Research Article

Carl Bootland*, Wouter Castryck, Alan Szepieniec and Frederik Vercauteren

A framework for cryptographic problems from linear algebra

https://doi.org/10.1515/jmc-2019-0032
Received July 15, 2019; accepted September 15, 2019

Abstract: We introduce a general framework encompassing the main hard problems emerging in lattice-based cryptography, which naturally includes the recently proposed Mersenne prime cryptosystem, but also problems coming from code-based cryptography. The framework allows to easily instantiate new hard problems and to automatically construct plausibly post-quantum secure primitives from them. As a first basic application, we introduce two new hard problems and the corresponding encryption schemes. Concretely, we study generalisations of hard problems such as SIS, LWE and NTRU to free modules over quotients of $\mathbb{Z}[X]$ by ideals of the form (f, g), where f is a monic polynomial and $g \in \mathbb{Z}[X]$ is a ciphertext modulus coprime to f. For trivial modules (i.e. of rank one), the case $f = X^n + 1$ and $g = q \in \mathbb{Z}_{>1}$ corresponds to ring-LWE, ring-SIS and NTRU, while the choices $f = X^n - 1$ and $g = X - 2$ essentially cover the recently proposed Mersenne prime cryptosystems. At the other extreme, when considering modules of large rank and letting $\deg(f) = 1$, one recovers the framework of LWE and SIS.

Keywords: LWE, SIS, NTRU, quotient ring, post-quantum

MSC 2010: 13M10, 11T71, 11H06

1 Introduction

Lattice-based and code-based cryptography are rapidly emerging as leading contenders for generating public-key cryptosystems that promise to withstand quantum attacks. The popularity of these branches of cryptography are due in large part to the simplicity and efficiency of their designs, but is certainly underscored by their strong security guarantees. Two hard problems in particular, the short integer solution (SIS) [3] and learning with errors (LWE) [46] problems, stand out in this regard. While these hard problems are expressible in the language of simple linear algebra over finite rings, and are hence easy to use, they are also provably hard-on-average, assuming the worst-case hardness of certain problems in lattices.

In response to the quadratic scaling of both operational cost and memory associated with a full matrix representation, many proposals switch to using structured matrices [34, 35, 48]. In essence, random matrices are replaced by matrices of multiplication by elements of the ring $R_q = \mathbb{Z}[X]/(f(X), q)$ resulting in the ring-based versions ring-SIS (RSIS) and ring-LWE (RLWE), respectively. Similar worst-to-average case reductions apply here, albeit from problems in structured lattices, which are potentially easier. Nevertheless, the low bandwidth requirements and high speed made possible by the designs from this category make their deployment an attractive option, and this in turn mandates careful study.

***Corresponding author: Carl Bootland,** ESAT/COSIC, KU Leuven, Kasteelpark Arenberg 10, 3000 Leuven, Belgium, e-mail: carl.bootland@kuleuven.be. http://orcid.org/0000-0002-8390-3410
Wouter Castryck, Department of Mathematics, KU Leuven, Celestijnenlaan 200B, 3000 Leuven, Belgium, e-mail: wouter.castryck@kuleuven.be. http://orcid.org/0000-0002-0191-5216
Alan Szepieniec, Frederik Vercauteren, ESAT/COSIC, KU Leuven, Kasteelpark Arenberg 10, 3000 Leuven, Belgium, e-mail: alan@nervos.org, frederik.vercauteren@kuleuven.be. http://orcid.org/0000-0002-7208-9599

Some recent constructions have similar features to these ring-based cryptosystems, but rely on modular big integer arithmetic rather than arithmetic involving polynomials. We classify the AJPS cryptosystem [1] and the I-RLWE cryptosystem of Gu [24] as members of this category, as well as several submissions to the NIST PQC project [51] such as Ramstake [49] and ThreeBears [25]. Despite relying on different types of rings, the underlying mechanisms of both categories bear a striking resemblance to each other in that a notion of "smallness" of elements is preserved under addition and multiplication operations. This operational similarity suggests the possibility of a unifying perspective and a generic framework for design and analysis. While the existence of such a unification is, perhaps, folklore, a detailed elaboration has not appeared in the literature before, at least not at the level of generality we have in mind.

Our main approach is to replace the ring R_q by a quotient ring of the form $R_g = \mathbb{Z}[X]/(f(X), g(X))$ with $f, g \in \mathbb{Z}[X]$ and some restrictions on which pairs one can take. This description captures both the familiar RLWE setting, where $g = q \in \mathbb{Z}_{>1}$, as well as the big integer arithmetic cryptosystems since, when $g(X) = X - b$ for some integer b, we have $(f(X), g(X)) = (f(b), X - b)$ so that $R_g = \mathbb{Z}[X]/(f(b), X - b) \cong \mathbb{Z}/(f(b))$. As such, our framework contains both RLWE and AJPS as special cases. To capture plain LWE and module-LWE, we will eventually work with free modules over R_g. Certain problems in code-based cryptography can also be seen to fit within our framework, such as decrypting a ciphertext in many code-based encryption schemes using only knowledge of the public key; this is sometimes called the general decoding (search) problem. The syndrome decoding problem (SDP), used in the Niederreiter cryptosystem [43] and its variants, is also closely related to the inhomogeneous version of the ideal-SIS problem. It is interesting to note that, just like with RLWE, code-based cryptosystems use additional structure (structured codes) which presents additional attack surfaces [47].

On top of the well-known examples, it should be clear that our framework will contain many more, possibly hard, problems that can be considered for use in cryptographic applications. A systematic treatment of the exact hardness of these problems would divert attention away from our current focus; hence we defer such analysis to a future work.

To identify some of the problems we face in this more general setting, consider the following standard noisy key agreement protocol. Let $\mathbf{G} \in R_g$ be a public parameter, typically sampled uniformly at random or generated pseudorandomly from a short seed. Alice samples two small elements $\mathbf{a}, \mathbf{b} \in R_g$, and Bob does the same for $\mathbf{c}, \mathbf{d}$. They then exchange $\mathbf{aG} + \mathbf{b}$ and $\mathbf{cG} + \mathbf{d}$, thus allowing Alice to obtain $\mathbf{a}(\mathbf{cG} + \mathbf{d})$ and Bob to obtain $\mathbf{c}(\mathbf{aG} + \mathbf{b})$ while thwarting any passive eavesdropper. If the difference $\mathbf{ad} - \mathbf{cb}$ is small, then, in principle, Alice can obtain secret key material identical to Bob's by correcting the errors or extracting an identical template, possibly with the aid of some additional reconciliation data. Several requirements are needed to make this protocol work.

Condition 1. The representation of elements of R_g must be conducive to efficient computation.

Condition 2. Sampling small elements must be possible, and moreover, whenever $\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}$ are small, then so is $\mathbf{ad} - \mathbf{cb}$.

Condition 3. The adversary must be unable to obtain $(\mathbf{a}, \mathbf{b})$ from $(\mathbf{G}, \mathbf{aG} + \mathbf{b})$ or $(\mathbf{c}, \mathbf{d})$ from $(\mathbf{G}, \mathbf{cG} + \mathbf{d})$.

Condition 4. It must be possible to correct small perturbations like $\mathbf{ad} - \mathbf{cb}$ or at least tolerate them somehow.

These conditions have been studied extensively in the standard case where $g = q \in \mathbb{Z}_{>1}$. This paper initiates the study of these same conditions in our more general setting. We view the aforementioned *ciphertext ring* R_g as the quotient of the *parent ring* $R := \mathbb{Z}[X]/(f(X))$ by the ideal gR. The parent ring is used to define smallness: informally, a small element of R_g is the reduction modulo g of an element of the parent ring having small coordinates (in absolute value) with respect to the *power basis* $1, X, X^2, \ldots, X^{\deg(f)-1}$. Furthermore, when computing in R_g, all variables are to be reduced into a set of representatives $\mathrm{Rep}(R_g)$; see Section 2.2 for details; this forces noisy expressions to wrap around so that they become hard to distinguish from random expressions. Against this framework, we will provide a thorough analysis of Conditions 1 and 2, thereby providing a new set of tools for the cryptographer's toolbox that are useful for various specific applications.

Condition 3 is addressed briefly in Section 3.1 but will be discussed in depth in a future work. Condition 4 will be discussed only superficially as it has a more ad hoc flavour.

Related work. The idea of using a polynomial rather than an integer for the *plaintext* modulus in lattice-based encryption schemes has already been considered by a number of authors [10, 13, 16, 27]. The idea of using general ideal lattices for the ciphertext space was also introduced in the context of fully homomorphic encryption by Gentry [22]; however, the hard problem he considers is different to ours. Attempts at unifying various lattice-based cryptographic problems is also not new, for example, the general learning with errors (GLWE) problem was proposed in [13]; their proposal essentially amounts to our ideal-LWE problem when restricting to $g = q \in \mathbb{Z}_{>1}$.

