $p \equiv 3 \mod 4$ is a prime of bit-length $k = 24, \ldots, 35$, fitted with a gamma distribution with shape parameter k, scale parameter θ and location parameter loc. (Continued.)



Taking a closer look at the peak occuring in the histograms in Table 3, we computed the most often occurring value of the class number $h_{k,ml}$ for the various bitlengths of discriminants, together with its factorisation $(h_{k,ml})_{\text{fact}}$ and its probability $\Pr_{k,ml}$. These can be found in Table 4 for $k = 24, \ldots, 35$. For practical applications where a random group is sampled and where the group order should be hard to compute, an adversary should in particular only be able to guess the order of the group with at most negligible probability. In particular we want $\Pr_{k,ml}$ to be negligible.

Table 4: Most often occurring class numbers, their factorisation and probability, for fundamental discriminants $-p$, where $p \equiv 3 \mod 4$ is a prime of bit-length $k = 24, \ldots, 35$.

	$h_{k,ml}$	$(h_{k,ml})_{\text{fact}}$	$\Pr_{k,ml}$
$k = 24$	585	$3^2 \cdot 5 \cdot 13$	0.00188432
$k = 25$	945	$3^3 \cdot 5 \cdot 7$	0.00144035
$k = 26$	1155	$3 \cdot 5 \cdot 7 \cdot 11$	0.000938484
$k = 27$	1575	$3^2 \cdot 5^2 \cdot 7$	0.000749473
$k = 28$	2205	$3^2 \cdot 5 \cdot 7^2$	0.000487806
$k = 29$	3465	$3^2 \cdot 5 \cdot 7 \cdot 11$	0.000380742
$k = 30$	4095	$3^2 \cdot 5 \cdot 7 \cdot 13$	0.000269540
$k = 31$	5985	$3^2 \cdot 5 \cdot 7 \cdot 19$	0.000189754
$k = 32$	10395	$3^3 \cdot 5 \cdot 7 \cdot 11$	0.000137720
$k = 33$	12285	$3^4 \cdot 5 \cdot 7 \cdot 13$	$9.90772 \cdot 10^{-5}$
$k = 34$	17325	$3^2 \cdot 5^2 \cdot 7 \cdot 11$	$7.17909 \cdot 10^{-5}$
$k = 35$	24255	$3^2 \cdot 5 \cdot 7^2 \cdot 11$	$4.95962 \cdot 10^{-5}$

Let $\overline{h_{k,3(4)}}$ be the average class number over all discriminants $-p$ where $p \equiv 3 \mod 4$ is a k-bit prime. From our discussion in Section 2.1.3, we deduce that asymptotically the average class number over *all fundamental* discriminants Δ of a given bitlength k behaves as

$$\begin{aligned} \overline{h_k} &:= \frac{\sum_{|\Delta| \leq 2^k - 1} h(\Delta) - \sum_{|\Delta| \leq 2^{k-1} - 1} h(\Delta)}{\#\{\Delta : |\Delta| \leq 2^k - 1\} - \#\{\Delta : |\Delta| \leq 2^{k-1} - 1\}} \\ &\sim c_1 \cdot \frac{(2^k - 1)^{3/2} - (2^{k-1} - 1)^{3/2}}{2^k - 2^{k-1}} =: c_1 \cdot S_k, \end{aligned} \tag{13}$$

where $c_1 = C\pi/9 \approx 0.30771495$. We calculated $\overline{h_{k,3(4)}}$ and the ratio $\overline{h_{k,3(4)}}/S_k$ for $k = 24, \ldots, 35$. These values can be found in Table 5. Comparing the computed averages with the asymptotic predictions from equation (13) seems to indicate that the class numbers $h(-p)$ belonging to primes $p \equiv 3 \mod 4$ are on average slightly larger than the average class numbers belonging to *all fundamental* discriminants.

Table 5: Average value of class numbers $h(-p)$ taken over all primes $p \equiv 3 \mod 4$ of $k = 24, \ldots, 35$ bits, and the ratio of this average with S_k from equation (13).

	$\overline{h_{k,3(4)}}$	$\overline{h_{k,3(4)}}/S_k$
$k = 24$	1845.22	0.348438
$k = 25$	2610.52	0.348568
$k = 26$	3693.61	0.348737
$k = 27$	5222.38	0.348658
$k = 28$	7386.82	0.348718
$k = 29$	10445.7	0.348689
$k = 30$	14772.8	0.348699
$k = 31$	20895.1	0.348752
$k = 32$	29550.6	0.348757
$k = 33$	41791.1	0.348759
$k = 34$	59102.7	0.348766
$k = 35$	83588.7	0.348786

Let $h_{k,\min}$ and $h_{k,\max}$ be the minimum and maximum class numbers occurring over all discriminants $-p$, where $p \equiv 3 \mod 4$ is a k-bit prime, respectively. We computed these minima and maxima, together with ratios inspired by the bounds in equation (6), for $k = 24, \ldots, 35$. These values can be found in Table 6. In comparison, the asymptotic statement from equation (6) predicts ratios $1/c_2 \approx 0.146990$ for the minimum and $c_3 \approx 1.13387$ for the maximum.

Table 6: Minimal and maximal occurring class numbers of discriminants $-p$, where $p \equiv 3 \mod 4$ is a prime of bit-length $k = 24, \ldots, 35$, and ratios with the respective upper and lower bounds from equation (6) (ignoring $o(1)$ and constants).

k	$h_{k,\min}$	$h_{k,\max}$	$\frac{h_{k,\min}}{2^{(k-1)/2}/\log\log(2^k)}$	$\frac{h_{k,\max}}{2^{k/2}\log\log(2^k)}$
24	211	8565	0.204825	0.743743
25	285	12153	0.198468	0.735536
26	409	17379	0.204166	0.733667
27	587	25817	0.192022	0.760735
28	785	36865	0.200951	0.758697
29	1101	52105	0.201652	0.749393
30	1481	76327	0.193970	0.767564
31	2111	108175	0.197615	0.760994
32	2889	155729	0.193213	0.766720
33	4005	223705	0.191279	0.771147
34	5391	319185	0.183798	0.770666
35	7929	463679	0.192904	0.784441

3.2 Cohen-Lenstra Heuristics

In this section we compare some of the asymptotic predictions made by the Cohen-Lenstra heuristics [CL84] regarding class groups belonging to fundamental discriminants with the experimental data we collected for class groups of discriminant $-p$ for $p \equiv 3 \mod 4$ a prime of length $k \in \{24, \ldots, 35\}$.

Let $\Pr_{k,\text{cyc}}$ be the probability that $\text{Cl}(-p)$ is cyclic and let $\Pr_{k,i}$ the probability that $\text{Cl}(-p)$ is a product of *exactly* $i \geq 2$ cyclic groups, both when p is picked uniformly at random from all k-bit primes $p \equiv 3 \mod 4$. We calculated these probabilities for $k = 24, \ldots, 35$, which can be found in Table 7. For reference, the Cohen-Lenstra heuristics [CL84] predict that for a randomly picked fundamental discriminant $\Delta < 0$, the probability that $\text{Cl}(\Delta)$ is cyclic asymptotically goes to 0.977575. Our experimentally calculated probability for $k = 35$ already agrees up to two decimal places with this predicted value.

Let $\mho_k(q)$ be the probability that an odd prime q divides the class number $h(-p)$ of a k-bit

Table 7: Probability that the class group is cyclic or the product of 2, 3, 4 groups, respectively, for fundamental discriminants $-p$, where $p \equiv 3 \mod 4$ is a prime of bit-length $k = 24, \ldots, 35$.

	$\Pr_{k,\text{cyc}}$	$\Pr_{k,2}$	$\Pr_{k,3}$	$\Pr_{k,4}$
$k = 24$	0.981414	0.0185396	$4.67188 \cdot 10^{-5}$	0
$k = 25$	0.981789	0.0181910	$2.02866 \cdot 10^{-5}$	0
$k = 26$	0.980974	0.0189829	$4.32822 \cdot 10^{-5}$	0
$k = 27$	0.980523	0.0194325	$4.44416 \cdot 10^{-5}$	0
$k = 28$	0.980175	0.0197806	$4.41131 \cdot 10^{-5}$	0
$k = 29$	0.980023	0.0199273	$4.99890 \cdot 10^{-5}$	0
$k = 30$	0.979728	0.0202228	$4.93749 \cdot 10^{-5}$	0
$k = 31$	0.979444	0.0205013	$5.49930 \cdot 10^{-5}$	$3.94498 \cdot 10^{-8}$
$k = 32$	0.979237	0.0207037	$5.97243 \cdot 10^{-5}$	0
$k = 33$	0.979048	0.0208897	$6.23107 \cdot 10^{-5}$	$2.10154 \cdot 10^{-8}$
$k = 34$	0.978884	0.0210516	$6.42961 \cdot 10^{-5}$	$2.16613 \cdot 10^{-8}$
$k = 35$	0.978730	0.0212023	$6.75587 \cdot 10^{-5}$	$1.67300 \cdot 10^{-8}$

prime number $p \equiv 3 \mod 4$. The values of $\mho_k(q)$ for $k = 24, \ldots, 35$ and $q = 3, 5, 7, 11$ can be found in Table 8. We recall that the Cohen-Lenstra [CL84] heuristics predict that asymptotically the probability $\mho(q)$ that an odd prime q divides $h(\Delta)$ for a fundamental discriminant $\Delta < 0$, goes to

$$\mho(3) \approx 0.439874, \quad \mho(5) \approx 0.239667, \quad \mho(7) \approx 0.163205, \quad \mho(11) \approx 0.0991673,$$

which agree up to at least two decimal places with our experimentally computed values for $k = 35$.

