



**HAL**  
open science

# Trapdoors in lattice-based cryptography: applications and implementation

Lucas Prabel

► **To cite this version:**

Lucas Prabel. Trapdoors in lattice-based cryptography: applications and implementation. Other [cs.OH]. Université de Rennes, 2023. English. NNT : 2023URENS035 . tel-04801186

**HAL Id: tel-04801186**

**<https://theses.hal.science/tel-04801186v1>**

Submitted on 25 Nov 2024

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# THÈSE DE DOCTORAT DE

L'UNIVERSITÉ DE RENNES

ÉCOLE DOCTORALE N° 601

*Mathématiques, Télécommunications, Informatique, Signal, Systèmes,  
Électronique*

Spécialité : *Informatique*

Par

**Lucas PRABEL**

## Trapdoors in Lattice-Based Cryptography : Applications and Implementation

Thèse présentée et soutenue à Rennes, le 5 octobre 2023

Unité de recherche : IRISA, UMR 6074

### Rapporteurs avant soutenance :

Sébastien CANARD Professeur, Télécom Paris  
Olivier BLAZY Professeur, Ecole Polytechnique

### Composition du Jury :

*Attention, en cas d'absence d'un des membres du Jury le jour de la soutenance, la composition du jury doit être revue pour s'assurer qu'elle est conforme et devra être répercutée sur la couverture de thèse*

Examineurs :	Sébastien CANARD	Professeur, Télécom Paris
	Olivier BLAZY	Professeur, Ecole Polytechnique
	Caroline FONTAINE	Directrice de recherche CNRS, LMF ENS Paris-Saclay
	Thomas RICOSSET	Ingénieur Cryptographe, Thales
Dir. de thèse :	Pierre-Alain FOUQUE	Professeur, Université Rennes
	Adeline ROUX-LANGLOIS	Chargée de recherche, CNRS, GREYC Caen
Encadr. de thèse :	Malika IZABACHÈNE	



# ACKNOWLEDGEMENT

---

Many people have accompanied me in one way or another during this thesis, by their presence or their discussion, and I wish to thank them all for having made these years particularly agreeable to live.

First, I want to thank my supervisors, Adeline Roux-Langlois, Pierre-Alain Fouque and Malika Izabachène for their support and guidance over the past three years, allowing me to evolve in a very pleasant working environment. I would also like to express my gratitude to all my co-authors who have contributed to the success of this thesis.

I am grateful to Sébastien Canard and Olivier Blazy, for their patience in reading my manuscript and for their thoughtful comments. I would also like to thank Caroline Fontaine and Thomas Ricosset for agreeing to serve on the jury of my PhD defense.

These years have been made all the more enjoyable by the presence of all the members of the EMSEC team first, and then the CAPSULE and SPICY teams. From PhD students and post-docs to permanent researchers and interns, I would like to thank them all (without naming them, so as not to forget any!) for the scientific and informal discussions, the coffee breaks, the lunches, the drinks exchanged, and so on. I also have a thought towards the Reading Circle organized by members of the Gender Equality Task Team, with whom I had very enriching discussions. The presence and cheerfulness of all these people had a positive effect and left a definite mark on my memory of this adventure. I now wish the new arrivals many happy years among those teams in the future.

I also have fond memories of all the people I met at conferences and on other trips, with whom I got to know and who left their mark on these events.

I'm also very grateful to all the laboratory staff, the administrative team, and especially the team assistants for their kindness.

Taking a look further back, I'd like to express my heartfelt gratitude to Mr Sougné and Mr Turrel who gave me a thirst for learning and aroused my curiosity, leaving an impression that is still with me today.

Finally, I want to finish my acknowledgments by expressing my great gratitude towards the people who were close to me in the last few years, family and friends, for their unfailing support and love.



# TABLE OF CONTENTS

---

<b>Résumé en français</b>	<b>13</b>
<b>Introduction</b>	<b>29</b>
<b>1 Preliminaries</b>	<b>43</b>
1.1 Notations . . . . .	44
1.2 Lattices . . . . .	45
1.2.1 First Definitions . . . . .	45
1.2.2 Structured Lattices . . . . .	48
1.2.3 Computational Problems . . . . .	48
1.3 Probabilities . . . . .	54
1.3.1 Basic Definitions . . . . .	54
1.3.2 Discrete Gaussians . . . . .	55
1.3.3 Leftover Hash Lemma . . . . .	59
1.4 Cryptographic Primitives . . . . .	59
1.4.1 Basic Public-Key Primitives . . . . .	60
1.4.2 More advanced primitives . . . . .	64
<b>2 Lattice Trapdoors on Modules: Implementation and Applications</b>	<b>71</b>
2.1 Introduction . . . . .	72
2.1.1 Presentation . . . . .	72
2.1.2 Contributions: A Technical Overview . . . . .	73
2.2 Gaussian Preimage Sampling on Module Lattices . . . . .	76
2.2.1 Module G-Lattice . . . . .	77
2.2.2 G-Sampling . . . . .	79
2.2.3 Perturbation Sampling . . . . .	80
2.2.4 Sampling Gaussian Preimages . . . . .	86
2.2.5 Implementation and Performance . . . . .	86
2.3 A GPV Signature Scheme on Modules . . . . .	88
2.3.1 The Scheme . . . . .	88

TABLE OF CONTENTS

---

2.3.2	Parameters and Security . . . . .	90
2.3.3	Implementation and Performance . . . . .	91
2.4	A Standard Model Signature Scheme on Modules . . . . .	93
2.4.1	The Scheme . . . . .	93
2.4.2	Parameters and Security . . . . .	94
2.4.3	Implementation and Performance . . . . .	95
2.5	An Identity-Based Encryption Scheme on Modules . . . . .	96
2.5.1	Implementation and Performance . . . . .	98
<b>3</b>	<b>Identity-Based Encryption Scheme Using Approximate Trapdoors</b>	<b>99</b>
3.1	Introduction . . . . .	100
3.1.1	Presentation . . . . .	100
3.1.2	Related Work . . . . .	100
3.1.3	Contributions: A Technical Overview . . . . .	102
3.2	Approximate ISIS Problem . . . . .	102
3.2.1	Definitions . . . . .	102
3.2.2	Hardness of AISIS . . . . .	103
3.3	Approximate Gadget Trapdoors . . . . .	104
3.3.1	Approximate Gadget Lattices . . . . .	104
3.3.2	Approximate F-Sampling . . . . .	105
3.3.3	Approximate Perturbation Sampling . . . . .	106
3.3.4	Sampling Approximate Gaussian Preimages . . . . .	106
3.4	Application to an Identity-Based Encryption Scheme . . . . .	107
3.4.1	The Scheme . . . . .	108
3.4.2	Correctness . . . . .	108
3.4.3	Parameters and Security . . . . .	109
3.4.4	Implementation . . . . .	112
3.5	Performance and Comparison . . . . .	113
<b>4</b>	<b>Improving Efficiency Using NTRU Lattices</b>	<b>115</b>
4.1	Introduction . . . . .	116
4.1.1	Presentation . . . . .	116
4.2	The DLP-IBE Scheme . . . . .	117
4.2.1	The Scheme . . . . .	117
4.2.2	Parameters and Security . . . . .	120

4.3	Solving the NTRU Equation . . . . .	121
4.3.1	HNF Method . . . . .	122
4.3.2	Resultant Method . . . . .	123
4.3.3	Using the Field Norm for Power-of-Two Cyclotomic Fields . . . . .	125
4.3.4	Tools for More General Fields . . . . .	127
4.4	A New Assumption: iNTRU . . . . .	128
4.4.1	Definition . . . . .	129
4.4.2	Hardness of iNTRU . . . . .	129
4.4.3	Approximate Trapdoors Based on iNTRU . . . . .	130
4.5	Application to a NTRU-Based IBE Scheme . . . . .	131
4.5.1	The Scheme . . . . .	132
4.5.2	Correctness . . . . .	132
4.5.3	Parameters and Security . . . . .	133
4.5.4	Cryptanalysis in the Overstretched Regime. . . . .	136
4.5.5	Implementation and Performance . . . . .	144
	<b>Conclusion</b>	<b>149</b>
	<b>Bibliography</b>	<b>153</b>





# NOTATIONS

---

This thesis employs a considerable number of notations, and in order to help the reader's comprehension, the following table provides a summary of all the notations that are used.