2 A recipe for generating problems

In this section, we present a general recipe for concocting problems on which to build cryptosystems. The recipe is given as a number of decisions to be taken before ending up with a problem. When following this recipe, it is instructive to think of having a fixed amount of resources (informally, this amount is the size of the problem) to allocate to the different ingredients. Here we simply state the choices to be made and do not attempt to answer the more difficult question of how to make the most appetising dish.

Throughout this section, we look at what choices are made in five different examples. Firstly, we start with plain LWE. Secondly, ring-LWE together with module-LWE are examined. Thirdly, we consider the problem underlying the NTRU Prime cryptosystem from [8]. Next, we have the problems underlying the two Mersenne prime cryptosystems due to Aggarwal, Joux, Prakash and Santha [1, 2]. Finally, we take an example from coding theory, that of the McEliece cryptosystem [37], in which quasi-cyclic codes are often used. We do not concern ourselves here with which specific codes are used.

2.1 Select the parent ring

The first choice one needs to make is the monic polynomial $f \in \mathbb{Z}[X]$ defining the parent ring $R = \mathbb{Z}[X]/(f)$. If we denote the degree of f by $n \geq 1$, then choosing a larger n requires allotting more of our resources to this ingredient. Furthermore, the size of the coefficients of f also affects the consumption of resources; one should keep these small in general so that Condition 2 holds. The parent ring naturally carries the structure of a free $\mathbb{Z}$-module with (power) basis $1, X, \ldots, X^{n-1}$.

Running example 1 (Plain LWE). Here f is taken to be a linear polynomial, the most obvious choice being $f = X$, so that $R = \mathbb{Z}[X]/(f) \cong \mathbb{Z}$. In this case, we use the least amount of resources possible.

Running example 2 (Ring-LWE and module-LWE). Here we let f be irreducible so that $R = \mathbb{Z}[X]/(f)$ is an order in a number field.[1]

Running example 3 (NTRU Prime). The NTRU Prime cryptosystem sets n to be an odd prime and takes $f = X^n - X - 1$, an irreducible polynomial.

Running example 4 (AJPS). The Mersenne prime cryptosystem lets $f = X^n - 1$ be such that $f(2) = 2^n - 1$ is a prime number; note that n is necessarily prime as well.

Running example 5 (McEliece). As with plain LWE, one chooses f to be linear and $R = \mathbb{Z}$.

1 More precisely, it is an order in the degree n number field $K = \mathbb{Q}[X]/(f)$. In fact, the formal definitions of ring-LWE [35] and module-LWE [30] require R to be the *maximal* such order, denoted by $\mathcal{O}_K$, which may not be true in our setting (if K is not monogenic, then this is even impossible). However, allowing for arbitrary orders would needlessly complicate our discussion, the more since there is no issue in the common scenario where f is a cyclotomic polynomial.

2.2 Select the ciphertext modulus

Next, we must choose a *ciphertext modulus* $g \in \mathbb{Z}[X]$, which defines the ciphertext ring $R_g = \mathbb{Z}[X]/(f, g)$ in terms of which our problems will be formulated. We impose some restrictions on the possible choices for g; throughout this paper, we assume that

(i) f and g are coprime, i.e., their only common divisors are ± 1: this ensures that R_g is a finite ring,
(ii) $\deg(g) < n$, which is not really a restriction since one can always replace g by $g \bmod f$,
(iii) there exists a positive integer a and a monic polynomial $r \in \mathbb{Z}[X]$ such that $(f, g) = (a, r)$ as ideals.

Assumption (iii) is the most restrictive, although not as badly as one might fear: a heuristic proportion of $6/\pi^2 \approx 60.8\,\%$ of all random pairs f and g satisfies this condition, which is confirmed by experiment (if satisfied, then r is linear with overwhelming probability). The reason for (iii) is it ensures that the ciphertext ring naturally comes equipped with a nice set of representatives

$$\mathrm{Rep}(R_g) = \{\alpha_{\deg(r)-1}X^{\deg(r)-1} + \cdots + \alpha_1 X + \alpha_0 \mid \alpha_i \in \{0, \ldots, a-1\}\}, \tag{2.1}$$

in which all computations are to be reduced; this ensures Condition 1 is satisfied. We stress that having such a nice set of representatives is our *only* reason for this assumption: it would be possible to weaken it if one is willing to end up with uglier or less canonical sets of representatives; though we avoid a detailed discussion. In Section 5, we will explain how to decide if such a and r exist, and if so, how to find them.

Just as with f, the degree of g and the size of the coefficients of g play a role in defining how much resources a certain g uses. In fact, it is better to consider the values of $\deg(r)$ and a as this is what defines the size of R_g: $\#R_g = \#\mathbb{Z}[X]/(f(X), g(X)) = \#\mathbb{Z}[X]/(a, r(X)) = a^{\deg(r)}$. It is also known that $\#R_g = |\mathrm{Res}(f, g)|$;[2] hence one does not need to first compute a and r to compute this value. Increasing this value naturally increases the size of the problem.

Running example 1 (Plain LWE). Here g is a positive integer, usually denoted by q, so that $R_g \cong \mathbb{Z}_q$ and $\#R_g = q$. In this case, one can take $a = q$ and $r = f$.

Running example 2 (Ring-LWE and module-LWE). Here again g is a positive integer q so that one can take $a = q$ and $r = f$, hence $\#R_g = q^n$.

Running example 3 (NTRU Prime). As above, g is a positive integer q and one takes $a = q$ and $r = f$.

Running example 4 (AJPS). Here $g = X - 2$, and one can take $a = 2^n - 1$ and $r = g = X - 2$ because we have equality of the two ideals $(X^n - 1, X - 2) = (2^n - 1, X - 2)$. Thus we have $\#R_g = 2^n - 1$.

Running example 5 (McEliece). As with plain LWE, we take g to be an integer q, but whereas, in plain LWE, q is relatively large, here we take $q = 2$, thus $\#R_g = 2$.

2.3 Select the rank

Thirdly, one must select a positive integer m, the *rank*, and construct the free R_g-module

$$M := R_g^m = \underbrace{R_g \times R_g \times \cdots \times R_g}_{m \text{ copies}}$$

consisting of vectors of length m with entries in R_g.

As with n (the degree of f), taking a larger m consumes more resources; indeed the size of an element of M is $m \deg(r) \log|a|$.

2 If $\phi_g : R \to R$ is the multiplication by g map, then [41] shows that $\det(\phi_g) = \mathrm{Res}(f, g)$; also $\ker \phi_g = \{0\}$ (since f and g are coprime), and we have $\mathrm{coker}\, \phi_g = R_g$. Looking at the Smith normal form of ϕ_g, we conclude that $\#R_g = |\det(\phi_g)|$.

Running example 1 (Plain LWE). Here m is a reasonably large integer and $M = R_q^m \cong \mathbb{Z}_q^m$.

Running example 2 (Ring-LWE and module-LWE). In ring-LWE, we take $m = 1$ so that $M = R_q$. In module-LWE, $m > 1$ is a relatively small integer, and the module M is given by R_q^m.

Running example 3 (NTRU Prime). Here $m = 1$ so that $M = R_q$.

Running example 4 (AJPS). Here again $m = 1$ so that $M = R_{X-2}$.

Running example 5 (McEliece). In this case, the value of m is the dimension of the code used.

2.4 Select the family of hard problems

After choosing the rank, we select one of the following three problems, which we call ideal-LWE, ideal-SIS and ideal-NTRU, respectively. Informally, these problems in their basic form are to solve a system of "noisy" linear equations, to find a non-zero solution to a system of linear equations which is "small" and to express a matrix as a quotient of two "small" matrices, respectively.[3] In each case, the base ring is $\mathbb{Z}_q$ for some positive integer q. These basic problems refer to standard LWE, standard SIS and a matrix variant of NTRU, alluded to in [26] when comparing NTRU to McEliece.[4]

The simplest way to generalise these basic problems is to replace the random matrix defining the linear system by a *matrix of multiplication*; that is a linear map on a free $\mathbb{Z}_q$-module defined by multiplying by an element of that module. This gives the matrix some structure allowing for a more compact representation and gives rise to the ring versions of the problems. In particular, this gives the standard NTRU problem.

The second main way to generalise the basic problem is to take entries from a larger ring than $\mathbb{Z}_q$, such as the ring R_g, which is a $\mathbb{Z}_a$ module itself.[5] Thus we can replace the ring elements by $\deg(r) \times \deg(r)$ matrices of multiplication with entries in $\mathbb{Z}_a$ which gives a block structure to the original matrix. This is the general module approach which gives rise to the module variants of the problems when $g = a \in \mathbb{Z}$.

Now that we have seen the two main generalisations; we give the details of how this can be applied to each problem.