Table 8: Probability that an odd prime $q \in \{3, 5, 7, 11\}$ divides the class number $h(-p)$ for odd primes $p \equiv 3 \mod 4$ of bit-length $k = 24, \cdots 35$.

k	$\mho_k(3)$	$\mho_k(5)$	$\mho_k(7)$	$\mho_k(11)$
24	0.425818	0.238219	0.162223	0.0990789
25	0.426901	0.237037	0.163510	0.0982562
26	0.428418	0.238500	0.162313	0.0997971
27	0.430204	0.238980	0.163087	0.0993342
28	0.430772	0.238603	0.162864	0.0992825
29	0.432068	0.238901	0.163004	0.0991006
30	0.432483	0.239063	0.163041	0.0990699
31	0.433422	0.239048	0.163105	0.0991497
32	0.434190	0.239209	0.163151	0.0990407
33	0.434744	0.239363	0.163034	0.0991211
34	0.435351	0.239317	0.163127	0.0991194
35	0.435822	0.239489	0.163167	0.0991594

Next we take a more in depth look at the q-rank of the class group. Let $\Pr_{k,q,r}$ be the probability that the q-rank of $\mathrm{Cl}(-p)$ is equal to r for some prime q and where again $p \equiv 3 \mod 4$ is a k-bit prime. The values of $\Pr_{k,q,r}$ for $k = 24, \ldots, 35$, $q = 3, 5, 7, 11$ and $r = 1, 2, 3, 4$ (the probability is zero for all higher values of r in these cases) can be found in Table 9. Comparing these with the asymptotically predicted probabilities over all fundamental discriminants from the Cohen-Lenstra heuristics from Table 12, we see that the predicted values agree up to at least two decimal places with our experimentally computed values for $k = 35$ and rank $r \in \{1, 2\}$. For higher q-rank r, the number of class groups calculated during our experiments is too small to make any meaningful comparisons.

Table 9: Probability that the q-rank of the class group $\mathrm{Cl}(-p)$ is equal to 1, 2, 3 or 4 for primes $p \equiv 3 \mod 4$ of bit-length $k = 24, \ldots, 35$ and $q \in \{3, 5, 7, 11\}$.

k	q	$\Pr_{k,q,1}$	$\Pr_{k,q,2}$	$\Pr_{k,q,3}$	$\Pr_{k,q,4}$
24	3	0.409844	0.0159272	$4.67188 \cdot 10^{-5}$	0
	5	0.236222	0.00199723	0	0
	7	0.161709	0.000513907	0	0
	11	0.0989776	0.000101224	0	0
25	3	0.411096	0.0157850	$2.02866 \cdot 10^{-5}$	0
	5	0.235213	0.00182377	0	0
	7	0.163029	0.000480793	0	0
	11	0.0981588	$9.73757 \cdot 10^{-5}$	0	0
26	3	0.411934	0.0164409	$4.22265 \cdot 10^{-5}$	0
	5	0.236560	0.00193820	$1.05566 \cdot 10^{-6}$	0
	7	0.161816	0.000497217	0	0
	11	0.0997285	$6.86181 \cdot 10^{-5}$	0	0
27	3	0.413293	0.0168670	$4.38930 \cdot 10^{-5}$	0
	5	0.237025	0.00195379	$5.48662 \cdot 10^{-7}$	0
	7	0.162568	0.000519583	0	0
	11	0.0992629	$7.13261 \cdot 10^{-5}$	0	0
28	3	0.413506	0.0172238	$4.26902 \cdot 10^{-5}$	0
	5	0.236627	0.00197570	$1.42301 \cdot 10^{-6}$	0
	7	0.162375	0.000488945	0	0
	11	0.0992071	$7.54193 \cdot 10^{-5}$	0	0
29	3	0.414633	0.0173853	$4.98415 \cdot 10^{-5}$	0
	5	0.236935	0.00196579	$1.47460 \cdot 10^{-7}$	0
	7	0.162530	0.000473347	0	0
	11	0.0990262	$7.43199 \cdot 10^{-5}$	0	0
30	3	0.414804	0.0176297	$4.89170 \cdot 10^{-5}$	0
	5	0.237052	0.00201010	$4.57882 \cdot 10^{-7}$	0
	7	0.162547	0.000493291	0	0
	11	0.0989983	$7.15822 \cdot 10^{-5}$	0	0
31	3	0.415441	0.0179265	$5.45196 \cdot 10^{-5}$	$3.94498 \cdot 10^{-8}$
	5	0.237051	0.00199636	$3.94498 \cdot 10^{-7}$	0
	7	0.162620	0.000485824	$7.88996 \cdot 10^{-8}$	0
	11	0.0990746	$7.51124 \cdot 10^{-5}$	0	0
32	3	0.416048	0.0180824	$5.90928 \cdot 10^{-5}$	0
	5	0.237177	0.00203146	$6.11095 \cdot 10^{-7}$	0
	7	0.162655	0.000496311	$2.03698 \cdot 10^{-8}$	0
	11	0.0989642	$7.65090 \cdot 10^{-5}$	0	0
33	3	0.416408	0.0182739	$6.17433 \cdot 10^{-5}$	$2.10154 \cdot 10^{-8}$
	5	0.237329	0.00203293	$5.35893 \cdot 10^{-7}$	0
	7	0.162543	0.000491246	$3.15231 \cdot 10^{-8}$	0
	11	0.0990453	$7.58236 \cdot 10^{-5}$	0	0
34	3	0.416853	0.0184340	$6.37167 \cdot 10^{-5}$	$2.16613 \cdot 10^{-8}$
	5	0.237280	0.00203636	$5.46947 \cdot 10^{-7}$	0
	7	0.162638	0.000489881	$3.24919 \cdot 10^{-8}$	0
	11	0.0990444	$7.49426 \cdot 10^{-5}$	0	0
35	3	0.417182	0.0185729	$6.68727 \cdot 10^{-5}$	$1.67300 \cdot 10^{-8}$
	5	0.237443	0.00204550	$6.55260 \cdot 10^{-7}$	0
	7	0.162675	0.000492055	$3.06717 \cdot 10^{-8}$	0
	11	0.0990840	$7.53604 \cdot 10^{-5}$	0	0

These q-rank probabilities also let us compute the average number of elements of order q in $\mathrm{Cl}(-p)$ for $q \in \{3, 5, 7, 11\}$ taken over all primes $p \equiv 3 \mod 4$ of bit-length k. We denote these averages as $\overline{\#\mathrm{Cl}_k[q]}$, and display them in Table 10. The Cohen-Lenstra heuristics [CL84] predict that when the average is taken over all fundamental discriminants these averages all approach 2 asymptotically, which has been proven unconditionally in the case $q = 3$ (see Section 1.7).

Table 10: Average number of q-torsion elements $\overline{\#\mathrm{Cl}_k[q]}$ for $q \in \{3, 5, 7, 11\}$ in the class group $\mathrm{Cl}(-p)$ over odd primes $p \equiv 3 \mod 4$ of bit-length $k = 24, \cdots 35$.

k	$\overline{\#\mathrm{Cl}_k[3]}$	$\overline{\#\mathrm{Cl}_k[5]}$	$\overline{\#\mathrm{Cl}_k[7]}$	$\overline{\#\mathrm{Cl}_k[11]}$
24	1.85138	1.95290	1.97336	1.99081
25	1.85369	1.94816	1.98107	1.98256
26	1.85665	1.95399	1.97389	1.99797
27	1.86030	1.95592	1.97852	1.99334
28	1.86147	1.95441	1.97718	1.99283
29	1.86403	1.95560	1.97802	1.99101
30	1.86473	1.95625	1.97824	1.99070
31	1.86684	1.95619	1.97863	1.99150
32	1.86803	1.95684	1.97891	1.99041
33	1.86948	1.95745	1.97820	1.99121
34	1.87032	1.95727	1.97876	1.99119
35	1.87125	1.95795	1.97900	1.99159

3.3 Most Likely Largest Prime Divisor

Finally, in this section we calculate the most likely largest prime divisor of the group order. Damgård and Koprowski propose as part of their definition of a *hard (cyclic) group family* that it should be hard to guess the largest prime divisor of the order of a group sampled from such a family [DK02, Definition 1], which we generalize to the setting of abelian groups [vBS21, Definition 4.4].

Let q_{lml} be the most often occurring largest prime divisor of the class number. That is, the prime that is the most likely to be the largest prime divisor of the class number $h(-p)$ for $p \equiv 3 \mod 4$ a prime of bit-length k. Furthermore, let $\Pr_{q_{lml}}$ be the corresponding probability that q_{lml} occurs as the largest prime in the prime factorisation of the class number. The values of q_{lml} and $\Pr_{q_{lml}}$ for $k = 24, \ldots, 35$ can be found in Table 11. For the group family $\mathcal{G}_\kappa := \{\mathrm{Cl}(-p) : p \equiv 3 \mod 4 \text{ a } k(\kappa)\text{-bit prime}\}$ to be hard, we want the probability $\Pr_{q_{lml}}$ to be negligible in the security parameter κ. To the author's knowledge, there do not exists any asymptotic statements about the probability $\Pr_{q_{lml}}$ for imaginary quadratic class groups belonging to fundamental discriminants.

Table 11: Most likely occurring largest prime divisor q_{lml} of the class numbers $h(-p)$ of primes $p \equiv 3 \mod 4$ of bit-length $k = 24, \ldots, 35$, together with the probability $\Pr_{q_{lml}}$ of this prime occurring as the largest prime in the prime factorization of the class number.

k	q_{lml}	$\Pr_{q_{lml}}$
24	19	0.0284478
25	19	0.0253745
26	19	0.0219008
27	19	0.0194523
28	23	0.0168287
29	23	0.0146186
30	31	0.0126955
31	31	0.0111087
32	31	0.00968324
33	43	0.00859419
34	47	0.00760201
35	47	0.00669802

4 Applications

4.1 Verifiable Delay Functions

In this section we will discuss the assumptions underlying two constructions of verifiable delay functions (VDFs), and explore how they hold up when implemented in class groups of imaginary quadratic number fields.