Notation	Meaning
$\mathbb{N}$	Set of natural numbers
$\mathbb{Z}$	Ring of integers
$\mathbb{Q}$	Field of rational numbers
$\mathbb{R}$	Field of real numbers
$\mathbb{C}$	Field of complex numbers
$\mathbb{Z}_q$	Ring of integers modulo $q \in \mathbb{N}$
$\lambda$	The security parameter
$\lfloor x \rfloor$	Floor of $x$
$\lceil x \rceil$	Ceil of $x$
$\llbracket x \rrbracket$	Closest integer to $x$
$\log$	The natural logarithm
$\log_b$	The logarithm to the base $b$
$O, \Omega, \Theta, o, \omega, \sim$	Asymptotic notations
$\ \cdot\ _2, \ \cdot\ $	The Euclidean norm
$\ \cdot\ _\infty$	The infinity norm
$\langle \cdot, \cdot \rangle$	The Euclidean inner product
$\otimes$	The Kronecker product
$\mathbf{A}^T$	The transpose of the matrix $\mathbf{A}$
$s_1(\mathbf{A})$	The largest singular value of the matrix $\mathbf{A}$
$\tilde{\mathbf{B}}$	The Gram-Schmidt Orthogonalization (GSO) of $\mathbf{B}$
$\mathbf{I}_n$	The identity matrix of size $n$
$\Lambda$	A lattice
$\Lambda_q^\perp$	The orthogonal lattice of $\Lambda \pmod q$
$\mathcal{L}(\mathbf{B})$	The lattice spanned by the vectors of $\mathbf{B}$

TABLE OF CONTENTS

---

$\lambda_1(\Lambda)$	The length of a shortest non-zero vector in $\Lambda$
$\det(\Lambda)$	The determinant of $\Lambda$
$\rho_{\sigma, \mathbf{c}}$	The (spherical) Gaussian function of standard deviation $\sigma$ , centered on $\mathbf{c}$
$D_{\Lambda, \sigma, \mathbf{c}}$	The discrete (spherical) Gaussian distribution of standard deviation $\sigma$ and center $\mathbf{c}$ , over the lattice $\Lambda$
$\rho_{\sqrt{\Sigma}, \mathbf{c}}$	The (ellipsoidal) Gaussian function of covariance matrix $\Sigma$ , centered on $\mathbf{c}$
$D_{\Lambda, \sqrt{\Sigma}, \mathbf{c}}$	The discrete (ellipsoidal) Gaussian distribution of covariance matrix $\Sigma$ and center $\mathbf{c}$ , over the lattice $\Lambda$
$\mathcal{U}(\mathcal{S})$	The uniform distribution over a set $\mathcal{S}$
$x \leftarrow \chi$	$x$ sampled from the distribution $\chi$
$\Delta(P, Q)$	The statistical distance between distributions $P$ and $Q$
$\zeta_m$	A primitive $m$ -th root of unity
$\phi_m$	The $m$ -th cyclotomic polynomial

Many acronyms are also used, listed in the following table.

Notation	Meaning
SVP	Shortest Vector Problem
CVP	Closest Vector Problem
SIS	Short Integer Solution
ISIS	Inhomogeneous Short Integer Solution
AISIS	Approximate Inhomogeneous Short Integer Solution
Ring-SIS	Ring Short Integer Solution
Module-SIS	Module Short Integer Solution
LWE	Learning With Errors
Ring-LWE	Ring Learning With Errors
Module-LWE	Module Learning With Errors
PPT	Probabilistic Polynomial Time
GSO	Gram-Schmidt Orthogonalization
FRD	Full-Rank Differences (encoding)
LHL	Leftover Hash Lemma

FFT	Fast Fourier Transform
NTT	Number Theoretic Transform
IBE	Identity-Based Encryption
ROM	Random Oracle Model
EUF – CMA	Existential Unforgeability under an Adaptive Chosen-Message Attack
IND – CPA	Indistinguishability of Ciphertexts under Chosen-Plaintext Attack
IND – sID – CPA	Indistinguishability of Ciphertexts under a Selective-Identity Chosen-Plaintext Attack
IND – ID – CPA	Indistinguishability of Ciphertexts under an Adaptive-Identity Chosen-Plaintext Attack



# RÉSUMÉ EN FRANÇAIS

---

**C**RYPTOLOGIE signifie littéralement la *science des secrets*. En effet, le mot vient du grec ancien, *κρυπτός* signifiant *secret* et *λόγος* signifiant *science*. Il englobe à la fois la *cryptographie*, dont l'objet d'étude est de protéger des messages en les rendant secrets, et la *cryptanalyse* qui, au contraire, cherche à attaquer ces protections pour trouver les messages originaux cachés derrière ces secrets. Cependant, les prérogatives de la cryptographie se sont étendues bien au-delà de l'objectif initial de rendre un message secret (*confidentialité*), puisqu'elle s'intéresse également à d'autres aspects de la sécurité, tels que l'*intégrité* (dont l'objectif est de traiter l'altération des données) ou l'*authentification* (dont l'objectif est de pouvoir tracer l'origine d'un message reçu). La cryptographie est aujourd'hui une science riche, dont les objectifs sont vastes et qui repose sur des bases scientifiques solides.

## Un Bref Historique

Mais l'histoire de la cryptographie est longue. Dans *The Codebreakers*, publié en 1967, David Kahn retrace cette histoire, en remontant jusqu'à ses premières traces d'utilisation en Égypte ancienne, il y a 4000 ans, où des substitutions de symboles hiéroglyphiques gravées dans la chambre principale de la tombe d'un noble par son scribe ont été observées. Au cours de ses premiers 3000 ans d'existence, le champ de la cryptologie n'a pas connu de croissance régulière. Au contraire, il est apparu de manière indépendante dans diverses régions du monde, mais les progrès sont restés lents et sporadiques. Ce n'est qu'avec l'avènement de la Renaissance occidentale que les connaissances accumulées en cryptologie commencèrent à légèrement prendre de l'ampleur.

Avant cela, nous pouvons souligner quelques exemples d'applications pratiques de techniques cryptographiques simples.

Le chiffre de César, nommé d'après Jules César et également appelé chiffrement par décalage, est l'une des plus anciennes et plus simples techniques de chiffrement ayant été utilisées. Il a été utilisé à des fins militaires, aux côtés d'autres méthodes plus sophistiquées, durant la Rome antique. Il s'agit d'une méthode cryptographique basée sur

la substitution monoalphabétique. Le chiffré d'un message est calculé en remplaçant les lettres du texte en clair par des lettres situées  $k$  positions plus loin dans l'alphabet (et en recommençant depuis le début de l'alphabet si l'on arrive à la lettre Z), où  $1 \leq k \leq 25$  est un nombre entier fixe, représentant la clef secrète du schéma de chiffrement.

A elle seule, cette méthode n'est pas sécurisée, en raison du très petit nombre de clefs possibles, ce qui permet, lorsque la méthode de chiffrement est connue, de toutes les essayer. De plus, comme tout chiffrement basé sur la substitution monoalphabétique, elle peut être très rapidement "cassée" par l'analyse de la fréquence des lettres apparaissant dans les messages chiffrés.

Le chiffrement de Vigenère, décrit pour la première fois par Blaise de Vigenère en 1586, est un chiffrement de substitution polyalphabétique qui utilise de manière répétée le chiffrement de César avec un décalage différent selon la position de la lettre dans le message. Le rang de ce décalage est donné par un autre texte qui sert de clef secrète et que l'on peut réutiliser en boucle suivant la longueur du message à chiffrer. Par conséquent, les mêmes lettres du texte en clair peuvent, en fonction de leur position, être remplacées par des lettres différentes dans le message chiffré, contrairement aux systèmes de chiffrement mono-alphabétiques qui remplacent systématiquement une lettre donnée par la même lettre. Cette méthode est donc résistante aux attaques basiques par analyse de fréquence. Toutefois, si l'on devine correctement la longueur de clef  $n$  du chiffrement de Vigenère, le texte chiffré peut alors être considéré comme  $n$  chiffrés de César entrelacés, qui peuvent facilement être cassés individuellement.

Mais le domaine de la cryptographie a réellement évolué et pris la forme et le formalisme que nous lui connaissons aujourd'hui à la fin du XIX<sup>e</sup> et durant tout le XX<sup>e</sup> siècle.

À la fin du XX<sup>e</sup> siècle, Auguste Kerckhoffs expose dans son livre *La cryptographie militaire* [Ker83] une liste de principes à respecter pour concevoir de bons systèmes de chiffrements dans un contexte militaire. L'un des plus importants de ces principes, aujourd'hui appelé *principe de Kerckhoffs*, stipule que "la méthode de chiffrement ne doit pas avoir besoin d'être secrète et doit pouvoir tomber entre les mains de l'ennemi sans inconvénient". Fondamentalement, cela signifie que la sécurité d'un système cryptographique ne doit pas reposer sur le caractère secret de la conception de ce système, mais sur le caractère secret de la clef, qui représente l'élément crucial du système et qui est gardé confidentiel et connu uniquement par une ou plusieurs personnes autorisées. Aujourd'hui, le principe de Kerckhoffs est largement reconnu et généralement interprété comme défendant l'idée que l'ensemble de la conception d'un cryptosystème doit être rendu public. Cela contraste

fortement avec le concept de "sécurité par l'obscurité", qui cherche à garder les algorithmes secrets afin d'en assurer leur sécurité.