Ideal-LWE. For the ideal-LWE problem, one chooses two further parameters k, the number of "keys", and ℓ, the number of samples (which will depend on the application).[6] The problem is then defined as follows.

Problem 1 (Ideal-LWE search problem). *Let χ be a distribution on R defining small elements, and let k and ℓ be positive integers. Sample a uniformly random element $\mathbf{s}$ from $R_g^{m\times k}$. The ideal-LWE search problem is to find $\mathbf{s}$ given the tuple $(\mathbf{a}, \mathbf{b}) \in R_g^{\ell\times m} \times R_g^{\ell\times k}$, where $\mathbf{a} \in R_g^{\ell\times m}$ is sampled uniformly at random and $\mathbf{b} = \mathbf{a} \times \mathbf{s} + \mathbf{e} \in R_g^{\ell\times k}$ with $\mathbf{e}$ sampled from $\chi^{\ell\times k}$.*

In a number of circumstances, one often wants to sample the secret $\mathbf{s}$ not from the whole space but some subset of elements, for example by sampling it using the error distribution. This so-called "small secret" case allows more powerful cryptographic constructions to be built as multiplying by $\mathbf{s}$ preserves smallness. See [14, Section 4] and [40] for a reduction from the general case to the small secret case.

Ideal-SIS. In the ideal-SIS and ideal-NTRU problems, we require a norm on the parent ring, $\|\cdot\| \colon R \to \mathbb{R}_{\geq 0}$. We abuse notation and write $\|\mathbf{a}\| < \rho$ for $\mathbf{a} \in R^m$ if, for all components $\mathbf{a}_i$ of $\mathbf{a}$, the relation $\|\mathbf{a}_i\| < \rho$ holds.

Problem 2 (Ideal-SIS search problem). *Given an integer $\ell > m$ and a bound ρ, sample ℓ elements from $M = R_g^m$ uniformly at random, denoted $\mathbf{a}_1, \ldots, \mathbf{a}_\ell$. The ideal-SIS problem is to find a non-zero $\mathbf{z} = (\mathbf{z}_1, \ldots, \mathbf{z}_\ell) \in R^\ell$ such that $\|\mathbf{z}\| \leq \rho$ and $\sum_{i=1}^{\ell} \mathbf{a}_i \cdot \mathbf{z}_i = \mathbf{0}$.*

3 The definition of what exactly "small" means and what a distribution of small elements is, is left to the next section.

4 See also [42], where this is elaborated in more detail.

5 Recall we have $(f, g) = (a, r)$ for some $a \in \mathbb{Z}$.

6 Often one considers ℓ to simply be polynomially bounded in the security parameter rather than fixed.

One often considers the inhomogeneous problem where, instead of finding a linear combination summing to zero, one is given a target vector which the linear combination must sum to; this is also sometimes called the knapsack problem.

Ideal-NTRU. The final problem we consider is the ideal-NTRU problem.

Problem 3 (Ideal-NTRU search problem). *Let χ be a distribution of small elements on R with appropriate bound ρ. Sample $\mathbf{u} \leftarrow \chi^{m\times m}$ such that it is invertible in $R_g^{m\times m}$ and $\mathbf{v} \leftarrow \chi^{m\times m}$.*[7] *Now, considering $\mathbf{u}$ and $\mathbf{v}$ as elements of $R_g^{m\times m}$, set $\mathbf{h} = \mathbf{v}\mathbf{u}^{-1} \in R_g^{m\times m}$.*[8] *Then, given $\mathbf{h}$ and ρ, the ideal-NTRU search problem is to find a pair $(\mathbf{u}', \mathbf{v}') \in R^{m\times m} \times R^{m\times m}$ with $\mathbf{u}'$ invertible modulo g, $\mathbf{h} = \mathbf{v}'\mathbf{u}'^{-1} \bmod g$, $\|\mathbf{u}'\| < \rho$ and $\|\mathbf{v}'\| < \rho$.*

Unlike with the previous choices, the cost of picking a certain problem is not so obvious; one could consider, for example, the size of the space to which the solution to the set of linear equations belongs, but this is not so easy to compute in the ideal-SIS and ideal-NTRU cases when the solution is restricted to being small. We point out that the size of the problem is related but not directly equivalent to the hardness of a problem. For most choices of parameters, the best known attacks rely on lattice reduction; hence, in general, the cost will depend on the dimension of the lattice being reduced which need not directly reflect the size of the problem.

Running example 1 (Plain LWE). Naturally, we select the ideal-LWE problem here.

Running example 2 (Ring-LWE and module-LWE). This again amounts to selecting the ideal-LWE problem.

Running example 3 (NTRU Prime). Here we select the ideal-NTRU problem.

Running example 4 (AJPS). The version of [1] amounts to selecting the ideal-NTRU problem, while the corresponding NIST submission [2] amounts to selecting ideal-LWE.

Running example 5 (McEliece). Here we consider the general decoding problem of decrypting a ciphertext using only the public key. One essentially takes the ideal-LWE problem with a fixed number of samples (the length of the code).

2.5 Distribution of small elements

Finally, we come to the issue of what a *small element* is. Informally spoken, by a small element of R, we mean an element having small coordinates (in absolute value) with respect to the power basis. The archetypal example is that each coordinate is sampled from a discrete Gaussian distribution with standard deviation σ. The LWE type problems all typically use this type of distribution. One can also consider the case when the coefficients are not sampled independently, as in the case of RLWE as defined in [35], as soon as one moves away from the 2-power cyclotomic case. When σ becomes small enough, the coefficients are, with high probability, in the set $\{-1, 0, 1\}$. When not sampled independently, it becomes possible to essentially sample vectors of a specified Hamming weight; this is the distribution used in the NTRU setting.

The question of precisely how small to take small elements is complex and depends on the problem and application. In general, larger errors give harder problems but may inhibit functionality and performance of certain cryptographic schemes.

7 The case of non-square $\mathbf{v}$ can also be considered.

8 We also have the choice of multiplying $\mathbf{v}$ on the left by $\mathbf{u}^{-1}$, but this leads to the same problem; however, there is a third option: to multiply $\mathbf{v}$ by the inverse of two small square matrices, one on the left and one on the right. This is done in [18].

3 A catalogue of problems

Now that we have a general outline for our recipe, we can consider what problems we can create using it. To this end, we start to build a catalogue of problems by looking at examples already in the literature, a number of which we have seen already.

Ideal-LWE. We first consider those using the ideal-LWE problem. If one takes the ciphertext modulus g to be an integer and set $k = 1$, then we get the familiar LWE type problems: when $\deg(f) = 1$ and $m > 1$, we get standard LWE, when $\deg(f) > 1$ and $m = 1$, we have the (poly-)RLWE problem,[9], and bridging them when $\deg(f) > 1$ and $m > 1$, we find module-LWE. An example for when $k > 1$ is the matrix LWE problem from [11] which still takes g to be an integer.

In contrast, if one takes $g(X) = X - b$ for some integer b and $\deg(f) > 1$, then one obtains LWE-like problems but associated with big integer arithmetic. We identify the I-MLWE problem of ThreeBears [25] ($m > 1$, $k = 1$) and I-RLWE problem of Gu [24] ($m = k = 1$) as members of this class. Further, the Mersenne-756839 submission to NIST [50] defines and uses the Mersenne low Hamming combination (MLHC) search problem for security; this is essentially the I-RLWE problem when $b = 2$ and the secret $\mathbf{s}$ is not uniformly random but sampled from the distribution χ. The Ramstake submission [49] also makes use of the MLHC problem.

Ideal-NTRU. Next we consider examples of the ideal-NTRU problem. When $m = 1$ and $\deg(f) > 1$, we capture standard NTRU [27] along with NTRU Prime [8] and many other variants when taking $g(X)$ an integer; in addition, we have the Mersenne low Hamming ratio (MLHR) problem [1] when $g(X) = X - 2$. Furthermore, for $m > 1$ and $g \in \mathbb{Z}$, we have the basic matrix formulation of NTRU [42] when $\deg(f) = 1$, while MaTRU [18] uses $\deg(f) > 1$.

Ideal-SIS. Finally, with the ideal-SIS problem, there are relatively few examples in the existing literature; all take g to be an integer. When $\deg(f) = 1$ and $m > 1$, we have the standard SIS problem [3], when $\deg(f) > 1$ and $m = 1$, we have the ring-SIS problem [39], and when both $\deg(f) > 1$ and $m > 1$, we reach the module-SIS problem [30]. In the case when both $\deg(f)$ and m are taken to be one, the resulting problem is the (homogeneous) modular subset sum problem (SSP).

We arrange all of these examples in Table 1 classified by the problem family they utilise, the degrees of f and g as well as whether the rank m is one or larger than one. We colour each cell either red (and mark with a *) when we do not consider the problem as $\deg(g) \geq \deg(f)$, yellow when there is a known example in the current literature, or green (marked with a question mark) when the problem has, to the best of our knowledge, not yet been considered.