We will discuss two constructions of VDFs in this section, namely Wesolowski's [Wes19] and Pietrzak's [Pie19]. Both these constructions are based on the time lock puzzle of Rivest, Shamir and Wagner [RSW96]. It essentially comes down to computing $Y = X^{2^t}$ for a randomly sampled element X from a finite abelian group $\mathbb{G}$ and a timing parameter t, together with a proof π such that any verifier can efficiently check that the computation was done correctly. The main difference between the two constructions is in how the proof π is produced. Wesolowski's construction requires $O(t/\log t)$ group multiplications to compute a proof consisting of a single group element, whereas Pietrzak's construction requires only $O(\sqrt{t} \cdot \log t)$ group multiplications but produces a proof consisting of $O(\log t)$ group elements. Moreover, verification of the proof takes two group exponentiations in Wesolowski's construction and $2 \cdot \log t$ group exponentiations in Pietrzak's.

4.1.1 Wesolowski's and Pietrzak's Construction

Both constructions can in fact be seen as *trapdoor* verifiable delay functions, which means that a party holding some secret key sk can efficiently produce the output of the VDF. Concretely, both VDF constructions can be described as in Definition 4.1. We describe the proofs generated during evaluation which allow for fast verification in Section 4.1.3. We consider a targeted evaluation time given by $T = t\delta$ for a timing parameter T, where δ is the time-cost (i.e. the amount of sequential work) of computing a single squaring in the group $\mathbb{G}$. Here T, t and δ are public parameters to the protocol and may depend on the security parameter $\kappa \in \mathbb{N}$. We denote $\mathscr{A}^*$ for the set of all arbitrary finite length strings over some alphabet $\mathscr{A}$.

Definition 4.1 ([Wes19, Pie19])**.** The construction $\mathsf{VDF} := (\mathsf{keygen}, \mathsf{trapdoor}, \mathsf{eval}, \mathsf{verify})$ is given by the following algorithms. Note that each algorithm receives 1^κ as input (which we will leave implicit for all but the keygen algorithm), where $\kappa \in \mathbb{N}$ is the security parameter, and each algorithm is required to run in time polynomial in κ.

- $\mathsf{keygen}(1^\kappa)$: The algorithm outputs a finite abelian group $\mathbb{G}$ and an efficiently computable hash function $H_\mathbb{G} : \mathscr{A}^* \to \mathbb{G}$ according to the security parameter κ. The public key is $\mathsf{pk} = (\mathbb{G}, H_\mathbb{G})$ and the secret key is $\mathsf{sk} = |\mathbb{G}|$.
- $\mathsf{trapdoor}_{\mathsf{sk}}(x, T)$: The algorithm receives a public key $\mathsf{pk} = (\mathbb{G}, H_\mathbb{G})$, a secret key $\mathsf{sk} = |\mathbb{G}|$, some input $x \in \mathscr{A}^*$, and a targeted evaluation time $T = t\delta$. It computes $X = H_\mathbb{G}(x)$, $e = 2^t \mod |\mathbb{G}|$, and computes $Y = X^e$ (in $O(\log |\mathbb{G}|)$ sequential group multiplications). It moreover computes some proof π (which we describe in Section 4.1.3) and outputs (Y, π).
- $\mathsf{eval}_{\mathsf{pk}}(x, T)$: The algorithm receives a public key $\mathsf{pk} = (\mathbb{G}, H_\mathbb{G})$, some input $x \in \mathscr{A}^*$, and a targeted evaluation time $T = t\delta$. It computes $X = H_\mathbb{G}(x)$ and $Y = X^{2^t}$ (through t sequential squarings). It moreover computes some proof π (which we describe later) and outputs (Y, π).
- $\mathsf{verify}_{\mathsf{pk}}(x, Y, \pi, T)$: The algorithm receives a public key $\mathsf{pk} = (\mathbb{G}, H_\mathbb{G})$, some input $x \in \mathscr{A}^*$, a targeted evaluation time $T = t\delta$, a VDF output Y and a proof π. It outputs true if the proof is "correct" (again, see section 4.1.3 for details) and outputs false otherwise.

Note that in the context where a group can be generated without revealing its order, as is the case for imaginary quadratic class groups $\mathrm{Cl}(\Delta)$ by sampling a random fundamental discriminant Δ, it makes sense to omit the secret key sk and $\mathsf{trapdoor}$ algorithm from the syntax. In this case the trapdoor $|\mathrm{Cl}(\Delta)|$ still exists, but is not known to be efficiently computable classically given the discriminant Δ (see Section 2). There do exist efficient quantum algorithms to compute $|\mathrm{Cl}(\Delta)|$ on input Δ (see Section 2.2), and so it would be possible to instantiate VDF as a trapdoor VDF in a post-quantum setting (see Section 4.1.4). If there does exist a way to efficiently compute the

trapdoor during setup, then the setup phase will need to be executed by a party who can be trusted not to reveal the trapdoor.

Given any key pair $(\mathsf{pk}, \mathsf{sk})$ generated by keygen, the scheme is *complete* if the following holds. Given any input $x \in \mathscr{A}^*$ and time parameter T, let $(Y, \pi) \leftarrow \mathsf{eval}_{\mathsf{pk}}(x, T)$ and $(Y', \pi') \leftarrow \mathsf{trapdoor}_{\mathsf{sk}}(x, T)$. Then $Y = Y'$ and the procedures $\mathsf{verify}_{\mathsf{pk}}(x, Y, \pi, T)$ and $\mathsf{verify}_{\mathsf{pk}}(x, Y', \pi', T)$ both output true. We also require the protocol to be *sound*, which is defined as follows.

Definition 4.2 (Soundness)**.** Let $T : \mathbb{N} \to \mathbb{R}_{>0}$ be a function of the security parameter κ. A (trapdoor) verifiable delay function VDF is sound if for any PPT algorithm $\mathcal{A}$ there exists a negligible function $\mathsf{negl} : \mathbb{N} \to \mathbb{R}_{\geq 0}$ such that for all $\kappa \in \mathbb{N}$ and $\mathsf{pk} \leftarrow \mathsf{keygen}(1^\kappa)$:

$$\Pr[Y' \neq \mathsf{eval}_{\mathsf{pk}}(x, T) \wedge \mathsf{verify}_{\mathsf{pk}}(x, Y', \pi', T) = \mathsf{true} \,:\, (x, Y', \pi') \leftarrow \mathcal{A}(\mathsf{pk}, 1^\kappa)] \leq \mathsf{negl}(\kappa).$$

The second security property is that the correct output can not be produced in time less than T without knowledge of the secret key sk. This is formalized via the T-evaluation race game, defined as follows. In the following definitions, we implicitly assume that the adversary works in some model of computation $\mathscr{M}$ and assume that it allows all operations a potential adversary could perform. We assume this model comes with a computational cost function C and a time-cost function T. For any algorithm $\mathcal{A}$ and input x, the cost $C(\mathcal{A}, x)$ measures the overall computational cost of computing $\mathcal{A}(x)$, while the time-cost $T(\mathcal{A}, x)$ abstracts the notion of time it takes to run $\mathcal{A}(x)$ in the model $\mathscr{M}$ (i.e. the cost of all the sequential operations computed by $\mathcal{A}$ on input x). See [Wes19, Section 3.1 and 3.2] for more details.

Definition 4.3 (T-evaluation race game)**.** Let $\mathcal{A}$ be a party playing the game. The parameter $T : \mathbb{N} \to \mathbb{R}_{>0}$ is a function of the security parameter κ. The T-evaluation race game $\mathsf{Exp}^{T\text{-eval}}_{\mathcal{A},\mathsf{VDF}}$ is defined as follows, where $\mathcal{D}_\kappa$ is a distribution over a finite subset of $\mathscr{A}^*$ of min-entropy at least κ.

$\underline{\mathsf{Exp}^{T\text{-eval}}_{\mathcal{A},\mathsf{VDF}}(\kappa) :}$
$\mathsf{pk} \leftarrow \mathsf{keygen}(1^\kappa)$
$\mathcal{B} \leftarrow \mathcal{A}(\mathsf{pk}, 1^\kappa)$
$x \xleftarrow{\$} \mathcal{D}_\kappa$
$Y \leftarrow \mathcal{B}^{\mathcal{O}(\cdot)}(x)$
`if` $T(\mathcal{B}, x) < T \wedge Y = \mathsf{eval}_{\mathsf{pk}}(x, T)$ `return` 1
`else return` 0

$\underline{\mathcal{O}(x') :}$
`if` $x' \neq x$ `return`
$\mathsf{trapdoor}_{\mathsf{sk}}(x', T)$
`else return` $\perp$

Definition 4.4 (T-sequential)**.** Let $T : \mathbb{N} \to \mathbb{R}_{>0}$ be a function of the security parameter κ. A trapdoor verifiable delay function VDF is T-sequential if for any PPT adversary $\mathcal{A}$ there exists a negligible function $\mathsf{negl} : \mathbb{N} \to \mathbb{R}_{\geq 0}$ such that for all $\kappa \in \mathbb{N}$:

$$\Pr[\mathsf{Exp}^{T\text{-eval}}_{\mathcal{A},\mathsf{VDF}}(\kappa) = 1] \leq \mathsf{negl}(\kappa).$$

For a non-trapdoor VDF one can analogously define T-sequentiality by omitting the oracle $\mathcal{O}(\cdot)$ from the T-evaluation race game.

4.1.2 Sequentiality

The assumption underlying the sequentiality of the VDF is the same for both VDF's and is basically a generalization of the time-lock assumption from Rivest, Shamir and Wagner [RSW96].