Au cours du XX<sup>e</sup> siècle, les deux guerres mondiales ont joué un rôle important dans le développement de la cryptographie et de la cryptanalyse. Alan Turing, réputé pour sa conception mathématique des premiers modèles d'ordinateurs, a participé à leur réalisation pendant la Seconde Guerre mondiale et a joué un rôle majeur dans la cryptanalyse de la machine Enigma utilisée par les armées nazies. L'essor des ordinateurs et des systèmes de communication au cours des années 1960 a entraîné un besoin croissant d'outils numériques pour protéger les informations et de services garantissant la sécurité. Cette demande a entraîné un large éventail de développements dans le domaine de la cryptographie, en commençant par les travaux de Feistel chez IBM au cours des années 1970, aboutissant finalement à l'adoption du *Data Encryption Standard* (DES), un algorithme de chiffrement à clef symétrique publié en tant que standard par les *Federal Information Processing Standard* (FIPS) pour les États-Unis en 1977. Par la suite, la recherche universitaire dans le domaine de la cryptographie va considérablement s'intensifier. En conséquence, une théorie complète a commencé à se développer, rendant possible une étude rigoureuse de la cryptographie en tant que domaine de recherche scientifique et de discipline mathématique.

## Cryptographie à Clef Publique

Avec la *cryptographie à clef secrète* (ou *symétrique*), l'utilisation d'une clef secrète commune entre l'expéditeur et le destinataire est nécessaire, ce qui exige une rencontre physique entre les parties au préalable, ou qu'elles fassent appel à un tiers de confiance. Cependant, les travaux de Diffie et Hellman [DH76] en 1976 vont changer la donne. Dans leur article intitulé *New directions in Cryptography*, ils introduisent le concept de *cryptographie à clef publique* (ou *asymétrique*). Chaque utilisateur peut désormais calculer une clef accessible au public, appelée *clef publique*, et sa clef privée associée, qu'il est le seul à connaître. Dans un système de chiffrement à clef publique, par exemple, la clef publique d'un utilisateur permet à n'importe qui de chiffrer un message, mais seul le propriétaire de la clef privée correspondante sera en mesure de le déchiffrer. Ce nouveau concept permet donc de résoudre le problème de la rencontre physique préalable mentionné ci-dessus pour la cryptographie symétrique. De nos jours, dans le domaine des communications, un système de cryptographie à clef publique est souvent utilisé conjointement avec un



système à clef privée : le premier permet aux deux parties qui souhaitent communiquer de se mettre d'accord sur une clef secrète commune et le second est utilisé pour le reste des communications. En effet, les systèmes de cryptographie à clef privée sont généralement plus efficaces que leurs homologues à clef publique, ce qui rend leur utilisation plus intéressante pour les protocoles cryptographiques.

Dans [DH76], Diffie et Hellman fournissent également une nouvelle méthode d'échange de clefs, dont la sécurité est basée sur la difficulté du problème du logarithme discret. L'échange de clefs est un protocole qui aide deux parties à se mettre d'accord sur une clef secrète commune, sans qu'il soit nécessaire d'échanger au préalable des informations privées. Dans leur article original, les paramètres publics de leur protocole d'échange de clefs sont un groupe cyclique  $\mathbb{G}$ , son ordre  $q$  et un générateur  $g \in \mathbb{G}$ . Le principe du protocole est alors le suivant. Alice choisit  $a \in \mathbb{Z}_q$  uniforme et calcule  $g^a$ . Elle envoie cette quantité à Bob. Ce dernier choisit également  $b \in \mathbb{Z}_q$  uniforme et calcule  $g^b$ , qu'il envoie à Alice. Alice et Bob peuvent alors calculer  $g^{ab} = (g^a)^b = (g^b)^a$ , qui constitue leur clef secrète commune.

Bien que Diffie et Hellman n'aient pas présenté de système de chiffrement à clef publique dans leur article, en 1978, Rivest, Shamir et Adleman ont décrit dans [RSA78] les premières réalisations pratiques d'un système de chiffrement et de signature à clef publique. Cette fois, la sécurité des schémas RSA repose sur le problème de la factorisation d'un produit de grands nombres entiers. Quelques années plus tard, dans [Gam84], El Gamal présente un autre système de chiffrement à clef publique, basé sur le problème du logarithme discret.

En cryptographie à clef publique, la sécurité des protocoles est donc basée sur la difficulté supposée de certains problèmes mathématiques, tels que les problèmes de factorisation de grands nombres entiers ou le problème du logarithme discret. Ces deux problèmes spécifiques sont omniprésents dans la cryptographie classique à clef publique. Cependant, les protocoles cryptographiques reposant sur ceux-ci sont menacés par la possible émergence de l'ordinateur quantique. En effet, il existe une importante différence de nature entre les opérations effectuées par les ordinateurs quantiques et les ordinateurs classiques. En particulier, les ordinateurs quantiques fonctionnent avec des *qubits* (qui sont la version quantique du bit binaire classique) qui possèdent un ensemble infini de valeurs alternatives en plus de 0 et 1. Cela permet aux ordinateurs quantiques d'être beaucoup plus rapides que les ordinateurs classiques pour certains problèmes spécifiques. En effet, Shor a montré dans [Sho94] comment un algorithme quantique pouvait résoudre efficacement

les problèmes du logarithme discret et de la factorisation, en temps polynomial.

Cependant, les ordinateurs quantiques n'existent pas encore à grande échelle et ne sont pas encore assez puissants. Alors que les ordinateurs quantiques actuels sont supposés pouvoir gérer jusqu'à quelques centaines de qubits, on estime que plus d'un million sont nécessaires pour résoudre le problème de la factorisation. Néanmoins, les progrès réalisés dans le domaine de l'informatique quantique sont constants et il est important d'envisager des alternatives pour un avenir où les ordinateurs quantiques deviendront largement accessibles. En outre, il existe un risque que des personnes malveillantes collectent aujourd'hui des messages chiffrés dans l'intention de les déchiffrer ultérieurement, une fois que les ordinateurs quantiques seront disponibles. Ce risque lié au possible avènement de ces machines a incité la communauté scientifique à rechercher de nouvelles hypothèses de sécurité, potentiellement résistantes face à cette menace, qui permettraient d'élaborer des systèmes qui resteraient sécurisés y compris face à des adversaires disposant de tels ordinateurs. Il s'agit d'un domaine de recherche actif connu sous le nom de *cryptographie post-quantique*.

## Cryptographie Post-Quantique

C'est avec cette perspective en tête qu'en 2016, le National Institute of Standards and Technology (NIST), une agence du Département du commerce des États-Unis, a lancé le programme et compétition "Post-Quantum Cryptography Standardization", dont l'objectif était d'établir de nouveaux standards de schémas cryptographiques post-quantiques. Les premières soumissions de la communauté scientifique à cette compétition comprenaient 23 schémas de signature et 59 schémas de chiffrement/de mécanisme d'encapsulation de clés. Les schémas cryptographiques candidats étaient basés sur une grande variété de problèmes supposés difficiles à attaquer efficacement par un ordinateur quantique. Aujourd'hui, on distingue essentiellement 5 familles principales qui composent le paysage de la cryptographie post-quantique. Ces familles reposent sur la difficulté de problèmes définis sur des réseaux euclidiens, des codes correcteurs d'erreurs, des isogénies de courbes elliptiques, des polynômes multivariés et des fonctions de hachage.

Dans cette thèse, nous nous intéresserons à la cryptographie post-quantique basée sur des problèmes mathématiques définis sur des réseaux euclidiens.

## Cryptographie Basée sur les Réseaux Euclidiens

La cryptographie basée sur les réseaux euclidiens est l'une des solutions les plus prometteuses contre la menace quantique. Ce domaine de recherche a été très actif depuis les travaux d'Ajtai [Ajt96], qui introduit le problème Short Integer Solution (SIS) et de Regev [Reg05], qui introduit le problème Learning With Errors (LWE). Dans ces articles, d'intéressantes réductions pire-cas moyen-cas sont prouvées, ce qui a attiré de nombreux chercheurs à s'intéresser au domaine de la cryptographie basée sur les réseaux euclidiens. Fondamentalement, ces réductions indiquent que si la sécurité d'un cryptosystème est basée sur un problème moyen-cas (tel que SIS ou LWE), attaquer des instances aléatoires de ce cryptosystème est au moins aussi difficile que résoudre toutes les instances d'un problème pire-cas défini sur les réseaux euclidiens. Ainsi, les problèmes moyen-cas tels que SIS et LWE sont particulièrement bien adaptés à la conception de systèmes cryptographiques. Ces réductions fournissent donc de solides garanties de sécurité pour la cryptographie basée sur les réseaux euclidiens. En outre, d'autres schémas efficaces ont fondé leur sécurité sur des problèmes moyen-cas définis sur des réseaux euclidiens structurés, tels que le schéma de chiffrement NTRU, qui a été introduit dans [HPS98].

**Réseaux Euclidiens.** Un réseau euclidien peut être considéré comme un arrangement périodique de points formant une "grille" (voir Figure 1). Plus formellement, étant donné un ensemble de vecteurs linéairement indépendants  $\mathbf{B} = \{\mathbf{b}_1, \dots, \mathbf{b}_k\} \subset \mathbb{R}^m$ , nous appelons réseau euclidien de base  $\mathbf{B}$  et de rang  $k$  l'ensemble suivant :

$$\mathcal{L}(\mathbf{B}) := \left\{ \sum_{i=1}^k \lambda_i \mathbf{b}_i, \lambda_i \in \mathbb{Z} \right\}.$$

Lorsque  $k = m$ , le réseau euclidien est dit de rang plein. Un réseau euclidien  $\Lambda$  admet une infinité de bases pour  $m \geq 2$ .