Looking at the green entries in the tables, we can immediately see a number of empty entries. Firstly, there seems to be no analogue of NTRU over the integers which appears to be hard; the problem can be solved easily by performing lattice reduction on the 2-dimensional lattice spanned by the row vectors $(1, h)$, $(q, 0)$ and $(0, q)$, where h is the quotient of small elements in $\mathbb{Z}_q$. Secondly, to the best of our knowledge, no one has proposed a matrix version of the NTRU problem over the AJPS ring $\mathbb{Z}[X]/(X^n - 1, X - 2) \cong \mathbb{Z}/(2^n - 1)$. Thirdly, the ring and module variants of the SIS problem have also not been considered when using this ring. Finally, as we have already stated, we know of no paper which explicitly considers the case when the modulus g has degree larger than one.

Cryptographic applications. In practice, as cryptographers, our end goal is to build cryptographic schemes which rely on the hardness of a given problem. Just as with deriving a problem by following the above recipe, many of the known cryptographic applications can equally be built almost automatically on top of the new problems in much the same way as when building them from the standard problems; see for example [5] for a detailed analysis of what can be built from certain primitives using algebraic structure. The motivating key-exchange example in the introduction essentially forms the basis for most applications we consider here.

9 We note that the RLWE problem is usually stated in terms of the codifferent $R^\vee$ [35, 36], but this can be avoided by using a different error distribution [15]. Therefore, we do not consider this option in detail.

	$m = 1$		$m > 1$	
$\deg(g)$	$\deg(f) = 1$	$\deg(f) > 1$	$\deg(f) = 1$	$\deg(f) > 1$
	Ideal-LWE			
0	1-dimensional LWE [14]	RLWE [35]	LWE, LPN [46], McEliece [37], matrix LWE [11]	M-LWE [13, 30]
1	*	I-RLWE [24], MLHC [2]	*	I-MLWE [25]
⋮	*	?	*	?
	Ideal-NTRU			
0	?	NTRU [27], NTRU Prime [8]	matrix NTRU [42]	MaTRU [18]
1	*	MLHR [1]	*	?
⋮	*	?	*	?
	Ideal-SIS			
0	modular SSP	RSIS [39]	SIS [3]	M-SIS [30]
1	*	?	*	?
⋮	*	?	*	?

Table 1: The catalogue of problems, separated into their separate problem families, and classified by whether m is one or larger, whether the degree of f is one or larger, and the degree of g. Known examples are filled in and the cell coloured yellow, red (*) boxes we do not consider due to the restriction on g, and green (?) give new problems.

In this respect, we find that the LWE family is the most useful to us, while the SIS family has the fewest known applications to date.

From the problems belonging to the LWE family, we can build basic primitives such as public key encryption [44, 46], key exchange [7, 19], digital signatures [4, 33][10] and oblivious transfer [12, 44], as well as more advanced constructs such as identity-based encryption [23] and fully homomorphic encryption [13, 21].

As for the NTRU family, there are known constructions for much the same primitives: public key encryption [8, 27], digital signatures [28], oblivious transfer [38], identity-based encryption [20] and fully homomorphic encryption [32]; although the latter is not considered competitive due to the attacks presented in [6, 17, 29].

The SIS family has turned out to be far less fruitful; however, it has still been used to create a digital signature scheme via hashing [23]. It is also known that one can build zero knowledge proofs from the inhomogeneous SIS problem [31].

We expect that most of the above primitives can be straightforwardly adapted to work using our more general problems, and we give some simple examples in the case of public-key encryption in the next section.

3.1 An introduction to security

Here, we briefly look at lattice attacks on the three families of problems. The general idea of such an attack is to construct a lattice from the publicly available information which either contains a short vector which depends on secret information (such as an element from the distribution of small elements used), or for which we know a vector (in the ambient space) which is close to a lattice point which again depends on a secret; by finding such a short or closest vector, we can recover information about the secret key. To be able to find such a lattice vector, one uses a technique called lattice reduction. This is a process which takes as input some generating set for the lattice with the goal of returning a basis of the lattice consisting of short and nearly orthogonal vectors. One important property of lattice reduction is that it works on *integer* lattices. Since we

10 See also the NIST competition for more constructions of these three primitives [51].

primarily work with the ring R_g and small elements are only defined in R, when describing such a lattice, we will have to include the generators $X^i g(X) \bmod f(X)$ which will account for the fact that we work modulo g. For example, in the simple case of the primal attack on the ideal-LWE problem instantiated with $k = m = 1$ and $\mathbf{a}$ being lifted to a vector of polynomials $(a_1, a_2, \ldots, a_\ell)^T \in R^\ell$ and similarly for $\mathbf{b}$, lattice reduction is performed on the $(\ell n + 1)$-dimensional lattice generated by the rows of the matrix

$$\left(\begin{array}{c|c|c|c} - \ b_1 \ - & \cdots & - \ b_\ell \ - & w \\ \hline - \ a_1 \ - & \cdots & - \ a_\ell \ - & \\ \vdots & & \vdots & \\ - \ X^{n-1}a_1 \bmod f \ - & \cdots & - \ X^{n-1}a_\ell \bmod f \ - & \\ \hline - \ g \bmod f \ - & & & \\ \vdots & & & \\ - \ X^{n-1}g \bmod f \ - & & & \\ \hline & \ddots & & \\ \hline & & - \ g \bmod f \ - & \\ & & \vdots & \\ & & - \ X^{n-1}g \bmod f \ - & \end{array}\right)$$

for some positive integer weight w. The aim is then to recover the short vector $\pm(e_1, \ldots, e_\ell, w)$.

In the classical setting, when $g(X) = q$ is simply an integer, this will mean we use as generators vectors which are zero in all but one component where it takes the value q. In applications using LWE and RLWE, the value of q will be rather large and does not cause any problems, but in code-based cryptography, it is typical to take $q = 2$. This implies that the lattice contains many trivial vectors of Euclidean length $\sqrt{2}$ essentially rendering lattice reduction attacks useless. Another consequence of taking $q = 2$ is that removing the errors introduced by the scheme is in general a hard task and why special codes which have an efficient decoding algorithm are needed in practical applications.

Just as in the integer ciphertext modulus case, when g is chosen as a polynomial, it may be that the constructed integer lattice has trivial short vectors, much shorter than any vector containing information about the secret key. In our Running example 4, for example, g is taken as $X - 2$ which, in combination with $f(X) = X^n - 1$, yields many vectors of length $\sqrt{5}$ being present in the lattice which once again renders straightforward lattice reduction attacks futile.

That is not to say lattice reduction does not have a place in attacking our problems for this choice of g; see for example [9], only that it is not the main cost in such attacks. The attack is very similar in spirit to the family of general information-set decoding attacks first introduced by Prange [45] and is more combinatorial in nature, involving finding a set of coordinates which are in some sense error-free. Similar to code-based cryptography, it is in general difficult to recover the small elements used when $g = X - 2$. This problem was avoided in [1] by ensuring decryption could be performed without recovering the error in the ciphertext; however, they were only able to encrypt one bit per ciphertext. To improve the efficiency of the scheme, by allowing a much larger plaintext space, the authors had to employ an error-correcting code in their scheme [2].

The cases of "large" g and "small" g are in some sense two ends of a spectrum, and the applicable attacks in each case are very different.[11] This leaves open the problem of finding the boundary between the two cases where lattice attacks stop working and combinatorial approaches start to become feasible. It may be possible that choices of parameters towards the middle of this spectrum offer superior security guarantees and/or allow for more efficient schemes.

More details on lattice attacks, and more general attacks, on our problems will appear in a future work; see also the forthcoming PhD thesis of the first author.

11 Technically, the distinction is whether gR contains a polynomial with a sufficiently short vector of coefficients.

4 New examples

4.1 Generalising the Gu encryption scheme to higher degree g

Here, we present a generalisation of the Gu encryption scheme [24] where, instead of taking g to be linear, we consider g of higher degree. We first define our parent ring as $R = \mathbb{Z}[X]/(X^n + 1)$, that is, we take $f(X) = X^n + 1$. Next, we carefully choose our ciphertext modulus $g = X^d + b$, where $b > 1$, such that $d \mid n$, $d < n$ and $q = b^{n/d} + (-1)^{n/d}$ is prime.[12] Then we have that the ideal generated by f and g is also generated by g and the prime q; this is because $f = (X^d)^{n/d} + 1 \equiv (-b)^{n/d} + 1 = (-1)^{n/d} q \bmod g$. Therefore, we have that $R_{X^d+b} \cong \mathbb{Z}_q^d$ as abelian groups by considering a polynomial of degree at most $d - 1$ as a vector of d coefficients. We will use this as a set of representatives of R_g; see equation (2.1). We also take the rank to be one to simplify the discussion somewhat, but one can easily consider a module version of our scheme. Finally, we choose a plaintext modulus p; the plaintext space will be $\mathbb{Z}_p^n$.