Definition 4.5. Consider an abelian group family $\mathcal{G} = (\mathcal{G}_\kappa)_{\kappa=1}^\infty$ with group sampling algorithm GGen as in definition 1.22. Fix a security parameter $\kappa \in \mathbb{N}$, and let $\mathcal{A}$ be an algorithm playing the game. The parameter t is a positive integer, and $\delta : \mathbb{N} \to \mathbb{R}_{>0}$ is a function. The (δ, t)-*time-lock game* (δ, t)-RSW is defined as follows:

(δ, t)-$\mathsf{RSW}^{\mathcal{A}}_{\mathcal{G}}(\kappa)$:
$\mathbb{G} \leftarrow \mathsf{GGen}(1^\kappa)$
$\mathcal{B} \leftarrow \mathcal{A}(1^\kappa, \mathbb{G})$
$X \xleftarrow{\$} \mathbb{G}$
$Y \leftarrow \mathcal{B}(X, \mathbb{G})$
`if` $T(\mathcal{B}, X) < t \cdot \delta(\kappa) \wedge Y = X^{2^t}$ `return` 1
`else return` 0

Definition 4.6. The *generalized time-lock assumption* for group family $(\mathcal{G}_\kappa)_{\kappa=1}^{\infty}$ is defined as follows. There exists a cost function $\delta : \mathbb{N} \to \mathbb{R}_{>0}$ such that the following two statements hold:

(1) There is an algorithm $\mathcal{S}$ such that for any group $\mathbb{G} \in \mathcal{G}_\kappa$, and any element $X \in \mathbb{G}$, the output of $S(\mathbb{G}, X)$ is the square of X, and $T(\mathcal{S}, (\mathbb{G}, X)) < \delta(\kappa)$.

(2) For any $t \in \mathbb{N}$ and any PPT algorithm $\mathcal{A}$ there exists a negligible function $\mathsf{negl} : \mathbb{N} \to \mathbb{R}_{\geq 0}$ such that for all $\kappa \in \mathbb{N}$:
$$\Pr[(\delta, t)\text{-}\mathsf{RSW}^{\mathcal{A}}_{\mathcal{G}}(\kappa) = 1] \leq \mathsf{negl}(\kappa).$$

This assumption basically says that: (1) there exists a function $\delta : \mathbb{N} \to \mathbb{R}_{>0}$ such that $\delta(\kappa)$ encodes the time-cost of computing a single squaring in any group of $\mathcal{G}_k$; (2) without more specific knowledge about these groups (such as their orders), there is no faster way to compute X^{2^t} than performing t sequential squarings. Wesolowski proved that, in the random oracle model, his VDF construction is $(t\delta)$-sequential for any $t \in \mathbb{N}$ if the generalized time-lock assumption holds with respect to the group family output by keygen and cost function $\delta : \mathbb{N} \to \mathbb{R}_{>0}$ [Wes19, Proposition 1].

Performing a single squaring in an imaginary quadratic class group $\mathrm{Cl}(\Delta)$ of a fundamental discriminant $\Delta < 0$ requires $O(\log^2 |\Delta|)$ bit operations using basic algorithms [Ham02, Section 7.1.4]. The asymptotically fastest algorithm is Schönhage's algorithm [Sch91], which takes $O(\mu(\log |\Delta|) \cdot \log \log |\Delta|)$ bit operations, where $\mu(n)$ is the complexity of multiplying two n-bit integers. Jacobson, Sawilla and Williams [JSW06a] argue that in practice Shanks and Atkins' NUCOMP algorithm (in the fomulation from [JSW06b]) is more efficient than Schönhage's algorithm. NUCOMP has an asymptotic complexity of $O(\mu(\log |\Delta|) \cdot \log |\Delta|)$. The version of NUCOMP specifically taylored to squaring is called NUDUPL. For the original formulation of the NUCOMP and NUDUPL algorithms, which are due to Shanks and Atkins, see [Coh93, Algorithm 5.4.8] and [Coh93, Algorithm 5.4.9], respectively. Plugging in an $O(n^2)$ integer multiplication algorithm, which is often fastest in practice, and has depth $O(n)$, this leads to an $O(\log^2 |\Delta|)$ depth for class group multiplications.

4.1.3 Soundness

To analyze the soundness of the schemes, we will now describe the proof generation protocol of both constructions. We describe the interactive version of both protocols. The protocols can be made non-interactive, while retaining soundness, by applying the Fiat-Shamir transform [FS86] to the interactive protocol. The non-interactive protocol is used to generate the proof as part of the eval algorithm from Section 4.1.1. In applications where VDF instances are used in interactive protocols with a limited time span, the interactive version of the proof generation is more useful. This especially makes sense in a post-quantum setting, as there exist polynomial time quantum algorithms breaking the hardness assumption underlying the soundness. See Section 4.1.4 for a discussion of the post-quantum security of the VDF construction.

Wesolowski's Protocol

Let $\mathrm{Primes}(2\kappa)$ be the set of the first $2^{2\kappa}$ primes. We stress that taking $\mathrm{Primes}(2\kappa)$ is essential since there is an $\tilde{\mathcal{O}}(2^{\kappa/2})$ attack on the soundness if one were to take $\mathrm{Primes}(\kappa)$ [BBF18, Section 3.3].

The protocol runs between a prover P and a verifier V, where both parties have input $(\mathbb{G}, X, Y, t)$, and P wants to convince V that $Y = X^{2^t}$ in $\mathbb{G}$. The protocol goes as follows.

- V samples $\ell \xleftarrow{\$} \text{Primes}(2\kappa)$ and sends ℓ to P.
- P computes $\pi := X^{\lfloor 2^t/\ell \rfloor}$ and sends π to V.
- V computes $r := 2^t \mod \ell$, and outputs true if $X, Y, \pi \in \mathbb{G}$ and $\pi^\ell X^r = Y$, and false otherwise.

A trapdoor prover P_{sk} holding the secret key $|\mathbb{G}|$ can produce a valid proof by computing $r := 2^t \mod \ell$, $q := (2^t - r) \cdot \ell^{-1} \mod |\mathbb{G}|$ and subsequently putting $\pi := X^q$. Since ℓ is sampled from a superpolynomially large set of primes, the probability that $\gcd(\ell, |\mathbb{G}|) > 1$ is negligible.

Note that if a malicious prover $\widetilde{\mathsf{P}}$ can compute an ℓ-th root V of an element $U \in \mathbb{G}$, $U \neq 1_{\mathbb{G}}$ of their choice. Then, given $Y = X^{2^t}$ and a valid proof π for $(\mathbb{G}, X, Y, t)$, $\widetilde{\mathsf{P}}$ can produce a valid proof for $(\mathbb{G}, X, \widetilde{Y} := U \cdot Y, t)$ by computing $\tilde{\pi} := V \cdot \pi$. Then indeed

$$\tilde{\pi}^\ell X^r = V^\ell \pi^\ell X^r = U \cdot Y = \widetilde{Y}.$$

On the other hand, $X^{2^t} = Y \neq U \cdot Y$; hence it is necessary that computing ℓ-th roots is difficult for the protocol to be sound. This is formalized in the adaptive root problem ARoot (see Table 2).

Wesolowski showed that the non-interactive version of the protocol is sound in the random oracle model if the *adaptive root problem* ARoot is hard [Wes19, Proposition 2]. That is, if for any PPT algorithm $\mathcal{A}$ there exists a negligible function $\mathsf{negl} : \mathbb{N} \to \mathbb{R}_{\geq 0}$ such that for all $\kappa \in \mathbb{N}$:

$$\Pr[\mathsf{ARoot}^{\mathcal{A}}_{\mathsf{keygen}}(\kappa) = 1] \leq \mathsf{negl}(\kappa),$$

where we recall that the adaptive root problem ARoot with respect to the group family output by keygen is defined as follows.

$\mathsf{ARoot}^{\mathcal{A}}_{\mathsf{keygen}}(\kappa)$:
$\mathbb{G} \leftarrow \mathsf{keygen}(1^\kappa)$
$X \leftarrow \mathcal{A}(1^\kappa, \mathbb{G})$
$\ell \xleftarrow{\$} \text{Primes}(2\kappa)$
$Y \leftarrow \mathcal{A}(X, \ell)$
`if` $X \neq 1_{\mathbb{G}} \wedge Y^\ell = X$ `return` 1
`else return` 0

As we mentioned in Section 1.6, it is easy to solve ARoot if a multiple of the order of $\mathbb{G}$ is known, that is, $\mathsf{ARoot} \Rightarrow \mathsf{MO}$. Hence it is a necessary condition that the order of $\mathbb{G}$ is hard to compute. That is, the VDF needs to be implemented in a hidden-order group, such as an imaginary quadratic class group $\text{Cl}(\Delta)$. Additionally, since the adversary can freely choose the element of which they want to try to find a root in the adaptive root game, it needs to be intractable for an adversary to find elements of low order in $\mathbb{G}$. For example, if one can produce an element of order two in $\mathbb{G}$, it is trivial to find odd roots of this element, since the element itself is. We will see in the next section that this is closely related to the assumption underlying the soundness of Pietrzak's protocol.

In an imaginary quadratic class group $\text{Cl}(\Delta)$, as mentioned in Section 1.6, there is no known more efficient method to compute e-th roots, when $\gcd(e, |\text{Cl}(\Delta)|) = 1$, than to compute a multiple of the group order. As we have seen in Section 2, the best algorithm for computing the order of $\text{Cl}(\Delta)$ has a conjectured expected running time of $L_{|\Delta|}[\frac{1}{2}, 1]$. For a 798-bit discriminant this gives a conjectured expected running time comparable to that of factoring a 1024-bit RSA modulus with the general number field sieve [BJS10]

Pietrzak's Protocol

For simplicity we assume that the timing parameter t is a power of two. See [Pie19] for a more general versions of the protocol.

The protocol runs between a prover P and a verifier V, where both parties have input $(\mathbb{G}, X, Y, t)$, and P wants to convince V that $Y = X^{2^t}$. The protocol goes as follows.

Set $(X_1, Y_1) := (X, Y)$.