De nombreux problèmes algorithmiques liés aux réseaux euclidiens ont été introduits et largement étudiés. L'un d'entre eux est le Shortest Vector Problem (SVP). Ce problème consiste, à partir d'une base  $\mathbf{B}$  d'un réseau euclidien  $\Lambda$ , à trouver le plus court vecteur non nul de ce réseau euclidien (voir Figure 2). On note  $\lambda_1(\Lambda)$  la norme euclidienne d'un plus petit vecteur non nul de  $\Lambda$ . Une variante approchée de ce problème, notée  $\text{SVP}_\gamma$ , consiste à trouver  $\mathbf{x} \in \Lambda$  de sorte que  $\|\mathbf{x}\| \leq \gamma \cdot \lambda_1(\Lambda)$ .

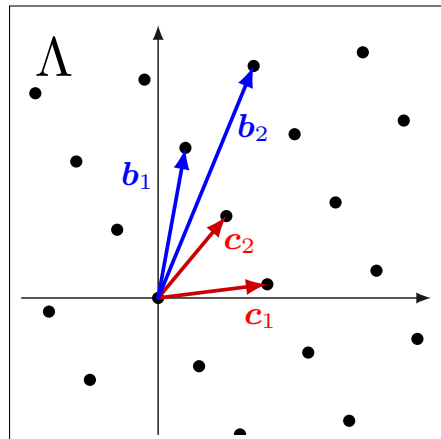


Figure 1 – Un réseau euclidien  $\Lambda$  de dimension 2 avec deux de ses bases :  $B = (b_1, b_2)$  et  $B' = (c_1, c_2)$ .

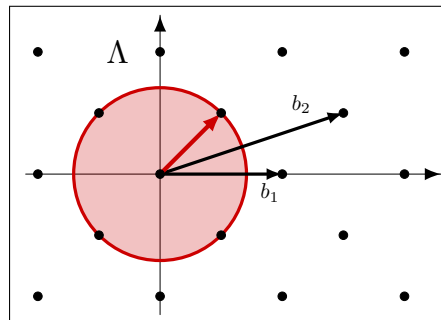


Figure 2 – Un réseau euclidien  $\Lambda$  de dimension 2 avec une base  $B = (b_1, b_2)$  et un **plus court vecteur**.

Ces problèmes sont supposés être difficiles à résoudre. Les meilleurs algorithmes à ce jour, LLL [LLL82] et BKZ [SE94], sont basés sur la réduction de réseaux euclidiens. L'algorithme LLL, introduit par Lenstra, Lenstra et Lovász en 1982, résout le problème  $SVP_\gamma$  en temps polynomial lorsque  $\gamma$  est exponentiel en la dimension du réseau euclidien, tandis que l'algorithme BKZ offre un compromis entre la taille du facteur d'approximation  $\gamma$  et le temps d'exécution de l'algorithme de résolution. Lorsque  $\gamma$  est polynomial en la dimension du réseau  $m$ , l'algorithme BKZ qui vise à résoudre  $SVP_\gamma$  a un temps d'exécution de  $2^{\tilde{O}(m)}$ . On pense qu'il n'existe pas d'algorithme classique ou quantique en temps polynomial capable de résoudre le problème  $SVP_\gamma$  pour des facteurs d'approximation  $\gamma$  polynomiaux en la taille de la dimension du réseau euclidien.

Cependant, les problèmes  $SVP$  et  $SVP_\gamma$  sont des problèmes pire-cas, ce qui signifie qu'ils sont difficiles à résoudre dans le pire des cas, mais pas qu'ils le sont pour n'importe quelle

instance d'un réseau euclidien. Par conséquent, ils ne sont pas particulièrement adaptés à la conception de protocoles cryptographiques, et les problèmes moyen-cas tels que SIS, LWE ou NTRU seront préférés lorsqu'il s'agit de construire des schémas cryptographiques, car ce sont des problèmes moyen-cas, ce qui signifie que des instances aléatoires sont difficiles à résoudre. Comme indiqué ci-dessus, les problèmes SIS et LWE bénéficient toutefois de réductions pire-cas moyen-cas, et sont donc également considérés comme résistants aux attaques des ordinateurs quantiques.

**Variantes Structurées.** Les schémas cryptographiques dont la difficulté est basée sur ces hypothèses ont cependant l'inconvénient d'être généralement moins efficaces en termes de taille de paramètres que leurs homologues classiques, dont la difficulté est basée sur les problèmes du logarithme discret ou de la factorisation. Cela s'explique par le fait qu'un réseau euclidien  $\Lambda$  est représenté par l'une de ses bases  $\mathbf{B} \in \mathbb{R}^{m \times m}$ , dont la taille est quadratique en la dimension  $m$  lorsque le réseau euclidien est de rang plein. Mais les opérations les plus utilisées dans les cryptosystèmes basés sur les réseaux euclidiens sont la multiplication matrice-matrice, dont le temps d'exécution est en  $O(m^3)$  et la multiplication matrice-vecteur, dont le temps d'exécution est en  $O(m^2)$ . Cependant, la difficulté de la résolution de problèmes tels que SVP n'augmente qu'exponentiellement avec  $m$ , et non avec  $m^2$ . Par conséquent, pour obtenir un système cryptographique basé sur les réseaux euclidiens avec  $\lambda$  bits de sécurité, il est nécessaire de travailler avec des paramètres de taille  $\lambda^2$ .

Pour résoudre ce problème, les réseaux euclidiens algébriques structurés offrent une solution possible pour améliorer l'efficacité des schémas cryptographiques basés sur les réseaux euclidiens. Des variantes structurées des problèmes difficiles mentionnés ci-dessus ont été introduites et largement étudiées, comme Ring-SIS [PR06], Ring-LWE [SST<sup>+</sup>09; LPR10] ou Module-SIS et Module-LWE [LS15]. Ces problèmes concernent les réseaux euclidiens structurés, tels que les réseaux *idéaux* ou *modules*. Les réseaux idéaux sont un sous-ensemble particulier des réseaux euclidiens qui possèdent des propriétés algorithmiques intéressantes. Fondamentalement, avec les réseaux idéaux, on travaille dans un anneau de polynômes  $\mathcal{R}$ , par exemple  $\mathcal{R} = \mathbb{Z}[X]/(X^m + 1)$  au lieu de travailler sur l'anneau des nombres entiers. Par conséquent, les vecteurs des réseaux idéaux correspondent à des polynômes et un polynôme, modulo  $X^m + 1$ , peut être vu comme un vecteur de dimension  $m$ . Un réseau idéal est alors représenté par la matrice de multiplication par un polynôme  $a \in \mathbb{Z}[X]/(X^m + 1)$ . L'utilisation d'algorithmes efficaces pour la multiplication

polynomiale tels que la Transformée de Fourier Rapide (FFT) permet alors d'effectuer une multiplication matrice-matrice en  $\tilde{O}(m^2)$  (au lieu de  $O(m^3)$ ) avec des réseaux non structurés) et  $\tilde{O}(m)$  (au lieu de  $O(m^2)$ ) avec des réseaux non structurés) opérations arithmétiques.

## Primitives Cryptographiques

Depuis les travaux d'Ajtai [Ajt96], les réseaux euclidiens ont été utilisés pour construire une large gamme de primitives cryptographiques, depuis les plus basiques telles que les fonctions de hachage à sens unique et résistantes aux collisions [Ajt96; Mic02], les schémas de signature [GGH97] ou les schémas de chiffrement [AD97; Reg05] à des primitives plus avancées telles que le chiffrement basé sur l'identité [GPV08; CHK<sup>+</sup>10; ABB10a], le chiffrement basé sur les attributs [ABV<sup>+</sup>12], les signatures de groupe [GKV10] ou le problème de longue date de la réalisation d'un chiffrement totalement homomorphe [Gen09]. Par conséquent, la cryptographie basée sur les réseaux euclidiens donne accès à de nombreuses fonctionnalités avancées, ce qui la distingue encore davantage des autres familles de cryptographie post-quantique.

Présentons quelques primitives de la cryptographie basée sur les réseaux euclidiens.

**Schémas de Signature.** Le premier schéma de signature basé sur les réseaux euclidiens a été présenté dans [GGH97]. Il s'agit d'un schéma de signature de type hacher et signer. Le principe du schéma est le suivant. La clef publique est une mauvaise base d'un réseau euclidien, tandis que la clef secrète détenue par le signataire est une bonne base (c'est-à-dire dont les vecteurs sont courts et presque orthogonaux) de ce même réseau euclidien. Pour signer un message  $M$ , le signataire commence par le hacher en une quantité  $H(M)$ , et utilise sa connaissance d'une bonne base du réseau euclidien pour calculer un point du réseau euclidien  $\nu$  proche de  $H(M)$ . Alors le signataire renvoie  $\nu$  comme signature du message  $M$ . Toute personne ayant connaissance d'une base quelconque du réseau euclidien peut alors vérifier que la signature est bien un point du réseau euclidien et qu'elle est proche du message haché  $H(M)$ . Cependant, ce schéma de signature tel que présenté dans [GGH97] n'était pas sécurisé, car les signatures révélaient des informations sur la base secrète utilisée pour les calculer, ce qui a conduit à diverses attaques [NR06].