Next, we define a distribution of small elements in R, χ_σ, by sampling n coefficients from a discrete Gaussian distribution with standard deviation σ, and forming a polynomial of degree $n - 1$ from these coefficients. This polynomial will then be reduced modulo g in our scheme to one with d coefficients, which need not be small with respect to q; indeed we expect them not to be. We denote by $\overline{\chi}_\sigma$ the distribution on $\mathbb{Z}_q^d$ given by sampling from χ_σ and reducing modulo g. In practice, to sample from $\overline{\chi}_\sigma$, one will, for each of the d entries, sample n/d coefficients from the discrete Gaussian, say ϵ_i, and compute $\sum_{i=0}^{n/d-1} \epsilon_i(-b)^i$ as the entry. Thus we see that σ should be much smaller than b.

Key generation. To generate a key, we sample an element $\mathbf{a}$ uniformly at random from $R_{X^d+b} \cong \mathbb{Z}_q^d$ as well as elements $\mathbf{s}, \mathbf{e} \leftarrow \overline{\chi}_\sigma$. Compute $\mathbf{b} = \mathbf{as} + p\mathbf{e}$. The public key is the pair $(\mathbf{a}, \mathbf{b})$, while the private key is $\mathbf{s}$.

Encryption. Given a plaintext $\mathbf{m} \in \mathbb{Z}_p^n$, consider it as a polynomial in R with coefficients in $[-p/2, p/2)$, and denote by $\overline{\mathbf{m}}$ the reduction of this polynomial modulo $X^d + b$. Sample elements $\mathbf{r}, \mathbf{e_1}, \mathbf{e_2} \leftarrow \overline{\chi}_\sigma$, and compute $\mathbf{c_1} = \mathbf{ar} + p\mathbf{e_1}$ and $\mathbf{c_2} = \mathbf{br} + p\mathbf{e_2} + \overline{\mathbf{m}}$, where $(\mathbf{a}, \mathbf{b})$ is the public key of the intended recipient. The ciphertext is the pair $(\mathbf{c_1}, \mathbf{c_2})$.

Decryption. Given a ciphertext $(\mathbf{c_1}, \mathbf{c_2})$ and a private key $\mathbf{s}$, one first computes $\mathbf{d} = \mathbf{c_2} - \mathbf{c_1 s}$. For each coefficient d_i, consider it an integer in $[-q/2, q/2)$, and compute the balanced expansion with base $-b$, say $d_i = \sum_j \alpha_{i,j}(-b)^j$, where $\alpha_{i,j} \in [-b/2, b/2)$. Then, for $k = 0, \ldots, n - 1$, define $m_k = \alpha_{i,j} \bmod p$, where $i = k \bmod d$ and $j = \lfloor kd/n \rfloor$. Return the vector $\mathbf{m} = (m_k)$.

Security. Just as in [24, Theorem 3.9], for the specific choices of f and g taken here, we can convert an RLWE sample with $f = X^n + 1$ and $g = b$ to an ideal-LWE sample with the same f but $g = X^d + b$ and conversely transform an ideal-LWE sample into a RLWE sample, in both cases with a growth in the noise present in the sample. The conversions are simple to write down. To go from RLWE to ideal-LWE, for each polynomial in R_b (i.e. $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{s}$), lift it to a polynomial in R with coefficients in the symmetric interval around zero, and then reduce modulo $X^d + b$. In the reverse direction, for each element in R_{X^d+b} with coefficients in the symmetric interval about zero, lift it to a polynomial in R by expanding the coefficients to the base b with the coefficients of powers of b in the range $[-b/2, b/2)$ and then substituting b with $-X^d$. Reduction modulo b gives an element of R_b.

A proof of the reductions is essentially the same as that given in [24] with the same bound on the growth of the noise.

12 If n/d is odd, then $b^{n/d} - 1$ is divisible by $b - 1$, so the only way for it to be prime is when $b = 2$ and n/d is prime; hence q must be a Mersenne prime. In our case, we want b to be large, so we will always require n/d to be even. The choice of n being a power of two gives generalised Fermat primes, and we, of course, require b to be even.

Somewhat homomorphic encryption. It is easy to transform this scheme into a somewhat homomorphic scheme akin to, for example, the Brakerski–Fan–Vercauteren scheme [21]. Implementing this, we found that, with the same parameters used in practice, we could perform on average between zero and three fewer multiplicative levels than with the original scheme.[13]

4.2 Module-NTRU over the AJPS ring

In this section, we briefly describe a cryptosystem employing the ideal-NTRU problem with rank larger than one and which takes as the underlying ring the AJPS ring; this means, we will take f as $X^n - 1$ for some prime n such that $q = 2^n - 1$ is also prime, and g as $X - 2$. We also choose positive integers d and $w \ll n$, where d will be the rank of the module used and w will be the Hamming weight of elements sampled from our distribution of small elements. Formally, we define χ_w to be the uniform distribution over the set $\{\sum_{i\in\mathcal{I}} 2^i \mid \mathcal{I} \subset \{0, 1, \ldots, n-1\}, \#\mathcal{I} = w\}$. The plaintext space will be $\{0, 1\}^d$, and for decryption, we will choose two thresholds t_l and t_u satisfying $0 \leq t_u < t_l \leq n$.

Key generation. To generate keys, first sample two matrices $\mathbf{u}$ and $\mathbf{v}$ from $\chi_w^{d\times d}$ with the condition that $\mathbf{u}$ is invertible modulo q. Compute $\mathbf{w} = \mathbf{v}\mathbf{u}^{-1}$. The public key is $\mathbf{w}$, and the private key is $\mathbf{u}$.

Encryption. Given a public key $\mathbf{w}$ and a message $m \in \{0, 1\}^d$, denote by $\mathbf{m}$ the $d \times d$ diagonal matrix with the message bits down the diagonal. To encrypt, sample two matrices $\mathbf{r}$ and $\mathbf{e}$ from $\chi_w^{d\times d}$ and a diagonal matrix $\mathbf{d}$ with uniformly random coefficients modulo q. Compute the ciphertext as $\mathbf{c} = \mathbf{r}\mathbf{w} + \mathbf{m}\mathbf{d} + \mathbf{e}$.

Decryption. To decrypt the ciphertext $\mathbf{c}$ with the private key $\mathbf{u}$, first compute the product $\mathbf{p} = \mathbf{c}\mathbf{u}$. Then, for each i in $\{1, \ldots, d\}$, consider the elements in the ith row of $\mathbf{p}$ as binary strings of length n, and compute the mean of the Hamming weights of these binary strings. If this mean is at most the threshold t_l, set $m_i = 0$; if this mean is no smaller than t_u, set $m_i = 1$, and otherwise abort. Return the vector (m_i).

Decryption works since we have $\mathbf{p} = \mathbf{c}\mathbf{u} = \mathbf{r}\mathbf{v} + \mathbf{m}\mathbf{d}\mathbf{u} + \mathbf{e}\mathbf{u}$, and the entries of $\mathbf{r}\mathbf{v}$ and $\mathbf{e}\mathbf{u}$ will still have relatively small Hamming weight, while the entries of $\mathbf{m}\mathbf{d}\mathbf{u}$ will be zero in the ith row if $m_i = 0$ and be uniformly random if $m_i = 1$. The probability that d uniformly random elements have a mean Hamming weight smaller than the threshold t_l can be made negligibly small by choosing the parameters appropriately.

5 Generic moduli

In this final section, we look at the structure of the ring R_g for generic g. Then our ring $R_g = \mathbb{Z}[X]/(f(X), g(X))$ does not have an obvious canonical set of representatives. In order to have useful representatives, we will try to find a pair $a \in \mathbb{Z}_{>0}$ and $r \in \mathbb{Z}[X]$ such that $(f, g) = (a, r)$. When r is monic, we can use the set of representatives from equation (2.1). We note that if r is not monic, then a set of representatives is still possible to write down but is not so user-friendly. Our choice of g will be constrained by R_g having such a set of representatives.

Now our task is to find such a and r if they exist. It is natural to choose a to be the smallest positive integer in (f, g) so that $(f, g) \cap \mathbb{Z} = (a)$ which always exists due to the coprimality of f and g. This integer is called the "congruence number" or "reduced resultant" of the polynomials f and g. Then r is defined only modulo a and up to units of $\mathbb{Z}_a[X]$. The overall strategy is first to find a. Afterwards, we search for an r using the Euclidean algorithm in the ring $\mathbb{Z}_a[X]$. When a is composite, $\mathbb{Z}_a$ is not an integral domain so that finding inverses modulo a can fail. However, in this case, we will have found a factor of a and can use this factor, with some work, to either split a into a product of coprime factors, work modulo each of these factors and combine the results using the Chinese remainder theorem, or write a as a power and use Hensel lifting to find r. Of course, these subroutines can also fail when a division fails, but we recurse until an r is found. We

13 We dropped the condition that $b^{n/d} + 1$ must be prime for this.

remark that if we do not assume r exists, then it is only possible to determine no r exists during the lifting procedure. This *ad hoc* recursion strategy allows us to bypass the need to factorise a at the onset.