- For $i = 1, \ldots, t-1$:
  - P sends $\mu_i := X_i^{2^{t/2^i}}$ to V.
  - V checks if $\mu_i, X_i, Y_i \in \mathbb{G}$. If not, outputs false. Otherwise, V samples $r_i \xleftarrow{\$} \mathbb{Z}_{2^\kappa}$ and sends r_i to P.
  - P and V compute $X_{i+1} := X_i^{r_i}\mu_i$ and $Y_{i+1} := \mu_i^{r_i} Y_i$, and output $(\mathbb{G}, X_{i+1}, Y_{i+1}, t/2^i)$.
- V outputs true if $Y_t = X_t^2$ and false otherwise.

Note that if a malicious prover $\widetilde{\mathsf{P}}$ can produce an element $U \in \mathbb{G}$ of order $d > 1$. Then given $Y = X^{2^t}$ and a valid proof $\pi = \{\mu_i\}_{i=1}^{t-1}$ for (X, Y), $\widetilde{\mathsf{P}}$ can produce a proof for $(\mathbb{G}, X, \widetilde{Y} := U \cdot Y, t)$ that will be accepted by the verifier with probability $1/d$ as follows. In the first step of the proof, the adversary replaces μ_1 by $\tilde{\mu}_1 := U \cdot X^{2^{t/2}} = U \cdot \mu_1$ and computes the rest of the proof as usual. For this proof to be accepted by the verifier, we require that

$$U^{r_1+1} X^{r_1 \cdot 2^{t/2}} Y = \tilde{\mu}_1^{r_1} \widetilde{Y}_1 = \widetilde{Y}_2 \stackrel{?}{=} X_2^{2^{t/2}} = (X_1^{r_1}\tilde{\mu}_1)^{2^{t/2}} = U^{2^{t/2}} X^{r_1 \cdot 2^{t/2}} Y,$$

which happens if and only if $r_1 + 1 \equiv 2^{t/2} \mod d$. Since r_1 is sampled uniformly from $\mathbb{Z}_{2^\kappa}$, this happens with probability $\approx 1/d$, which is non-negligible when $d \ll 2^\kappa$, i.e. when u is an element of *low order*. This is formalizing by requiring the hardness of the low order problem LO (see Table 2).

Pietrzak showed that the above protocol is sound when implemented in the subgroup of quadratic residues in an RSA group (where the modulus is the product of two safe primes) [Pie19, Theorem 1]. This result was generalized to arbitrary finite abelian groups of unknown order by Boneh, Bünz and Fisch, who showed that the protocol is sound if the *low order problem* LO is hard [BBF18, Theorem 1]. That is, if for any PPT algorithm $\mathcal{A}$ there exists a negligible function $\mathsf{negl} : \mathbb{N} \to \mathbb{R}_{\geq 0}$ such that for all $\kappa \in \mathbb{N}$:

$$\Pr[\mathsf{LO}^{\mathcal{A}}_{\mathsf{keygen}}(\kappa) = 1],$$

where we recall that the low order problem LO with respect to the group family output by keygen is defined as follows.

$\mathsf{LO}^{\mathcal{A}}_{\mathsf{keygen}}(\kappa)$:
$\mathbb{G} \leftarrow \mathsf{keygen}(1^\kappa)$
$(X, d) \leftarrow \mathcal{A}(1^\kappa, \mathbb{G})$
`if` $X \neq 1_{\mathbb{G}} \wedge 1 < d \ll 2^\kappa \wedge X^d = 1_{\mathbb{G}}$ `return` 1
`else return` 0

It is clear that $\mathsf{LO} \Rightarrow \mathsf{MO}$ provided that an adversary can *find* an element of low order in the group $\mathbb{G}$. In [vBS21, Proposition 6.2] we show that this is the case if there exists an oracle which outputs a prime $p \ll 2^\kappa$ that divides $|\mathbb{G}|$ with non-negligible probability. Heuristically this is the case for imaginary quadratic class groups by the Cohen-Lenstra heuristics [CL84], as we will discuss below. On the other hand, if there is a prime p of size $|p| \approx 2^\kappa$ dividing $|\mathbb{G}|$, then by Lemma 1.25 the probability that p does not divide the order of a random element is $\leq 2^{-\kappa}$, which is negligible. Hence, the probability that an adversary randomly samples an element of low order is negligible.

Boneh, Bünz and Fisch showed that $\mathsf{ARoot} \Rightarrow \mathsf{LO}$ [BBF18]. That is, if we can find an element $1_{\mathbb{G}} \neq U \in \mathbb{G}$ and an integer $1 < d \ll 2^\kappa$ with $U^d = 1_{\mathbb{G}}$. Then for a random prime ℓ sampled uniformly from Primes(2κ), we can compute an ℓ-th root of U as $U^{\ell^{-1} \mod d}$ as long as $\gcd(d, \ell) = 1$, which happens with all but negligible probability. In particular this implies that soundness of Wesolowski's protocol implies soundness of Pietrzak's protocol.

One way of selecting a group $\mathbb{G}$ in which LO is hard is by constructing $\mathbb{G}$ in such a way that there are no non-trivial elements of low order in $\mathbb{G}$. This results in LO trivially being hard in $\mathbb{G}$. This is done in Pietrzak's original construction by considering an RSA group with modulus $N = pq$ the product of two safe primes $p = 2p' + 1$, $q = 2q' + 1$, and taking $\mathbb{G} := \mathcal{QR}_N$ the subgroup of quadratic residues, which then has order $|\mathbb{G}| = p'q'$ the product of two large primes [Pie19].

Similarly, one could try to construct a class group $\mathrm{Cl}(\Delta)$ of an imaginary quadratic number field by selecting the discriminant Δ in such a way that the number of low-order torsion elements is limited. We saw in Section 2 that by taking $\Delta = -p$ where $p \equiv 3 \mod 4$, there are no elements of order two in $\mathrm{Cl}(\Delta)$. However, the Cohen-Lenstra heuristics [CL84] (see Conjecture 2.1) state that the probability $\mathfrak{U}(p)$ that an odd prime p divides $h(\Delta)$ for a random fundamental discriminant $\Delta < 0$, asymptotically goes to

$$\mathfrak{U}(3) \approx 0.439874, \quad \mathfrak{U}(5) \approx 0.239667, \quad \mathfrak{U}(7) \approx 0.163205, \quad \mathfrak{U}(11) \approx 0.0991673, \quad \ldots.$$

Provided that the heuristics hold up over the specific choice of discriminant $\Delta = -p$ with $p \equiv 3 \mod 4$, which our experiments in Section 3 confirm for small discriminants, there is a reasonable chance that the class group contains elements of low order. As we mentioned in Section 1.7, the Cohen-Lenstra heuristics predict that for p an odd prime, the average number of p-torsion elements approaches two (which has been proven for the case $p = 3$). More precisely, the Cohen-Lenstra heuristics [CL84, (C 5)] state that for a randomly chosen fundamental discriminant $\Delta < 0$ with $|\Delta| \leq D$ and a positive integer r, the probability that the p-rank $r_p(\mathrm{Cl}(\Delta))$ is equal to r approaches

$$\mathrm{Pr}_{p,r} := \lim_{D \to \infty} \Pr[r_p(\mathrm{Cl}(\Delta)) = r] = \frac{\prod_{n=1}^{\infty}(1 - p^{-n})}{p^{r^2} \prod_{n=1}^{r}(1 - p^{-n})^2},$$

which decreases very quickly as r increases. The first few values of $\mathrm{Pr}_{p,r}$ for $p = 3, 5, 7, 11$ and $r = 1, 2, 3, 4, 5$ are given in Table 12. Since the number of elements of order p in $\mathrm{Cl}(\Delta)$ is

Table 12: Values of $\mathrm{Pr}_{p,r}$ for $p = 3, 5, 7, 11$ and $r = 1, 2, 3, 4, 5$

r \ p	3	5	7	11
1	0.420	0.238	0.163	0.0991
2	0.0197	0.00206	0.000494	0.0000757
3	0.0000874	$6.71 \cdot 10^{-7}$	$2.96 \cdot 10^{-8}$	$4.70713 \cdot 10^{-10}$
4	$4.10 \cdot 10^{-8}$	$8.61 \cdot 10^{-12}$	$3.60 \cdot 10^{-14}$	$2.416 \cdot 10^{-17}$
5	$2.10 \cdot 10^{-12}$	$4.41 \cdot 10^{-18}$	$8.91 \cdot 10^{-22}$	$1.025 \cdot 10^{-26}$

determined by the p-rank as $p^{r_p(\mathrm{Cl}(\Delta))}$, the Cohen-Lenstra heuristics imply that there is a small probability that a large number of elements of low order exist in $\mathrm{Cl}(\Delta)$. Hence, if we pick a random negative fundamental discriminant from a large enough interval such that the corresponding class group is sufficiently large, then there is a very small probability of randomly sampling an element of low order in the class group.

In order for the low order problem LO to be hard, it must be hard to *purposely* find these low-order torsion elements. In Section 1.7 we discuss a possible avenue for computing non-trivial torsion elements in the class group. Recent results [OT20, Theorem 3], [CPV20, Lemma 8] show that it is possible to easily construct an element of order three in the non-maximal order $\mathcal{O} = \mathbb{Z}[\sqrt{-p}]$ in $K = \mathbb{Q}(\sqrt{-p})$ where p is a prime $\equiv 3 \mod 8$, but that this ideal becomes principal in the full ring of integers $\mathcal{O}_K = \mathbb{Z}[\frac{1+\sqrt{-p}}{2}]$. For the other two possible ways to select a discriminant discussed in Section 2, the even part of $\mathrm{Cl}(\Delta)$ is either 2 or 8, and the discriminant is composite. Moreover, we saw that in these cases, finding the elements of order two (i.e. the ambiguous classes of $\mathrm{Cl}(\Delta)$) is equivalent to factoring the composite discriminant. So if the factorization of Δ is kept a secret in this case, LO might still be hard. However, in this case a trusted setup is needed, which is not necessary when selecting the discriminant to be a negative prime.