Mais en 2008, Gentry, Peikert et Vaikuntanathan ont présenté dans [GPV08] un nouveau schéma de signature basé sur les réseaux euclidiens, qui évite le problème de la fuite d'informations sur la clef secrète en utilisant l'*échantillonnage Gaussien* pour générer

les signatures. En effet, une contribution cruciale de leur travail a été la présentation d'un algorithme d'échantillonnage (connu sous le nom d'échantillonnage GPV) qui permet d'utiliser une base courte comme trappe pour générer des vecteurs courts appartenant au réseau euclidien.

Concrètement, imaginons que le signataire veuille signer un message  $M$ . La clef publique du système de signature est un réseau euclidien représenté par une matrice  $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$  et la clef secrète est une trappe pour cette matrice  $\mathbf{A}$  (une base courte du réseau euclidien). L'échantillonnage Gaussien consiste alors à échantillonner un point suivant une distribution Gaussienne discrétisée sur les points du réseau euclidien et centrée sur le point  $H(M)$ . Avec l'échantillonnage Gaussien, les signatures ne divulguent aucune information sur la base courte qui a été utilisée.

Depuis, cet outil est largement utilisé dans la cryptographie basée sur les réseaux euclidiens.

**Schémas de Chiffrement.** Regev a présenté dans [Reg05] un système de chiffrement à clef publique dont la sécurité est basée sur la difficulté du problème LWE. Fondamentalement, la clef publique du système est un couple  $(\mathbf{A}, \mathbf{b}^T = \mathbf{s}^T \mathbf{A} + \mathbf{e}^T \text{ mod } q)$  et la clef secrète est le vecteur  $\mathbf{s}$ . Afin de chiffrer un message  $\mu \in \{0, 1\}$ , l'utilisateur choisit  $r$  uniformément aléatoire dans  $\{0, 1\}^m$  et calcule le chiffré  $(\mathbf{C} = \mathbf{A}r, C' = \mathbf{b}^T r + \lfloor q/2 \rfloor \cdot \mu)$ . Le destinataire peut alors déchiffrer le message en utilisant sa connaissance de la clef secrète  $\mathbf{s}$  en calculant  $C' - \mathbf{s}^T \mathbf{C} = \mathbf{e}^T r + \lfloor q/2 \rfloor \cdot \mu$ . Si le terme d'erreur  $\mathbf{e}^T r$  est suffisamment petit, il permet au destinataire de récupérer le message  $\mu$  en vérifiant si  $\mathbf{e}^T r + \lfloor q/2 \rfloor \cdot \mu$  est plus proche de 0 ou de  $\lfloor q/2 \rfloor$ .

**Schémas de Chiffrement Basés sur l'Identité.** Un système de chiffrement basé sur l'identité (IBE) est un système avancé de chiffrement à clef publique dans lequel une identité, telle qu'un nom d'utilisateur, une adresse mail ou un numéro de sécurité sociale, joue le rôle de clef publique. Dans un tel système, l'expéditeur chiffre un message avec l'identité unique du destinataire, et le destinataire déchiffre ensuite le texte chiffré avec sa clef privée pour obtenir le message original. De cette manière, une partie peut envoyer un message chiffré à n'importe quelle autre partie sans demander au préalable la clef publique du destinataire. En effet, la paire "identité" et "clef secrète associée" agit comme une paire classique de clefs publique et secrète dans un système classique de chiffrement à clef publique.

La notion de système cryptographique basé sur l'identité a été proposée par Shamir en 1984 dans [Sha84]. L'idée était d'éliminer la nécessité d'un certificat public dans les systèmes de messagerie électronique. En effet, ces systèmes permettent une communication sécurisée sans échange de clés d'utilisateur. Shamir a présenté une solution pour un système de signature basé sur l'identité, mais les premières constructions d'IBE ne sont apparues qu'en 2001 dans [BF01; Coc01] et étaient basées respectivement sur des fonctions bilinéaires et des résidus quadratiques. Cependant, ces schémas sont vulnérables aux attaques quantiques dues à l'algorithme de Shor [Sho94].

Dans [GPV08], Gentry, Peikert et Vaikuntanathan ont décrit le premier IBE basé sur des réseaux euclidiens, en s'appuyant sur le schéma de chiffrement Dual-Regev. Ils introduisent notamment un échantillonneur Gaussien qu'ils utilisent pour construire un IBE basé sur un réseau euclidien, qui s'est avéré adaptativement sécurisé dans le modèle de l'oracle aléatoire contre les attaques par texte choisi. Cependant, la clé publique maîtresse et les clés secrètes des utilisateurs avaient des tailles importantes, en  $O(n^2)$  bits. Plus tard, une construction d'un système IBE hiérarchique (HIBE) dans le modèle standard a été proposée dans [CHK<sup>+</sup>10] sur la base d'un nouveau mécanisme de délégation de clés des utilisateurs. Ce système IBE s'est avéré sécurisé dans le modèle sélectif. Dans ce modèle, l'adversaire doit cibler une identité à attaquer au préalable. En 2010, Agrawal et al. [ABB10a] ont proposé un IBE basé sur le problème **LWE** avec des performances comparables au schéma **GPV**. Cette construction considère une identité comme une séquence de bits et attribue une matrice à chaque bit. Elle utilise un algorithme d'échantillonnage pour obtenir une base dont la norme de Gram-Schmidt est petite pour la clé secrète maître et forme une famille de réseaux euclidiens avec deux trappes associées pour générer des vecteurs courts ; l'une pour tous les réseaux euclidiens de la famille et l'autre pour tous les réseaux euclidiens à l'exception d'un seul.

Le premier IBE basé sur la variante structurée **Ring-LWE** a été proposé par Ducas, Lyubashevsky et Prest [DLP14] (**DLP-IBE**), qui est toujours considéré comme le schéma le plus efficace à ce jour en raison de la taille réduite des clés. L'utilisation de la structure d'anneau améliore l'efficacité en réduisant la taille de la clé publique et des chiffrés à  $O(n)$ . La sécurité de leur système est valable dans le modèle de l'oracle aléatoire et est liée à l'hypothèse de difficulté **NTRU**. Une implémentation efficace en C de ce schéma et une analyse détaillée des performances ont été fournies dans [MSO17]. En 2017, Campbell et Grover ont présenté un schéma HIBE [CG17], appelé **Latte**, qui peut être considéré comme une combinaison du schéma **DLP-IBE** avec le mécanisme de délégation de [CHK<sup>+</sup>10]. Une



mise en œuvre optimisée et une analyse affinée de **Latte** ont été récemment proposées dans [ZMS<sup>+</sup>21].

L'article [BFR<sup>+</sup>18] présente un IBE utilisant la notion de trappes gadget, introduite par [MP12], pour des réseaux idéaux. De telles trappes peuvent être considérées comme des applications linéaires qui transforment des instances difficiles de problèmes cryptographiques sur certains réseaux euclidiens en instances faciles sur un réseau euclidien défini par une "matrice gadget" publique. L'IBE de [BFR<sup>+</sup>18] a également utilisé les algorithmes efficaces d'échantillonnage de préimages Gaussiennes de [GM18] pour proposer une implémentation de leur schéma. Dans [ZMS<sup>+</sup>21], les auteurs ont proposé de nouveaux algorithmes efficaces d'échantillonnage pour les trappes gadget, qui ne nécessitaient pas d'arithmétique à virgule flottante et qui sont aussi rapides que l'échantillonneur original [GM18].

## Contributions

Ce manuscrit présente les travaux réalisés au cours de ma thèse. J'ai principalement travaillé sur la construction et l'implémentation de schémas cryptographiques efficaces, en particulier les schémas de chiffrement basés sur l'identité, en me concentrant sur l'amélioration de l'efficacité des algorithmes basés sur les trappes dans la conception de ces protocoles. Un aperçu rapide de mes contributions est présenté ci-dessous.

**Trappes sur des Réseaux Modules.** J'ai travaillé dans un premier temps sur le développement et l'implémentation de techniques d'échantillonnage de préimages gaussiennes sur des réseaux modules, qui s'appuient sur des travaux de Micciancio et Peikert en 2012 [MP12], et de Micciancio et Genise en 2018 [GM18].

L'objectif était de concevoir des schémas cryptographiques efficaces basés sur des trappes. Alors que les premières constructions basées sur les trappes GPV [GPV08] n'ont jamais été instanciées en pratique en raison de leur inefficacité, j'ai travaillé avec une autre notion de trappes introduite dans [MP12], appelée trappes gadget, et récemment améliorée en 2018 dans [GM18], que j'ai adaptée pour le cas des réseaux modules.