Lemma 1. *Let $s, t \in \mathbb{Z}[X]$ be such that $sf + tg \in \mathbb{Z}$, with $\deg(s) < \deg(g)$ and $\deg(t) < \deg(f)$, and further assume that the greatest common divisor of s and t is 1. Then $a = sf + tg$ is a generator of the ideal $(f, g) \cap \mathbb{Z}$.*

Proof. We proceed by assuming $(f, g) \cap \mathbb{Z}$ is not generated by $sf + tg$ but some proper divisor and derive a contradiction.

For some prime factor p of $sf + gt$, we must have $(sf + gt)/p \in (f, g) \cap \mathbb{Z}$ and thus $(sf + gt)/p = s'f + t'g$ for some $s', t' \in \mathbb{Z}[X]$. We therefore have $sf + tg = ps'f + pt'g$, and rearranging gives $(s - ps')f = (pt' - t)g$. Since f and g are coprime, we must have $s - ps' = kg$ as well as $pt' - t = kf$ for some polynomial $k \in \mathbb{Z}[X]$.

Denote by $\bar{\cdot} : \mathbb{Z}[X] \to \mathbb{F}_p[X]$ the reduction modulo p map. Then $\bar{k}\bar{g} = \bar{s}$ and $\bar{k}\bar{f} = -\bar{t}$. The polynomial f is monic, so the assumption $\deg(t) < \deg(f)$ implies $\deg(\bar{t}) < \deg(\bar{f})$. Since $\mathbb{F}_p[X]$ is an integral domain, $\bar{k}\bar{f} = -\bar{t}$ can only hold if $\bar{k} = \bar{t} = 0$, which implies $\bar{s} = 0$. But $\bar{t} = \bar{s} = 0$ implies p divides both s and t, which contradicts the assumption that s and t have greatest common divisor 1. □

The question is thus how to find such s and t. One way to proceed is by computing, using the extended Euclidean algorithm over $\mathbb{Q}[X]$, rational polynomials s' and t' such that $s'f + t'g = 1$ and $\deg(s') < \deg(g)$ and $\deg(t') < \deg(f)$; then, multiplying by the lowest common multiple of all the denominators appearing in the coefficients of both s' and t', we find such s and t. The a we require is this lowest common multiple.

Next we show that, when it does not fail, we can use Euclid's algorithm to find r modulo a positive divisor of a. Thus we assume in the lemma that an r exists.

Lemma 2. *Let d be a positive divisor of a, and suppose that applying Euclid's algorithm to f and g in the ring $\mathbb{Z}_d[X]$ does not fail and outputs the polynomial ρ. Then $\rho \equiv r \bmod d$ up to units in $\mathbb{Z}_d[X]$.*

Proof. Denote by $\bar{\cdot}$ the reduction modulo d. Since $(f, g) = (a, r)$, we have $(\bar{f}, \bar{g}) = (\bar{a}, \bar{r}) = (\bar{r})$ since $d \mid a$. Now, by the properties of Euclid's algorithm, we have $(\bar{f}, \bar{g}) = (\rho)$. Therefore, $r \equiv \rho \bmod d$ up to a unit of $\mathbb{Z}_d[X]$. □

If d is taken to be a prime p, then Euclid's algorithm never fails, so we can use it to find a suitable r modulo p. However, it is possible that a larger power of the prime divides a, say p^e, and in this case, if Euclid's algorithm fails modulo p^e, we need to use Hensel lifting to lift ρ, our solution modulo p, to one modulo p^e. Algorithm 1 shows how to do this iteratively from p^j to p^{j+1}. It is at this point where a solution may fail to exist, showing that no such r exists.

Lemma 3. *Algorithm 1 for Hensel lifting is correct.*

Proof. Firstly, we assume that ρ' exists. By the preconditions, there exist α, β, and further μ and ν such that $\rho \equiv \alpha f + \beta g, f \equiv \mu\rho$ and $g \equiv \nu\rho$ modulo p^j, and we write each of these in p-ary form with the subscript indexing the digit, starting at zero. Note that α_0 and β_0 can be computed from f_0 and g_0 using the extended Euclidean algorithm over $\mathbb{F}_p[X]$. Also, μ and ν can easily be computed from f, g and ρ. Then $f - \rho\mu$ is divisible by p^j, so defining u via $f - \rho\mu = p^j u \bmod p^{j+1}$, ρ_j and μ_j must satisfy

$$0 \equiv f - (\rho + p^j\rho_j)(\mu + p^j\mu_j) \equiv p^j(u - (\rho_j\mu + \rho\mu_j)) \bmod p^{j+1},$$

or equivalently $\rho_j\mu + \rho\mu_j \equiv u \bmod p$. Hence, $u \in (\rho_0, \mu_0) = (\gamma)$, where γ is the greatest common divisor of ρ_0 and μ_0 in $\mathbb{F}_p[X]$, say with Bézout coefficients ξ and ζ so that $\gamma = \xi\rho_0 + \zeta\mu_0$. So γ divides u, and all solutions for ρ_j and μ_j are given by

$$\rho_j = \zeta\frac{u}{\gamma} - \kappa\frac{\rho_0}{\gamma} \quad \text{and} \quad \mu_j = \xi\frac{u}{\gamma} + \kappa\frac{\mu_0}{\gamma} \quad \text{for some } \kappa \in \mathbb{F}_p[X]. \tag{5.1}$$

The same computation for g implies that δ must divide ν, where $\delta = \phi\rho_0 + \psi\nu_0$ is the greatest common divisor of ρ_0 and ν_0 over $\mathbb{F}_p[X]$ and $\nu = (g - \rho\nu)/p^j \bmod p$. The solutions for ρ_j and ν_j are given by

$$\rho_j = \psi\frac{\nu}{\delta} - \lambda\frac{\rho_0}{\delta} \quad \text{and} \quad \nu_j = \phi\frac{\nu}{\delta} + \lambda\frac{\nu_0}{\delta} \quad \text{for some } \lambda \in \mathbb{F}_p[X]. \tag{5.2}$$

Input: Polynomials f, g, ρ in $\mathbb{Z}[X]$ (with f monic), a prime p and a positive integer j, satisfying $\alpha f + \beta g \equiv \rho \bmod p^j$ for some $\alpha, \beta \in \mathbb{Z}[X]$, as well as $f \equiv \rho\mu \bmod p^j$ and $g \equiv \rho\nu \bmod p^j$ for some $\mu, \nu \in \mathbb{Z}[X]$.
Output: A polynomial $\rho' \in \mathbb{Z}[X]$ such that $\rho' \equiv \alpha' f + \beta' g \bmod p^{j+1}$ for some $\alpha', \beta' \in \mathbb{Z}[X]$, as well as $\rho' \mid f$ and $\rho' \mid g$ in $\mathbb{Z}_{p^{j+1}}[X]$, or Fail if no such polynomial exists.

$\mu \leftarrow f/\rho$ ▷ Arithmetic in $\mathbb{Z}_{p^j}[X]$.
$\nu \leftarrow g/\rho$ ▷ Arithmetic in $\mathbb{Z}_{p^j}[X]$.
$u \leftarrow ((f - \rho\mu)/p^j) \bmod p$ ▷ Thus $f \equiv \rho\mu + p^j u \bmod p^{j+1}$.
$v \leftarrow ((g - \rho\nu)/p^j) \bmod p$ ▷ Thus $g \equiv \rho\nu + p^j v \bmod p^{j+1}$.
$\gamma, \xi, \zeta = \mathrm{xgcd}_{\mathbb{F}_p[X]}(\rho, \mu)$ ▷ Thus $\gamma = \xi\rho + \zeta\mu \bmod p$.
$\delta, \phi, \psi = \mathrm{xgcd}_{\mathbb{F}_p[X]}(\rho, \nu)$ ▷ Thus $\delta = \phi\rho + \psi\nu \bmod p$.
$\theta \leftarrow \zeta\psi(u\nu - v\mu) \bmod p$
$\rho_0 \leftarrow \rho \bmod p$
if $\gamma \nmid u$ **or** $\delta \nmid v$ **or** $\rho_0 \nmid \theta$ **then**
  return Fail
$\kappa \leftarrow (\theta/\rho_0 + \zeta\phi u - \psi\xi v)\tau$
$\rho_j \leftarrow (\zeta u - \kappa\rho_0)/\gamma \bmod \rho_0$ ▷ Hence $\deg(\rho_j) < \deg(\rho_0)$.
$\rho' \leftarrow \rho + p^j\rho_j$ ▷ Arithmetic in $\mathbb{Z}[X]$.
return ρ'

Algorithm 1: Hensel lifting.