In all these cases, if the multiple order problem MO is easy, the low order problem LO is going to be easy (assuming that the Cohen-Lenstra heuristics hold up). By the Cohen-Lenstra heuristics, the order of a randomly sampled group $\mathbb{G}$ is divisible by 3 with probability $\mathfrak{U}(3) \approx 0.439874$. If

this holds, the order of a randomly sampled $X \xleftarrow{\$} \mathbb{G}$ is divisible by 3 with probability at least $2/3$ by Lemma 1.25. Assume that a multiple N of the group order $|\mathbb{G}|$ is known. By recursively computing $X^{N/3^i}$ for $i = 1, \ldots, \lfloor \log_3(N) \rfloor$, we eventually obtain a non-trivial element of order 3. This procedure succeeds with probability at least $2/3 \cdot \mho(3) \approx 0.293249$ over the random choice of group $\mathbb{G}$ and element X. This can readily be turned into a formal proof that $\mathsf{LO} \Rightarrow \mathsf{MO}$ under the assumption of the Cohen-Lenstra heuristics. In [vBS21, Proposition 6.2] we show this reduction under the more general assumption that an oracle exists which outputs a small prime divisor of the group order with non-negligible probability.

Since there are no other known methods of solving LO and ARoot than through their reduction to MO (see Section 4.1.3 and Section 1.6, respectively), we refer to Section 2.1 and Section 2.2 for a discussion on the required classical discriminant size. Note that we can easily generate a random class group $\mathrm{Cl}(\Delta)$, without needing a trusted setup, by picking a prime $p \equiv 3 \mod 4$ and setting $\Delta := -p$. Additionally, class groups belonging to discriminants of this form do not contain any elements of order 2, which ensures that LO is at least not trivial to solve given the factorization of the discriminant.

4.1.4 Post-Quantum Security

As mentioned in Section 2.2, class groups are less secure in a post-quantum setting due to the existence of quantum polynomial time algorithms to determine the group order and discrete logarithms. However, VDF instances are typically used in interactive protocols with a bounded time span. Hence, they can potentially still be used, albeit using larger security parameters such that the quantum attack time exceeds any VDF instance's use time span.

In this setting we need to define an appropriate notion of security where the 'timer' starts running as soon as the adversary receives the group description and the VDF instance. Here the adversary succeeds to break sequentiality if they can evaluate the VDF before the timer runs out. Similarly, they succeed to break soundness if they can compute a valid proof on a false evaluation before the timer runs out. Because of this time constraint, the non-interactive versions of the proof protocols described in Section 4.1.3 are no longer suitable. The verifier needs to start a timer as soon as the group parameters are sent to the prover, and the validity of a proof can only be guaranteed if the prover's response is received before the timer runs out. For sequentiality we can define the following adapted version of Definition 4.3.

Definition 4.7 (T-single-evaluation race game)**.** Let $\mathcal{A}$ be a party playing the game. The parameter $\Delta : \mathbb{N} \to \mathbb{R}_{>0}$ is a function of the security parameter κ. The T-single-evaluation race game $\mathsf{Exp}^{T\text{-1-eval}}_{\mathcal{A},\mathsf{VDF}}$ is defined as follows, where keygen' additionally samples an instance x from a distribution over a finite subset of $\mathscr{A}^*$ of min-entropy at least κ.

$\underline{\mathsf{Exp}^{T\text{-1-eval}}_{\mathcal{A},\mathsf{VDF}}(\kappa):}$
$(\mathsf{pk}, x) \leftarrow \mathsf{keygen}'(1^\kappa)$
$Y \leftarrow \mathcal{A}^{\mathcal{O}(\cdot)}(1^\kappa, \mathsf{pk}, x)$
`if` $T(\mathcal{A}, x) < T \wedge Y = \mathsf{eval}_{\mathsf{pk}}(x, T)$ `return` 1
`else return` 0

$\underline{\mathcal{O}(x'):}$
`if` $x' \neq x$ `return`
$\mathsf{trapdoor}_{\mathsf{sk}}(x', T)$
`else return` $\perp$

Accordingly, we reach the following definition of single-evaluation sequentiality.

Definition 4.8 (T-single-evaluation sequential)**.** Let $T : \mathbb{N} \to \mathbb{R}_{>0}$ be a function of the security parameter κ. A trapdoor verifiable delay function VDF is Δ-single-evaluation sequential if for any PPT (potentially quantum) algorithm $\mathcal{A}$ there exists a negligible function $\mathsf{negl} : \mathbb{N} \to \mathbb{R}_{>0}$ such that for all $\kappa \in \mathbb{N}$:

$$\Pr[\mathsf{Exp}^{T\text{-1-eval}}_{\mathcal{A},\mathsf{VDF}}(\kappa) = 1] \leq \mathsf{negl}(\kappa).$$

For a non-trapdoor VDF one can analogously define T-single-evaluation sequentiality by omitting the oracle $\mathcal{O}(\cdot)$ from the T-single-evaluation race game.

For soundness, we similarly arrive at the following adapted version of Definition 4.2.

Definition 4.9 (T-single-evaluation soundness)**.** Let $T : \mathbb{N} \to \mathbb{R}_{>0}$ be a function of the security parameter κ. A (trapdoor) verifiable delay function VDF is T-single-evaluation sound if for any PPT (potentially quantum) algorithm $\mathcal{A}$ there exists a negligible function $\mathsf{negl} : \mathbb{N} \to \mathbb{R}_{\geq 0}$ such that for all $\kappa \in \mathbb{N}$ and $\mathsf{pk} \leftarrow \mathsf{keygen}(1^\kappa)$:

$$\Pr[Y' \neq \mathsf{eval}_{\mathsf{pk}}(x,T) \wedge \mathsf{verify}_{\mathsf{pk}}(x,Y',\pi',T) = \mathsf{true} \wedge T(\mathcal{A}) < T \,:\, (x,Y',\pi') \leftarrow \mathcal{A}(\mathsf{pk},T,1^\kappa)] \leq \mathsf{negl}(\kappa).$$

Precise cost analyses of quantum attacks for class groups are not well studied, but some inferences can be made from similar results on RSA. Given security parameter κ and an instance $X \in \mathrm{Cl}(\Delta)$ with targeted evaluation time T and timing parameter $t = T/\delta$, consider the following quantum adversary $\mathcal{A}$ attempting to break sequentiality of the VDF construction: $\mathcal{A}$ computes $|\langle X \rangle|$ using a (quantum) circuit of depth $S(\Delta)$ and subsequently computes $Y := X^{2^T \bmod |\langle X \rangle|}$ using approximately $\log|\langle X \rangle| \approx \log|\mathrm{Cl}(\Delta)|$ multiplications in $\mathrm{Cl}(\Delta)$. In Section 2.2 we have seen that the quantum algorithms for computing the order of $X \in \mathrm{Cl}(\Delta)$ are dominated by approximately $2\log|\mathrm{Cl}(\Delta)|\log\log|\mathrm{Cl}(\Delta)|$ sequential multiplications in $\mathrm{Cl}(\Delta)$. In this case $\mathcal{A}$ does not break T-single-instance sequentiality as long as

$$T < T(\mathcal{A}, X) \approx (2\log|\mathrm{Cl}(\Delta)|\log\log|\mathrm{Cl}(\Delta)| + \log|\mathrm{Cl}(\Delta)|) \cdot \delta(\Delta),$$

which, plugging in that $|\mathrm{Cl}(\Delta)| \approx |\Delta|^{1/2}$, roughly comes down to the condition that

$$t < 2\log|\mathrm{Cl}(\Delta)|\log\log|\mathrm{Cl}(\Delta)| + \log|\mathrm{Cl}(\Delta)| \approx \log|\Delta| \cdot \left(\log\log|\Delta| + \frac{1}{2} - \log 2\right).$$

Note that we assumed the time $\delta(\Delta)$ to compute multiplications in $\mathrm{Cl}(\Delta)$ is the same for both honest parties and the adversary, ignored the cost for the Fourier transforms in the order computing algorithm (which is asymptotically dominated by the above cost estimate), and ignored all constant factors in our complexity estimates. Similar estimates can be made to give a rough upper bound on the timing parameters for single-evaluation soundness, considering an adversary that attacks ARoot or LO through order computations.

As mentioned in Section 4.1.1, in a classical setting, it is not possible to use the VDF construction of Weselowski and Pietrzak as a *trapdoor* VDF when built on class groups, since there are no known efficient classical algorithms to compute the order of a class group. The existence of a quantum polynomial time algorithm to determine the class group order implies one *can* build trapdoor VDFs on class groups: The setup algorithm can use the polynomial time quantum algorithm to compute the order $|\mathbb{G}|$ which is the trapdoor for the VDF instance in the finite abelian group $\mathbb{G}$. However, there are two notable issues with this: (1) the setup algorithm needs to be executed by a trusted party who does not reveal the group before the trapdoor computation has finished; (2) this setting also implies that for every trapdoor VDF instance with targeted evaluation time T, one would need a setup phase taking time $> T$, which might not be desirable.

4.2 Zero-Knowledge Arguments

An *interactive proof system* is an interactive cryptographic protocol in which a prover can convince a verifier they know a witness for some NP statement. If the prover can do so without revealing anything to the verifier that can not be learned from the statement alone, the proof system is called *zero-knowledge*. The verifier should accept a proof of a true statement with high probability, which is called *completeness*, and it should be hard for a (potentially malicious) prover to convince the verifier of a false statement, which is called *soundness*. Intuitively, we want the time it takes to verify the proof to be more efficient than the time it would take to compute a witness. If additionally the communication cost between the prover and verifier is much smaller than a witness to the computation, the proof system is called *succinct*.

Interactive proofs can be made *non-interactive* by the Fiat-Shamir heuristic, in the process moving from zero-knowledge against honest verifiers for the interactive protocol to zero-knowledge against potentially malicious verifiers for the non-interactive protocol. The proof is called an *argument* when it only achieves soundness for computationally bounded provers, whereas a proof

achieves statistical soundness (for unbounded provers). A *succinct non-interactive argument* for general NP statement is also called a SNARG, and a zk-SNARG if it additionally achieves zero-knowledge. If additionally there exists a polynomial time *extractor* which outputs a witness on input a valid proof for a statement, we call the proof system a *succinct non-interactive argument of knowledge* or SNARK. This does not contradict zero-knowledge since the extractor receives some additional input, which is part of the provers input rather than part of the proof. This makes it also possible to construct zk-SNARKS. Finally, a proof system is called *transparent* if it does not need a trusted setup.