Cet objectif de construire des algorithmes efficaces pour les schémas cryptographiques basés sur les gadgets dans le cadre des réseaux modules semblait intéressant car ces derniers possèdent une structure algébrique supplémentaire, s'appuyant sur un anneau polynomial sous-jacent qui conduit à une représentation plus compacte des paramètres

et à un temps d'exécution amélioré. Alors que les anneaux et les réseaux idéaux [PR06; SST<sup>+</sup>09; LPR10], généralement basés sur des anneaux de la forme  $\mathcal{R}_q = \mathbb{Z}_q[X]/\langle X^n + 1 \rangle$ , sont souvent le premier choix pour les constructions efficaces basées sur les réseaux euclidiens, les réseaux modules [LS15], basés sur des modules de la forme  $\mathcal{R}_q^d$ , se situent à mi-chemin entre les réseaux idéaux et les réseaux non structurés. Les constructions basées sur les réseaux modules sont aussi efficaces que celles basées sur les anneaux, et présentent d'autres avantages en pratique. En effet, par rapport aux réseaux idéaux, les réseaux modules ont une structure algébrique plus faible qui se traduit en pratique par un choix plus souple de paramètres pour les schémas cryptographiques et un meilleur contrôle du compromis entre efficacité et sécurité.

J'ai donc travaillé à l'adaptation de différents algorithmes basés sur les trappes gadget, tels que les algorithmes de génération de trappes et d'échantillonnage gaussien, dans le cadre des réseaux modules. Afin d'évaluer les performances de ces algorithmes, j'ai également travaillé sur une implémentation en C. Cette implémentation bénéficie d'une grande modularité : elle ne nécessite pas de dépendances externes, est facile à modifier si nécessaire et les blocs peuvent être facilement substitués par d'autres (il est possible de changer l'arithmétique sur l'anneau  $\mathcal{R}_q$ , le générateur de nombres pseudo-aléatoires et l'échantillonneur gaussien peuvent aisément être remplacés par d'autres instanciations). Par conséquent, je pense que cette implémentation pourrait être utile en tant qu'outil pour mettre en œuvre diverses constructions cryptographiques basées sur les réseaux et utilisant des trappes gadgets.

En m'appuyant sur ces outils, en guise d'applications, j'ai travaillé sur deux instanciations et implémentations de schémas de signature à base de trappes, basés sur les réseaux modules : GPV dans le modèle de l'oracle aléatoire et une variante de celui-ci dans le modèle standard. À ma connaissance, il s'agit de la première mise en œuvre efficace d'un système de signature basé sur les réseaux dans le modèle standard. En m'appuyant sur cette dernière signature, j'ai également travaillé à la mise en œuvre d'un IBE dans le modèle standard reposant sur les réseaux modules. Bien que les schémas résultants ne soient pas compétitifs avec les schémas les plus efficaces sélectionnés par le NIST, ils restent efficaces en pratique, ce qui ouvre la voie à des constructions pratiques avancées basées sur les trappes.

Ce travail est l'objet principal du Chapitre 2 de ce manuscrit.

- [BEP<sup>+</sup>21] Pauline Bert, Gautier Eberhart, Lucas Prabel, Adeline Roux-Langlois, Mohamed Sabt: Implementation of Lattice Trapdoors on Modules and Applications.

PQCrypto 2021: 195-214

**Trappes Approchées et Applications.** Par la suite, le reste de mes travaux s'est naturellement orienté vers la recherche de nouvelles méthodes pour améliorer l'efficacité des schémas cryptographiques utilisant des trappes gadget.

Sur la base de mes travaux brièvement présentés dans le paragraphe précédent, j'ai travaillé sur l'amélioration de l'efficacité du schéma de chiffrement basé sur l'identité. J'ai étudié la possibilité d'utiliser des trappes approchées, avec des hypothèses SIS et LWE, plutôt que les trappes exactes de [MP12], afin d'obtenir un schéma plus efficace, sans réduire drastiquement la sécurité intrinsèque.

Les trappes approchées ont été introduites par Chen, Genise et Mukherjee dans [CGM19]. Dans leur article, ils introduisent le problème approché (AISIS), qui demande, étant donné  $\mathbf{A} \in \mathcal{R}_q^{d \times m}$  et  $\mathbf{y} \in \mathcal{R}_q^d$ , de trouver un vecteur  $\mathbf{x} \in \mathcal{R}^m$  tel que  $\|\mathbf{x}\| \leq \beta$  et tel qu'il existe un vecteur  $\mathbf{z} \in \mathcal{R}^d$  satisfaisant :  $\|\mathbf{z}\| \leq \alpha$  et  $\mathbf{Ax} = \mathbf{y} + \mathbf{z} \bmod q$ .

Ils montrent que ce problème est aussi difficile que le problème original ISIS pour certains ensembles de paramètres, et introduisent le concept de trappes gadget approchées, lié à ce problème AISIS. En pratique, les trappes approchées sont générées en supprimant les entrées  $\ell$  correspondant aux petites puissances de  $b$  de la matrice gadget  $\mathbf{G}$  afin d'obtenir la matrice suivante :  $\mathbf{F} = \mathbf{I}_d \otimes \mathbf{f}^T = \mathbf{I}_d \otimes \begin{bmatrix} b^\ell & b^{\ell+1} & b^{\ell+2} & \dots & b^{k-1} \end{bmatrix} \in \mathcal{R}^{d \times d(k-\ell)}$ .

Cela permet de réduire la taille de la matrice publique, de la trappe et des préimages utilisées dans les constructions cryptographiques par un facteur 2 sans drastiquement impacter la sécurité concrète de ces schémas.

Cependant, l'utilisation de trappes approchées conduit à l'apparition de termes d'erreur qui doivent être pris en compte dans la phase de déchiffrement. J'ai travaillé sur l'utilisation de ces trappes approchées pour concevoir un IBE qui peut gérer ce terme d'erreur tout en utilisant des matrices tags. Le schéma que j'ai obtenu vérifie la sécurité IND-sID-CPA dans le modèle standard et a donné lieu à une implémentation montrant qu'il est plus efficace que l'équivalent IBE utilisant des trappes exactes.

- [IPR23] Malika Izabachène, Lucas Prabel and Adeline Roux-Langlois: Identity-Based Encryption From Lattices Using Approximate Trapdoors, ACISP 2023

**NTRU Lattices.** Pendant ma thèse, je me suis aussi intéressé à l'étude des réseaux NTRU, à l'efficacité (en termes de taille des paramètres et de temps d'exécution) de schémas cryptographiques utilisant cette famille de réseaux, aux perspectives d'implémentation

et à la sécurité concrète des systèmes cryptographiques basés sur une variante de cette hypothèse de difficulté.

J'ai donc travaillé sur cette variante de NTRU, appelée iNTRU, introduite dans [GGH<sup>+</sup>19]. Elle peut être considérée comme une variante inhomogène de NTRU. Fondamentalement, la version matricielle de iNTRU demande de distinguer entre une matrice  $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$  définie comme  $\mathbf{A} = \mathbf{S}^{-1}(\mathbf{G} - \mathbf{E}) \bmod q$  (où  $\mathbf{S} \in \mathbb{Z}_q^{n \times n}$  est une matrice aléatoire inversible,  $\mathbf{E}$  est une matrice de petite norme et  $\mathbf{G}$  est la matrice gadget) à partir d'une matrice choisie uniformément aléatoire dans  $\mathbb{Z}_q^{n \times m}$ .

L'utilisation de trappes gadget n'étant pas directement compatible avec l'hypothèse de difficulté de NTRU, j'ai utilisé cette variante particulière de NTRU pour construire un schéma de chiffrement basé sur l'identité et reposant sur le paradigme [MP12], afin d'obtenir un IBE plus efficace en termes de temps d'exécution et de tailles de paramètres. Ce nouveau schéma, dont la sécurité est prouvée dans le modèle de l'oracle aléatoire, a également donné lieu à une implémentation, ce qui a permis d'évaluer ses performances en termes de temps et de les comparer à celles de mon précédent IBE.

Je me suis également intéressé à la sécurité concrète des schémas cryptographiques basés sur l'hypothèse de difficulté iNTRU, la littérature étant encore récente sur ce sujet, ce nouveau problème ayant été introduit en 2019. Je me suis plus particulièrement intéressé à l'adaptation des attaques de type "overstretched" sur cette variante. Ces attaques ont été introduites dans [KF17] dans le cas des réseaux NTRU, et s'appliquent aux instances qui nécessitent un grand module  $q$  par rapport à la dimension  $2n$  du réseau sous-jacent au schéma.

En outre, je me suis intéressé plus récemment aux résolutions des équations NTRU, qui apparaissent en particulier dans plusieurs cryptosystèmes tels que Falcon [FHK<sup>+</sup>17] ou BAT [FKP<sup>+</sup>22], pendant la phase de génération de clefs. Mon objectif était d'étudier les techniques permettant de résoudre efficacement les équations NTRU, lorsque l'on travaille avec des corps de nombres qui n'admettent pas de tour de sous-corps. Dans ce cas, la résolution efficace de ces équations telle que présentée dans [PP19] en utilisant la norme et la structure de sous-corps n'est plus possible, et d'autres solutions doivent être trouvées. Je me suis également intéressé à la possibilité d'éviter l'arithmétique en virgule flottante pendant la phase de réduction des algorithmes de résolution des équations NTRU, afin d'obtenir une implémentation efficace.

- [IPR23] Malika Izabachène, Lucas Prabel and Adeline Roux-Langlois: Identity-Based Encryption From Lattices Using Approximate Trapdoors, ACISP 2023

Une partie de ce travail a également été publiée dans l'article [IPR23] cite précédemment, et une autre partie est toujours en cours et n'a pas encore été soumise pour publication.