Equating the two expression for ρ_j in equations (5.1) and (5.2), we see that $(\kappa\delta - \lambda\gamma)\rho_0 = \zeta u\delta - \psi v\gamma$. Now, using our expressions for γ and δ, we have $(\kappa\delta - \lambda\gamma)\rho_0 = (\zeta u\phi - \psi v\xi)\rho_0 + \zeta\psi(u\nu_0 - v\mu_0)$. Thus we must have that ρ_0 divides $\theta := \zeta\psi(u\nu_0 - v\mu_0)$ and then $\kappa\delta - \lambda\gamma = \zeta u\phi - \psi v\xi + \theta/\rho_0$.

Next we note that $\gcd(\gamma, \delta) = 1$ as otherwise there would be a non-trivial factor of μ_0 and ν_0, and then ρ_0 could not be the highest-degree common factor of f and g modulo p. Therefore, we can write $1 = \sigma\gamma + \tau\delta$ for some $\sigma, \tau \in \mathbb{F}_p[X]$, and all solutions for κ and λ are given by

$$\kappa = (\theta/\rho_0 + \zeta\phi u - \psi\xi v)\tau + \epsilon\gamma \quad \text{and} \quad \lambda = -(\theta/\rho_0 + \zeta\phi u - \psi\xi v)\sigma + \epsilon\delta \quad \text{for some } \epsilon \in \mathbb{F}_p[X],$$

and each such ϵ will give a valid solution. Algorithm 1 chooses to take $\epsilon = 0$ at first but implicitly changes its value later via modular reduction. We find ρ_j by plugging in the expression for κ in equation (5.1) then reducing modulo ρ_0. If this modular reduction subtracts $k\rho_0$, then this is equivalent to choosing $\epsilon = k$.

The post-conditions are satisfied because there is a solution for μ_j and ν_j whenever there is one for ρ_j. Setting $\mu' = \mu + \mu_j p^j$ and $\nu' = \nu + \nu_j p^j$, this shows that necessarily $\rho'\mu' = f$ and $\rho'\nu' = g$ in $\mathbb{Z}_{p^{j+1}}[X]$. Moreover, the requirement

$$\rho' = (\alpha + p^j\alpha_j)\rho'\mu' + (\beta + p^j\beta_j)\rho'\nu' \bmod p^{j+1}$$

is equivalent to $w + \alpha_0\mu_j + \alpha_j\mu_0 + \beta_0\nu_j + \beta_j\nu_0 = 0 \bmod p$, where $w = (\alpha\mu + \beta\nu - 1)/p^j \bmod p$, which always has a solution for α_j and β_j as μ_0 and ν_0 are coprime. Therefore, for any such solution, $\alpha' = \alpha + p^j\alpha_j$ and $\beta' = \beta + p^j\beta_j$ satisfy $\rho' = \alpha' f + \beta' g \bmod p^{j+1}$.

The proof up until this point shows that if a ρ_j exists, then Algorithm 1 finds one. Therefore, if the algorithm fails, such a ρ_j does not exist. □

Remark 1. The algorithm can be modified to avoid computing γ, ξ, ζ and δ, ϕ, ψ every iteration as these variables change only when p does. Also, it is possible to output α', β', μ' and ν' along with ρ', if required, but we opted here for brevity and simplicity.

One case where this additional output is useful is when computing inverses in R_g. This can be done in the same way in which r is computed, only replacing the inputs f and g by r and s and using the extended version of Euclid's algorithm; here $s \in R_g$ is the element to be inverted considered as an element of $\mathbb{Z}[X]$. Assuming this does not fail, this gives an expression of the form $h \equiv \alpha r + \beta s \bmod a$, and if s is invertible in R_g, then h will be an integer coprime with a, so by multiplying by a constant, we can assume $h = 1$; then β is the inverse.

Computing inverses is required in the ideal-NTRU problem, and this again shows that a factorisation of a is not needed to do this.

In practice, one will not check whether we are working modulo a prime, and the requirement that p is a prime in Algorithm 1 and Lemma 3 is there only to guarantee that the various calls to the Euclidean algorithm return a valid result and will not fail. In practice, if the Euclidean algorithm fails, it will be because it was unable to invert an integer modulo p, and hence we will have found a factor of p and can split it appropriately and try again on each factor until it succeeds.

In more detail, if one is working modulo a and finds a factor d, then one can find the largest power of d dividing a, say d^k. Then if a/d^k is coprime to d, we can work modulo a/d^k and d^k. Otherwise, $h = \gcd(a/d^k, d)$ is such that $1 < h < d$; then we find the largest power of h dividing d and the largest power of h dividing a/d^k, say h^l and h^m, respectively. Then h^{kl+m} divides a, and recurse using factors h^{kl+m}, $(d/h^l)^k$ and $a/(d^k h^m)$ until all factors are coprime. A solution modulo a is then found by using the Chinese remainder theorem, and this may result in a non-monic r if the degrees modulo each factor are different.

Our calculations (and some heuristics) suggest that $6/\pi^2 \approx 60.8\,\%$ of all random pairs f and g satisfy this condition, and that r is linear with overwhelming probability in this case. Of the remaining $39.2\,\%$, a little over $25\,\%$ give non-monic r, and in just under $14\,\%$ of the cases, no r exists. We leave open the question whether non-monic r can be useful in ways that a monic r cannot.

Finally, we note that we can use the fact that $|\mathrm{Res}(f, g)| = a^{\deg(r)}$ whenever such a monic r exists as a test of whether such an r exists. Compute a and $|\mathrm{Res}(f, g)|$, and test if the latter is an integer power of the former; if not, then we know that if an r exists, it will not be monic. As a small example, we can compute $\mathrm{Res}(X^4 + 1, X^3 + 4X + 1) = 306$, while $a = 102$ in this case, implying no monic r exists such that $(X^4 + 1, X^3 + 4X + 1) = (102, r(X))$; indeed $r(X) = 68X^2 + 101X + 19$ in this case.

Funding: This work was supported in part by the Research Council KU Leuven grants C14/18/067 and STG/17/019. Carl Bootland is funded by a FWO fellowship. Wouter Castryck is affiliated on a free basis with imec-COSIC at KU Leuven and with the Department of Mathematics: Algebra and Geometry at Ghent University. Alan Szepieniec was supported by an IWT doctoral grant.

References

[1] D. Aggarwal, A. Joux, A. Prakash and M. Santha, A new public-key cryptosystem via Mersenne numbers, Cryptology ePrint Archive (2017), https://eprint.iacr.org/2017/481/20170530:072202.

[2] D. Aggarwal, A. Joux, A. Prakash and M. Santha, A new public-key cryptosystem via Mersenne numbers, in: *Advances in Cryptology—CRYPTO 2018. Part III*, Lecture Notes in Comput. Sci. 10993, Springer, Cham, (2018), 459–482.

[3] M. Ajtai, Generating hard instances of lattice problems (extended abstract), in: *Proceedings of the 28th Annual ACM Symposium on the Theory of Computing*, ACM, New York (1996), 99–108.

[4] S. Akleylek, N. Bindel, J. Buchmann, J. Krämer and G. A. Marson, An efficient lattice-based signature scheme with provably secure instantiation, in: *Progress in Cryptology—AFRICACRYPT 2016*, Lecture Notes in Comput. Sci. 9646, Springer, Cham (2016), 44–60.

[5] N. Alamati, H. Montgomery, S. Patranabis and A. Roy, Minicrypt primitives with algebraic structure and applications, in: *Advances in Cryptology—EUROCRYPT 2019*, Springer, Cham (2019), 55–82.

[6] M. Albrecht, S. Bai and L. Ducas, A subfield lattice attack on overstretched NTRU assumptions: cryptanalysis of some FHE and graded encoding schemes, in: *Advances in Cryptology—CRYPTO 2016. Part I*, Lecture Notes in Comput. Sci. 9814, Springer, Berlin (2016), 153–178.

[7] E. Alkim, L. Ducas, T. Pöppelmann and P. Schwabe, Post-quantum key exchange—a new hope, in: *Proceedings of the 25th USENIX Security Symposium*, USENIX, Berkeley (2016), 327–343.

[8] D. J. Bernstein, C. Chuengsatiansup, T. Lange and C. van Vredendaal, NTRU prime: Reducing attack surface at low cost, in: *Selected Areas in Cryptography—SAC 2017*, Lecture Notes in Comput. Sci. 10719, Springer, Cham (2018), 235–260.

[9] M. Beunardeau, A. Connolly, R. Géraud and D. Naccache, On the hardness of the Mersenne low hamming ratio assumption, in: *Progress in Cryptology—LATINCRYPT 2017*, Lecture Notes in Comput. Sci. 11368, Springer, Cham (2019), 166–174.