Bünz, Fisch and Szepieniec construct transparent SNARKs from a polynomial commitment scheme [BFS20]. The main contribution of this work is a polynomial commitment scheme which does not need a trusted setup, and can be built upon trustless hidden-order groups such as imaginary quadratic class groups. The security of their polynomial commitment scheme relies on the adaptive root assumption AdaptiveRoot and a variant of the strong RSA problem (similar to StrongRoot) in hidden-order groups of odd order [BFS20, Section 4.4], such as class groups $\mathrm{Cl}(-p)$ with $p \equiv 3 \mod 4$ prime. To construct zk-SNARKS, the authors construct a *hiding* variant of their polynomial commitment scheme, whose security is based on the strong RSA assumption (similar to StrongRoot) and the assumption that is hard to find any element of known nontrivial order (similar to LO, and implied by hardness of AdaptiveRoot) [BFS20, Theorem 6]. The authors use Wesolowski's proof of exponentiation (i.e., the protocol form Section 4.1.3 generalized to arbitrary coefficients) to reduce the time it takes for a verifier to validate commitments.

Block et al. [BHR+21] identify a gap in the proof of security in the polynomial commitment scheme from [BFS20]. They modify the construction to overcome this gap. Additionally, instead of using Wesolowski's proof of exponentiation, the authors generalize Pietrzak's proof of exponentiation (see Section 4.1.3) to arbitrary coefficients. Thanks to their adaptions the security of their polynomial commitment scheme solely relies on the hidden order assumption (EO in our context) [BHR+21, Theorem 4.2]. Finally the authors use slightly different techniques to construct a time- and space-efficient public-coin zero-knowledge argument system from their polynomial commitment scheme, again based on the hardness of EO [BHR+21, Theorem 4.1].

4.2.1 Post-Quantum Security

Just as in our discussion on the post-quantum security of VDFs in hidden-order groups from Section 4.1.4, zero-knowledge arguments based on hidden-order groups can still be useful in a post-quantum setting if soundness only needs to hold up for a certain amount of time and if the protocol has *statistical* zero-knowledge. This limited time makes it difficult to use *non-interactive* zero-knowledge arguments based on hidden-order groups in a post-quantum setting.

The time duration for an interactive protocol needs to be shorter than the time it takes to compute the group order, provided that a fresh group is sampled for every proof instance. If the time between the verifier sending their first message to the prover and the prover sending their final message to the verifier is shorter than the time it takes to compute the group order, soundness can still be guaranteed against the class of malicious provers who compute the group order as a subroutine in their attempt to convince the verifier of a false statement. The estimate from Section 4.1.4 can be adapted to give a lower bound on the required discriminant size $\log|\Delta|$ for a desired duration T of an interactive protocol whose soundness relies on the hardness of computing the order of a class group $\mathrm{Cl}(\Delta)$. Only taking the dominating step of the quantum order computation algorithm from Section 2.2 into account gives the relation

$$T < \log|\Delta| \cdot (\log\log|\Delta| - \log 2) \cdot \delta(\Delta),$$

where $\delta(\Delta)$ is the time a (malicious) prover uses to compute a single multiplication in $\mathrm{Cl}(\Delta)$.

Protocols whose zero-knowledge relies on some computational assumptions in a hidden-order group would not be suitable for post-quantum use, since information about the witness might be learned from the protocol transcript in polynomial time. This is not the case for protocols with *statistical* honest verifier zero-knowledge, such as the protocol from [BFS20], which remain zero-knowledge against quantum adversaries.

References

[Bac90] E. Bach. Explicit bounds for primality testing and related problems. *Mathematics of Computation*, 55(191):355–380, 1990.

[BBBF18] D. Boneh, J. Bonneau, B. Bünz, and B. Fisch. Verifiable delay functions. In *CRYPTO (1)*, volume 10991 of *Lecture Notes in Computer Science*, pages 757–788. Springer, 2018.

[BBF18] D. Boneh, B. Bünz, and B. Fisch. A survey of two verifiable delay functions. Cryptology ePrint Archive, Report 2018/712, 2018. `https://ia.cr/2018/712`.

[BBT94] I. Biehl, J. A. Buchmann, and C. Thiel. Cryptographic protocols based on discrete logarithms in real-quadratic orders. In *CRYPTO*, volume 839 of *Lecture Notes in Computer Science*, pages 56–60. Springer, 1994.

[Bea03] S. Beauregard. Circuit for shor's algorithm using 2n+3 qubits. *Quantum Inf. Comput.*, 3(2):175–185, 2003.

[Ber02] D. Bernstein. Arbitrarily tight bounds on the distribution of smooth integers. In *Proceedings of the Millennial Conference on Number Theory*, 2002.

[BFS20] B. Bünz, B. Fisch, and A. Szepieniec. Transparent snarks from DARK compilers. In *EUROCRYPT (1)*, volume 12105 of *Lecture Notes in Computer Science*, pages 677–706. Springer, 2020.

[BH01] J. Buchmann and S. Hamdy. A survey on IQ cryptography. In *In Proceedings of Public Key Cryptography and Computational Number Theory*, pages 1–16. De Gruyter, 2001.

[BHLV17] D. J. Bernstein, N. Heninger, P. Lou, and L. Valenta. Post-quantum RSA. In *PQCrypto*, volume 10346 of *Lecture Notes in Computer Science*, pages 311–329. Springer, 2017.

[BHR+21] A. R. Block, J. Holmgren, A. Rosen, R. D. Rothblum, and P. Soni. Time- and space-efficient arguments from groups of unknown order. In *CRYPTO (4)*, volume 12828 of *Lecture Notes in Computer Science*, pages 123–152. Springer, 2021.

[Bia10] J. Biasse. Improvements in the computation of ideal class groups of imaginary quadratic number fields. *Adv. Math. Commun.*, 4(2):141–154, 2010.

[BJ07] A. Bauer and A. Joux. Toward a rigorous variation of coppersmith's algorithm on three variables. In *EUROCRYPT*, volume 4515 of *Lecture Notes in Computer Science*, pages 361–378. Springer, 2007.

[BJS10] J. Biasse, M. J. Jacobson, and A. K. Silvester. Security estimates for quadratic field based cryptosystems. In *ACISP*, volume 6168 of *Lecture Notes in Computer Science*, pages 233–247. Springer, 2010.

[BK72] D. Boyd and H. Kisilevsky. On the exponent of the ideal class groups of complex quadratic fields. *Proceedings of the American Mathematical Society*, 31(2):433–436, 1972.

[BKSW20] K. Belabas, T. Kleinjung, A. Sanso, and B. Wesolowski. A note on the low order assumption in class group of an imaginary quadratic number fields. Cryptology ePrint Archive, Report 2020/1310, 2020. `https://ia.cr/2020/1310`.

[BMM00] J. Buchmann, M. Maurer, and B. Möller. Cryptography based on number fields with large regulator. *Journal de Théorie des Nombres de Bordeaux*, 12(2):293–307, 2000.

[BP96] E. Bach and R. Peralta. Asymptotic semismoothness probabilities. *Math. Comput.*, 65(216):1701–1715, 1996.

[BS06] W. D. Banks and I. E. Shparlinski. Integers with a large smooth divisor. *INTEGERS: Electronic Journal of Combinatorial Number Theory*, 7(#A17):1–11, 2006.

[BS13] E. Bach and J. P. Sorenson. Approximately counting semismooth integers. In *ISSAC*, pages 23–30. ACM, 2013.

[Bue84] D. A. Buell. The expectation of success using a monte carlo factoring method—some statistics on quadratic class numbers. *Math. Comput.*, 43(167):313–327, 1984.

[BV07] J. Buchmann and U. Vollmer. *Binary quadratic forms - an algorithmic approach*, volume 20 of *Algorithms and computation in mathematics*. Springer, 2007.

[BV16] M. Bhargava and I. Varma. The mean number of 3-torsion elements in the class groups and ideal groups of quadratic orders. *Proceedings of the London Mathematical Society*, 112(2):235–266, 2016.

[BW89] J. Buchmann and H. C. Williams. A key exchange system based on real quadratic fields. In *CRYPTO*, volume 435 of *Lecture Notes in Computer Science*, pages 335–343. Springer, 1989.

[Cas78] J. W. S. Cassels. *Rational quadratic forms*. London Mathematical Society Monographs 13. Academic Press, London, 1978.

[Cho34] S. Chowla. An extension of heilbronn's class-number theorem. *The Quarterly Journal of Mathematics*, os5(1):304–307, 1934.

[CL84] H. Cohen and H. W. Lenstra. Heuristics on class groups of number fields. In *Number Theory Noordwijkerhout 1983*, pages 33–62. Springer Berlin Heidelberg, 1984.

[CLM+18] W. Castryck, T. Lange, C. Martindale, L. Panny, and J. Renes. CSIDH: an efficient post-quantum commutative group action. In *ASIACRYPT (3)*, volume 11274 of *Lecture Notes in Computer Science*, pages 395–427. Springer, 2018.

[Coh93] H. Cohen. *A course in computational algebraic number theory*, volume 138 of *Graduate texts in mathematics*. Springer, 1993.

[Cop97] D. Coppersmith. Small solutions to polynomial equations, and low exponent RSA vulnerabilities. *J. Cryptol.*, 10(4):233–260, 1997.

[Cox89] D. Cox. *Primes of the form $x^2 + ny^2$: Fermat, class field theory, and complex multiplication*. A Wiley-Interscience publication. Wiley, New York, 1989.