# INTRODUCTION

---

**C**RYPTOLOGY is literally the *science of secrets*. Indeed, the word comes from ancient Greek, *κρυπτός* meaning *secret* and *λόγος* meaning *science*. It encompasses both *cryptography*, whose object of study is to protect messages by making them secret; and *cryptanalysis*, which, on the contrary, seeks to attack these protections to find the original messages hidden behind those secrets. However, the prerogatives of cryptography have extended well beyond the original objective of making a message secret (called *confidentiality*), since it is also concerned with other aspects of security, such as *integrity* (whose objective is to deal with alterations of data) or *authentication* (whose objective is to track the origin of a received message). Cryptography is nowadays a rich science, with broad objectives and based on strong scientific foundations.

## A Brief History

But the history of cryptography is a long one. In *The Codebreakers*, published in 1967, David Kahn recounts this history, going back to its first traces of use in ancient Egypt 4,000 years ago, with hieroglyphic symbol substitutions carved in the main chamber of the tomb of a nobleman by his scribe.

During its initial 3,000-year span, the field of cryptology did not experience steady growth. Instead, cryptology independently emerged in various regions but progress remained slow and sporadic, with more knowledge being lost than preserved. It is only with the advent of the Western Renaissance that accumulated knowledge in cryptology begins to gain momentum.

Prior to this, we can highlight a few occurrences of practical applications of simple cryptographic techniques.

The Caesar cipher, named after Julius Caesar, is one of the earliest and simplest known encryption techniques. It has been used, along with other more sophisticated techniques, in Ancient Rome for military purposes. It is a cryptographic method based on monoalphabetic substitution. A cipher is computed from a plaintext by replacing letters from the plaintext with letters standing  $k$  positions further down the alphabet (and starting

again from the beginning for the last letters of the alphabet), where  $1 \leq k \leq 25$  is a fixed integer, representing the secret key of the scheme. On its own, this method does not satisfy communication security, because of the very small number of possible keys, which means that they can be systematically tried out when the encryption method is known, but also because, like any encryption based on monoalphabetic substitution, it can very quickly be "broken" by frequency analysis of the letters appearing in the ciphers.

The Vigenère cipher, described for the first time during Late Renaissance, is a polyalphabetic substitution cipher which uses a Caesar cipher with a different shift at each position in the plaintext. The value of this shift is given by another text which serves as a secret key which can be reused repeatedly depending on the plaintext's length. Therefore, the same letters of the plaintext can, depending on their position in it, be replaced by different letters, unlike mono-alphabetic encryption systems such as the Caesar cipher. This method is thus resistant to a straightforward frequency analysis attack. However, correctly guessing the key's length  $n$  of the Vigenère cipher, the ciphertext can be seen as  $n$  interleaved Caesar ciphers, which can easily be broken individually.

But the field of cryptography really evolved and took the form and the formalism we know today during the late 1890s and the 20th century.

In the late 19th century, Auguste Kerckhoffs expounded in his book *La cryptographie militaire* [Ker83] a few principles to respect to design good military ciphers. One of the most important of these principles, which is now called *Kerckhoffs' principle*, states that "the cipher method must not be required to be secret, and it must be able to fall into the hands of the enemy without inconvenience". Basically, it means that the security of an encryption scheme should not rely on the encryption scheme's design being secret but on the secrecy of the key, which represents the crucial element of the system and which is kept confidential and known only by one or more authorised persons. Nowadays, Kerckhoffs' principle is widely acknowledged and commonly interpreted as promoting the idea that the entire design of a cryptosystem should be openly available to the public. This stands in sharp contrast to the concept of "security through obscurity," which considers design or implementation secrecy as the main method to guarantee the security of a cryptosystem.

During the 20th century, both World Wars played an important role in the development of cryptography and cryptanalysis. Alan Turing, renowned for his mathematical design of the early computer models, participated in their realization during the Second World War and played a major role in the cryptanalysis of the Enigma machine used by the Nazi armies.

The rise of computers and communication systems during the 1960s led to a growing need for digital tools to protect information and services ensuring security. This demand sparked a wide range of developments in the field of cryptography, from Feistel's work at IBM during the 1970s, ultimately resulting in the adoption of the Data Encryption Standard (DES), a symmetric-key algorithm published as an official *Federal Information Processing Standard* (FIPS) for the United States in 1977. Consequently, from this point, academic research in the field of cryptography will greatly intensify. As a result, a comprehensive theory started to unfold, making possible a rigorous study of cryptography as both a scientific and mathematical field of research.

Since the 1990s, the development of the use of the Internet for various purposes including commercial operations advocated for a widespread standard for encryption. Once largely confined to military and government use, cryptography is now ubiquitous in our daily lives.

## Public-Key Cryptography

With *symmetric* (or *private*) *key cryptography*, the use of a common secret key between the sender and the receiver is necessary, thus requiring both parties to have a physical meeting beforehand, or to use a trusted third party beforehand. However, the breakthrough work [DH76] of Diffie and Hellman in 1976 would change the deal. In their article *New Directions in Cryptography*, they introduce the concept of *public* (or asymmetric) *key cryptography*. Each user can now compute a publicly accessible key known as a public-key, and its associated private-key, which is only known by him. In a public-key encryption scheme for example, the public-key of a user allows anyone to encrypt a message, but only the owner of the corresponding private-key would be able to decrypt it. Therefore, this new concept makes it possible to solve the problem of a prior physical meeting mentioned above for symmetric key cryptography. Nowadays, in communications, a public-key cryptographic scheme will often be used together with a private-key scheme: the first one allows two parties who wish to communicate to agree on a common secret key and the second one is used for the rest of the communications. Indeed, private-key cryptography schemes are usually more efficient than their public-key counterparts, which makes them more attractive to use for cryptographic protocols.

In [DH76], Diffie and Hellman also provide a new method for key exchange, whose security is based on the hardness of the discrete logarithm problem. Key exchange is



a protocol that helps two parties to agree on a common secret key, without the need to exchange some private information beforehand. In their original paper, the public parameters of their key exchange protocol are a cyclic group  $\mathbb{G}$ , its order  $q$  and a generator  $g \in \mathbb{G}$ . Then, Alice chooses  $a \in \mathbb{Z}_q$  uniform and computes  $g^a$ . She sends this quantity to Bob. The latter also chooses  $b \in \mathbb{Z}_q$  uniform and computes  $g^b$ , which he sends to Alice. Then, both Alice and Bob can compute  $g^{ab} = (g^a)^b = (g^b)^a$ , which becomes their common secret key.

While Diffie and Hellman didn't present a public-key encryption scheme in their article, in 1978, Rivest, Shamir, and Adleman introduced in [RSA78] the first practical realization of public-key encryption and signature schemes. This time, the security of RSA schemes relies on the factorization problem of a product of large integers. In [Gam84], another public-key encryption scheme is introduced by El Gamal, based on the discrete logarithm problem.

In public-key cryptography, the security of protocols is therefore based on the conjectured hardness of some mathematical problems, such as the discrete logarithm or large integers factorization problems. Those two specific problems are omnipresent in classical public-key cryptography. However, cryptographic protocols relying on these two problems are threatened by the possible emergence of quantum computers. Indeed, there is an important difference in nature between operations done by quantum and classical computers. In particular, quantum computers work with quantum bit or *qubit* (which is the quantum version of the classical binary bit) which has an infinite set of alternative values besides 0 and 1. This allows quantum computers to become much more efficient than classical computers for some specific problems. Indeed, Shor showed in [Sho94] how quantum algorithms could efficiently solve the discrete logarithm and factorization problems in polynomial time.

However, quantum computers do not currently exist on a large scale and are not powerful enough yet. While the current quantum computers can handle around a couple of hundred qubits, it is believed that more than 1 million are necessary for breaking the factorization problem. Nonetheless, progress made in quantum computing is steady, and it is important to contemplate alternatives for a future in which quantum computers become widely accessible. Additionally, there exists the risk of malicious individuals collecting encrypted messages today with the intent to decrypt them at a later time, once quantum computers are available. This potential risk of the advent of such machines has prompted the scientific community to search for new security assumptions, potentially resistant

against this threat, which would make it possible to build schemes that remain secure even when adversaries have such computers at their disposal. This is an active research field known as *post-quantum cryptography*.

## Post-Quantum Cryptography

It is with this perspective in mind that in 2016, the National Institute of Standards and Technology (NIST), an agency of the United States Department of Commerce, launched the Post-Quantum Cryptography Standardization program and competition, whose goal was to update their standards to include post-quantum cryptographic schemes. The initial submissions from the scientific community to this competition included 23 signature schemes and 59 encryption/key encapsulation mechanism schemes. The candidate cryptographic schemes were based on a wide variety of problems supposedly difficult to attack efficiently by a quantum computer. Nowadays, we can distinguish in particular 5 main families that make up the landscape of post-quantum cryptography. Those families rely on the hardness of problems defined over lattices, error-correcting codes, isogenies of elliptic curves, multivariate polynomials, and hash functions.

In this thesis, we will be interested in post-quantum cryptography based on mathematical problems defined over lattices.