[10] C. Bootland, W. Castryck, I. Iliashenko and F. Vercauteren, Efficiently processing complex-valued data in homomorphic encryption, Cryptology ePrint Archive (2018), https://eprint.iacr.org/2018/785.

[11] J. W. Bos, C. Costello, L. Ducas, I. Mironov, M. Naehrig, V. Nikolaenko, A. Raghunathan and D. Stebila, Frodo: Take off the ring! Practical, quantum-secure key exchange from LWE, in: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, ACM, New York (2016), 1006–1018.
[12] Z. Brakerski and N. Döttling, Two-message statistically sender-private OT from LWE, in: *Theory of Cryptography. Part II*, Lecture Notes in Comput. Sci. 11240, Springer, Cham (2018), 370–390.
[13] Z. Brakerski, C. Gentry and V. Vaikuntanathan, (Leveled) fully homomorphic encryption without bootstrapping, *ACM Trans. Comput. Theory* **6** (2014), no. 3, Article ID 13.
[14] Z. Brakerski, A. Langlois, C. Peikert, O. Regev and D. Stehlé, Classical hardness of learning with errors (extended abstract), in: *Proceedings of the 2013 ACM Symposium on Theory of Computing—STOC'13*, ACM, New York (2013), 575–584.
[15] W. Castryck, I. Iliashenko and F. Vercauteren, On error distributions in ring-based LWE, *LMS J. Comput. Math.* **19** (2016), 130–145.
[16] H. Chen, K. Laine, R. Player and Y. Xia, High-precision arithmetic in homomorphic encryption, in: *Topics in Cryptology—CT-RSA 2018*, Lecture Notes in Comput. Sci. 10808, Springer, Cham (2018), 116–136.
[17] J. H. Cheon, J. Jeong and C. Lee, An algorithm for NTRU problems and cryptanalysis of the GGH multilinear map without a low-level encoding of zero, *LMS J. Comput. Math.* **19** (2016), 255–266.
[18] M. Coglianese and B.-M. Goi, MaTRU: A new NTRU-based cryptosystem, in: *Progress in Cryptology—INDOCRYPT 2005*, Lecture Notes in Comput. Sci. 3797, Springer, Berlin (2005), 232–243.
[19] J. Ding, X. Xie and X. Lin, A simple provably secure key exchange scheme based on the learning with errors problem, Cryptology ePrint Archive (2012), https://eprint.iacr.org/2012/688.
[20] L. Ducas, V. Lyubashevsky and T. Prest, Efficient identity-based encryption over NTRU lattices, in: *Advances in Cryptology—ASIACRYPT 2014. Part II*, Lecture Notes in Comput. Sci. 8874, Springer, Heidelberg (2014), 22–41.
[21] J. Fan and F. Vercauteren, Somewhat practical fully homomorphic encryption, Cryptology ePrint Archive (2012), https://eprint.iacr.org/2012/144.
[22] C. Gentry, Fully homomorphic encryption using ideal lattices, in: *Proceedings of the 2009 ACM International Symposium on Theory of Computing—STOC'09*, ACM, New York (2009), 169–178.
[23] C. Gentry, C. Peikert and V. Vaikuntanathan, Trapdoors for hard lattices and new cryptographic constructions [extended abstract], in: *Proceedings of the 40th Annual ACM Symposium on Theory of Computing—STOC'08*, ACM, New York (2008), 197–206.
[24] C. Gu, Integer version of ring-LWE and its applications, Cryptology ePrint Archive (2017), https://eprint.iacr.org/2017/641.
[25] M. Hamburg, Post-quantum cryptography proposal: ThreeBears, 2018.
[26] J. Hoffstein, J. Pipher and J. H. Silverman, NTRU: A new high speed public key cryptosystem, (1996), https://web.securityinnovation.com/hubfs/files/ntru-orig.pdf.
[27] J. Hoffstein, J. Pipher and J. H. Silverman, NTRU: A ring-based public key cryptosystem, in: *Algorithmic Number Theory*, Lecture Notes in Comput. Sci. 1423, Springer, Berlin (1998), 267–288.
[28] J. Hoffstein, J. Pipher, W. Whyte and Z. Zhang, A signature scheme from learning with truncation, Cryptology ePrint Archive (2017), https://eprint.iacr.org/2017/995.
[29] P. Kirchner and P.-A. Fouque, Revisiting lattice attacks on overstretched NTRU parameters, in: *Advances in Cryptology—EUROCRYPT 2017. Part I*, Lecture Notes in Comput. Sci. 10210, Springer, Cham (2017), 3–26.
[30] A. Langlois and D. Stehlé, Worst-case to average-case reductions for module lattices, *Des. Codes Cryptogr.* **75** (2015), no. 3, 565–599.
[31] S. Ling, K. Nguyen, D. Stehlé and H. Wang, Improved zero-knowledge proofs of knowledge for the ISIS problem, and applications, in: *Public-key Cryptography—PKC 2013*, Lecture Notes in Comput. Sci. 7778, Springer, Heidelberg (2013), 107–124.
[32] A. López-Alt, E. Tromer and V. Vaikuntanathan, On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption, in: *Proceedings of the 2012 ACM Symposium on Theory of Computing—STOC'12*, ACM, New York (2012), 1219–1234.
[33] V. Lyubashevsky, Lattice signatures without trapdoors, in: *Advances in Cryptology—EUROCRYPT 2012*, Lecture Notes in Comput. Sci. 7237, Springer, Heidelberg (2012), 738–755.
[34] V. Lyubashevsky and D. Micciancio, Generalized compact knapsacks are collision resistant, in: *Automata, Languages and Programming. Part II*, Lecture Notes in Comput. Sci. 4052, Springer, Berlin, (2006), 144–155.
[35] V. Lyubashevsky, C. Peikert and O. Regev, On ideal lattices and learning with errors over rings, in: *Advances in Cryptology—EUROCRYPT 2010*, Lecture Notes in Comput. Sci. 6110, Springer, Berlin (2010), 1–23.
[36] V. Lyubashevsky, C. Peikert and O. Regev, On ideal lattices and learning with errors over rings, in: *Advances in Cryptology—EUROCRYPT 2010*, Lecture Notes in Comput. Sci. 6110, Springer, Berlin (2010), 1–23.
[37] R. J. McEliece, A public-key cryptosystem based on algebraic coding theory, *JPL DSN Progress Report* **42–44** (1978), 114–116.
[38] B. Mi, D. Huang, S. Wan, L. Mi and J. Cao, Oblivious transfer based on NTRUEncrypt, *IEEE Access* **6** (2018), 35283–35291.
[39] D. Micciancio, Generalized compact knapsacks, cyclic lattices, and efficient one-way functions, *Comput. Complexity* **16** (2007), no. 4, 365–411.
[40] D. Micciancio, On the hardness of learning with errors with binary secrets, *Theory Comput.* **14** (2018), Article ID 13.

[41] G. Myerson, On resultants, *Proc. Amer. Math. Soc.* **89** (1983), no. 3, 419–420.
[42] R. Nayak, C. V. Sastry and J. Pradhan, A matrix formulation for NTRU cryptosystem, in: *16th IEEE International Conference on Networks*, IEEE Press, Piscataway (2008), 1–5.
[43] H. Niederreiter, Knapsack-type cryptosystems and algebraic coding theory, *Probl. Control Inf. Theory* **15** (1986), no. 2, 159–166.
[44] C. Peikert, V. Vaikuntanathan and B. Waters, A framework for efficient and composable oblivious transfer, in: *Advances in Cryptology—CRYPTO 2008*, Lecture Notes in Comput. Sci. 5157, Springer, Berlin (2008), 554–571.
[45] E. Prange, The use of information sets in decoding cyclic codes, *IRE Trans.* **IT-8** (1962), S5–S9.
[46] O. Regev, On lattices, learning with errors, random linear codes, and cryptography, in: *Proceedings of the 37th Annual ACM Symposium on Theory of Computing—STOC'05*, ACM, New York (2005), 84–93.
[47] P. Santini, E. Persichetti and M. Baldi, Reproducible codes and cryptographic applications, Cryptology ePrint Archive (2018), https://eprint.iacr.org/2018/666.
[48] D. Stehlé, R. Steinfeld, K. Tanaka and K. Xagawa, Efficient public key encryption based on ideal lattices (extended abstract), in: *Advances in Cryptology—ASIACRYPT 2009*, Lecture Notes in Comput. Sci. 5912, Springer, Berlin (2009), 617–635.
[49] A. Szepieniec, Ramstake, Technical report, National Institute of Standards and Technology, 2018.
[50] NIST. Post-quantum crypto standardization, 2018.
[51] NIST. Submission to the NIST call for PQC proposals, 2018.