[CPV20] W. Castryck, L. Panny, and F. Vercauteren. Rational isogenies from irrational endomorphisms. In *EUROCRYPT (2)*, volume 12106 of *Lecture Notes in Computer Science*, pages 523–548. Springer, 2020.

[DGS20] S. Dobson, S. D. Galbraith, and B. Smith. Trustless unknown-order groups. Cryptology ePrint Archive, Report 2020/196, 2020. `https://ia.cr/2020/196`.

[DH71] H. Davenport and H. A. Heilbronn. On the density of discriminants of cubic fields. II. *Proc. R. Soc. Lond. A*, 322(1551):405–420, 1971.

[DK02] I. Damgård and M. Koprowski. Generic lower bounds for root extraction and signature schemes in general groups. In *EUROCRYPT*, volume 2332 of *Lecture Notes in Computer Science*, pages 256–271. Springer, 2002.

[EFKP20] N. Ephraim, C. Freitag, I. Komargodski, and R. Pass. Continuous verifiable delay functions. In *EUROCRYPT (3)*, volume 12107 of *Lecture Notes in Computer Science*, pages 125–154. Springer, 2020.

[EV07] J. S. Ellenberg and A. Venkatesh. Reflection Principles and Bounds for Class Group Torsion. *International Mathematics Research Notices*, 2007, 2007.

[FKPS21] C. Freitag, I. Komargodski, R. Pass, and N. Sirkin. Non-malleable time-lock puzzles and applications. In *TCC (3)*, volume 13044 of *Lecture Notes in Computer Science*, pages 447–479. Springer, 2021.

[FS86] A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer, 1986.

[GE21] C. Gidney and M. Ekerå. How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits. *Quantum*, 5:433, 2021.

[Ham02] S. Hamdy. *Über die Sicherheit und Effizienz kryptographischer Verfahren mit Klassengruppen imaginär-quadratischer Zahlkörper*. PhD thesis, Technische Universität Darmstadt, 2002.

[HBP17] D. R. Heath-Brown and L. B. Pierce. Averages and moments associated to class numbers of imaginary quadratic fields. *Compositio Mathematica*, 153(11):2287–2309, 2017.

[HM00] S. Hamdy and B. Möller. Security of cryptosystems based on class groups of imaginary quadratic orders. In *ASIACRYPT*, volume 1976 of *Lecture Notes in Computer Science*, pages 234–247. Springer, 2000.

[HMT98] D. Hühnlein, A. Meyer, and T. Takagi. Rabin and RSA analogues based on non-maximal imaginary quadratic orders. In *ICISC*, pages 221–240. Korea Institute of Information Security and Cryptology (KIISC), 1998.

[HS97] S. Hunter and J. Sorenson. Approximating the number of integers free of large prime factors. *Math. Comput.*, 66(220):1729–1741, 1997.

[HS06] S. Hamdy and F. Saidak. Arithmetic properties of class numbers of imaginary quadratic fields. *JP Journal of Algebra, Number Theory and Applications*, 6:129–148, 2006.

[HT99] D. Hühnlein and T. Takagi. Reducing logarithms in totally non-maximal imaginary quadratic orders to logarithms in finite fields. In *ASIACRYPT*, volume 1716 of *Lecture Notes in Computer Science*, pages 219–231. Springer, 1999.

[HvdH21] D. Harvey and J. van der Hoeven. Integer multiplication in time $O(n\log n)$. *Annals of Mathematics*, 193(2):563 – 617, 2021.

[Jac99] M. J. Jacobson. *Subexponential class group computation in quadratic orders*. PhD thesis, Darmstadt University of Technology, Germany, 1999.

[JSW06a] M. J. Jacobson, R. E. Sawilla, and H. C. Williams. Efficient ideal reduction in quadratic fields. *International Journal of Mathematics and Computer Science*, 1:83–116, 2006.

[JSW06b] M. J. Jacobson, R. Scheidler, and H. C. Williams. An improved real-quadratic-field-based key exchange procedure. *J. Cryptol.*, 19(2):211–239, 2006.

[Kap74] P. Kaplan. Sur le 2-groupe des classes des corps quadratiques. *Mémoires de la Société Mathématique de France*, 37:115–116, 1974. URL `http://eudml.org/doc/94660`.

[Kit96] A. Y. Kitaev. Quantum measurements and the abelian stabilizer problem. *Electron. Colloquium Comput. Complex.*, 3, 1996.

[KL14] J. Katz and Y. Lindell. *Introduction to Modern Cryptography, Second Edition*. CRC Press, 2014.

[KO62] A. Karatsuba and Y. Ofman. Multiplication of many-digital numbers by automatic computers. *Dokl. Akad. Nauk SSSR*, 145:293–294, 1962.

[Lag80] J. C. Lagarias. Worst-case complexity bounds for algorithms in the theory of integral quadratic forms. *J. Algorithms*, 1(2):142–186, 1980.

[Lan94] S. Lang. *Algebraic Number Theory*. Graduate Texts in Mathematics. Springer, 1994.

[Lan01] E. Landquist. The quadratic sieve factoring algorithm. *Math*, 488:1–11, 2001.

[Lin97] S. C. Lindhurst. *Computing roots in finite fields and groups, with a jaunt through sums of digits*. PhD thesis, University of Wisconsin – Madison, 1997.

[Lit28] J. E. Littlewood. On the Class-Number of the Corpus $P(\sqrt{-k})$. *Proceedings of the London Mathematical Society*, s2-27(1):358–372, 1928.

[LSS20] E. Landerreche, M. Stevens, and C. Schaffner. Non-interactive cryptographic timestamping based on verifiable delay functions. In *Financial Cryptography*, volume 12059 of *Lecture Notes in Computer Science*, pages 541–558. Springer, 2020.

[LV01] A. K. Lenstra and E. R. Verheul. Selecting cryptographic key sizes. *J. Cryptol.*, 14(4):255–293, 2001.

[ME98] M. Mosca and A. Ekert. The hidden subgroup problem and eigenvalue estimation on a quantum computer. In *QCQC*, volume 1509 of *Lecture Notes in Computer Science*, pages 174–188. Springer, 1998.

[Mol96] R. Mollin. Solutions of diophantine equations and divisibility of class numbers of complex quadratic fields. *Glasgow Mathematical Journal*, 38(2):196–197, 1996.

[NIST20] N. I. of Standards and T. (NIST). Recommendation for key management: Part 1 – general. SP 800-57 Part 1 Rev. 5, `https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57pt1r5.pdf`, May 2020.

[NV10] P. Q. Nguyen and B. Vallée, editors. *The LLL Algorithm - Survey and Applications*. Information Security and Cryptography. Springer, 2010.

[OT20] H. Onuki and T. Takagi. On collisions related to an ideal class of order 3 in CSIDH. In *IWSEC*, volume 12231 of *Lecture Notes in Computer Science*, pages 131–148. Springer, 2020.

[Pie19] K. Pietrzak. Simple verifiable delay functions. In *ITCS*, volume 124 of *LIPIcs*, pages 60:1–60:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

[Rab83] M. O. Rabin. Transaction protection by beacons. *J. Comput. Syst. Sci.*, 27(2):256–267, 1983.

[Réd28] L. Rédei. Über die klassenzahl des imaginären quadratischen zahlkörpers. *Journal für die reine und angewandte Mathematik*, 159:210–219, 1928. URL `http://eudml.org/doc/149666`.

[RS75] J. B. Rosser and L. Schoenfeld. Sharper bounds for the chebyshev functions θ(x) and ψ(x). *Math. Comput.*, pages 243–269, 1975.

[RSW96] R. L. Rivest, A. Shamir, and D. A. Wagner. Time-lock puzzles and timed-release crypto. Technical report, Massachusetts Institute of Technology, 1996.

[Sch76] L. Schoenfeld. Sharper bounds for the chebyshev functions $\theta(x)$ and $\psi(x)$. II. *Math. Comput.*, 30(134):337–360, 1976.

[Sch91] A. Schönhage. Fast reduction and composition of binary quadratic forms. In *ISSAC*, pages 128–133. ACM, 1991.

[She13] R. L. Shepherd. Binary quadratic forms and genus theory. Master's thesis, The University of North Carolina at Greensboro, 2013.

[Sho97a] P. W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509, 1997.

[Sho97b] V. Shoup. Lower bounds for discrete logarithms and related problems. In *EUROCRYPT*, volume 1233 of *LNCS*, pages 256–266. Springer, 1997.

[SL84] C.-P. Schnorr and H. W. Lenstra. A monte carlo factoring algorithm with linear storage. *Math. Comput.*, 43(167):289–311, 1984.

[SS71] A. Schönhage and V. Strassen. Schnelle multiplikation großer zahlen. *Computing*, 7(3-4):281–292, 1971.

[Sta67] H. M. Stark. A complete determination of the complex quadratic fields of class-number one. *Michigan Mathematical Journal*, 14(1):1 – 27, 1967.

[Ste08] P. Stevenhagen. The number field sieve. *Algorithmic Number Theory*, 44:83–100, 2008.

[Sut07] A. V. Sutherland. *Order computations in generic groups*. PhD thesis, Massachusetts Institute of Technology, 2007.

[vBS21] A. van Baarsen and M. Stevens. On time-lock cryptographic assumptions in abelian hidden-order groups. In *ASIACRYPT (2)*, volume 13091 of *Lecture Notes in Computer Science*, pages 367–397. Springer, 2021.

[Vol00] U. Vollmer. Asymptotically fast discrete logarithms in quadratic number fields. In *ANTS*, volume 1838 of *Lecture Notes in Computer Science*, pages 581–594. Springer, 2000.

[vOW99] P. C. van Oorschot and M. J. Wiener. Parallel collision search with cryptanalytic applications. *J. Cryptol.*, 12(1):1–28, 1999.

[Wes19] B. Wesolowski. Efficient verifiable delay functions. In *EUROCRYPT (3)*, volume 11478 of *Lecture Notes in Computer Science*, pages 379–407. Springer, 2019.