## Lattice-Based Cryptography

Indeed, lattice-based cryptography is one of the most promising solutions against the quantum threat. This field of research has been very active since the works of Ajtai [Ajt96], which introduces the Short Integer Solution problem (SIS) and Regev [Reg05], which introduces the Learning With Error problem (LWE). In those articles, interesting worst-case to average-case reductions are proved, which attracted many researchers into the field of lattice-based cryptography. Basically, those reductions state that if the security of a cryptosystem is based on an average-case problem (such as SIS or LWE), attacking random instances of this cryptosystem is at least as hard as solving all instances of the underlying worst-case lattice problem. Thus, average-case problems such as SIS and LWE are particularly well-suited for designing cryptographic schemes. Therefore, those reductions provide strong security guarantees for lattice-based cryptography. Besides, other efficient schemes have based their security on average-case problems over structured lattice such

as the NTRU encryption scheme, which was introduced in [HPS98].

**Euclidean Lattices.** A lattice can be seen as a periodic arrangement of points forming a "grid" (see Figure 3). More formally, given a set of linearly independent vectors  $\mathbf{B} = \{\mathbf{b}_1, \dots, \mathbf{b}_k\} \subset \mathbb{R}^m$ , we call lattice with basis  $\mathbf{B}$  and rank  $k$  the following set:

$$\mathcal{L}(\mathbf{B}) := \left\{ \sum_{i=1}^k \lambda_i \mathbf{b}_i, \lambda_i \in \mathbb{Z} \right\}.$$

When  $k = m$ , the lattice is said to be full-rank. A lattice  $\Lambda$  admits infinitely many bases for  $m \geq 2$ .

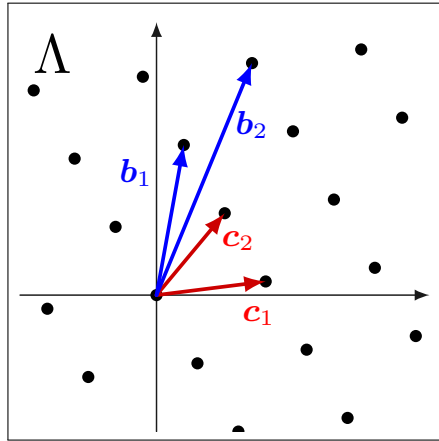


Figure 3 – A 2-dimensional lattice  $\Lambda$  with two of its basis:  $\mathbf{B} = (\mathbf{b}_1, \mathbf{b}_2)$  and  $\mathbf{B}' = (\mathbf{c}_1, \mathbf{c}_2)$ .

Many algorithmic problems related to lattices have been introduced and widely studied. One of them is the Shortest Vector Problem (SVP). This problem asks, when given a basis  $\mathbf{B}$  of a lattice  $\Lambda$ , to find a shortest non-zero vector of this lattice (see Figure 4). We denote  $\lambda_1(\Lambda)$  the Euclidean norm of any smallest non-zero vector of the lattice  $\Lambda$ . An approximate variant of this problem, denoted  $\text{SVP}_\gamma$ , asks to find  $\mathbf{x} \in \Lambda$  such that  $\|\mathbf{x}\| \leq \gamma \cdot \lambda_1(\Lambda)$ .

Those problems are supposed to be difficult to solve. The best algorithms to date seeking to solve them, LLL [LLL82] and BKZ [SE94], are based on lattice reductions. The LLL algorithm, introduced by Lenstra, Lenstra and Lovász in 1982, solves the  $\text{SVP}_\gamma$  problem in polynomial time when  $\gamma$  is exponential in the dimension of the lattice, while the BKZ algorithm offers a trade-off between the size of the approximation factor  $\gamma$  and

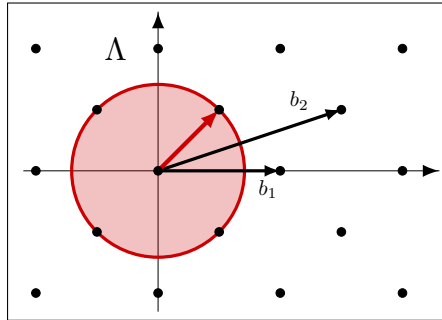


Figure 4 – A 2-dimensional lattice  $\Lambda$  with basis  $\mathbf{B} = (\mathbf{b}_1, \mathbf{b}_2)$  and a **shortest vector**.

the running time of the solving algorithm. When  $\gamma$  is polynomial in the lattice dimension  $m$ , the BKZ algorithm which aims to solve  $\text{SVP}_\gamma$  has a running time of  $2^{\tilde{O}(m)}$ . It is believed that there is no polynomial time classical or quantum algorithm able to solve the  $\text{SVP}_\gamma$  problem for approximation factors  $\gamma$  polynomial in the size of the dimension of the lattice.

However, the  $\text{SVP}$  and  $\text{SVP}_\gamma$  problems are worst-case problems, meaning they are hard to solve in the worst-case, and not for any instance of a lattice. Therefore, they are not particularly well-suited for designing cryptographic protocols, and average-case problems such as  $\text{SIS}$ ,  $\text{LWE}$  or  $\text{NTRU}$  will be preferred when it comes to building cryptographic schemes, because they are average-case problems, meaning random instances are hard to solve. As written above,  $\text{SIS}$  and  $\text{LWE}$  do however benefit from worst-case to average-case reductions, and are thus also believed to be quantum resistant.

**Structured Variants.** Cryptographic schemes whose hardness is based on those lattice assumptions however have the disadvantage of being usually less efficient in terms of parameter sizes than their classical counterparts, whose hardness is based on the discrete logarithm or factorization problems. This is explained by the fact that a lattice  $\Lambda$  is represented by one of its basis  $\mathbf{B} \in \mathbb{R}^{m \times m}$ , whose size is quadratic in the dimension  $m$  when the lattice is full-rank. But the most used operations in lattice-based cryptosystems are matrix-matrix multiplication, whose running time is in  $O(m^3)$  and matrix-vector multiplication, whose running time is in  $O(m^2)$ . However, the hardness of solving problems such as  $\text{SVP}$  only increases exponentially with  $m$ , and not with  $m^2$ . Therefore, in order to have a lattice-based cryptographic scheme with  $\lambda$  bits of security, we need to work with parameters of sizes  $\lambda^2$ .

In order to deal with this problem, algebraically structured lattices offer a possible solution to improve the efficiency of lattice-based cryptographic schemes. Structured vari-

ants of the hard problems mentioned above have been introduced and widely studied, such as Ring-SIS [PR06], Ring-LWE [SST<sup>+</sup>09; LPR10] or Module-SIS and Module-LWE [LS15]. These problems deal with structured lattices, such as *ideal* or *module* lattices. Ideal lattices are a special subset of lattices that possess interesting computational properties. Basically, when working with ideal lattices, we work in a ring of polynomials  $\mathcal{R}$ , often chosen as  $\mathcal{R} = \mathbb{Z}[X]/(X^m + 1)$  instead of working over the ring of integers. Therefore, vectors of ideal lattices correspond to polynomials and a polynomial, modulo  $X^m + 1$ , can be seen as a vector of dimension  $m$ . An ideal lattice is then represented by the skew-circulant matrix of multiplication by a polynomial  $a \in \mathbb{Z}[X]/(X^m + 1)$ . Using efficient polynomial multiplication algorithms such as the Fast Fourier Transform (FFT) then allows performing matrix-matrix multiplication in  $\tilde{O}(m^2)$  (instead of  $O(m^3)$  with unstructured lattices) and  $\tilde{O}(m)$  (instead of  $O(m^2)$  with unstructured lattices) arithmetic operations.

## Cryptographic Primitives

Since the work of Ajtai [Ajt96], lattices have been used to construct a wide range of cryptographic primitives, from basic primitives such as one-way and collision-resistant hash functions [Ajt96; Mic02], signature schemes [GGH97] or encryption schemes [AD97; Reg05] to more advanced primitives such as identity-based encryption [GPV08; CHK<sup>+</sup>10; ABB10a], attribute-based encryption [ABV<sup>+</sup>12], group signatures [GKV10] or the long-standing problem of realizing fully homomorphic encryption [Gen09]. Therefore, lattice-based cryptography enables many advanced functionalities, which further sets it apart from other families of post-quantum cryptography.

Let's start by presenting some primitives of lattice-based cryptography.

**Lattice-Based Signature Schemes.** The first lattice-based signature scheme was introduced in [GGH97]. It is a hash-and-sign signature scheme. The principle of the scheme is as follows. The public key is a *bad* basis of a lattice, while the secret key owned by the signer is a *good* basis of this same lattice. To sign a message  $M$ , the signer first hashes it to a quantity  $H(M)$ , and uses his knowledge of a good basis of the lattice to compute a lattice point  $\nu$  close to  $H(M)$ . Then, the signer outputs  $\nu$  as a signature of the message  $M$ . Anyone knowing a basis of the lattice can then check that the signature is indeed a lattice point and that it is close to the hashed message  $H(M)$ . However, this signature scheme as presented in [GGH97] was not secure, because signatures revealed information about the secret basis used to compute them, leading to various attacks breaking the

scheme [NR06].

However, in 2008, Gentry, Peikert and Vaikuntanathan presented in [GPV08] a new lattice-based signature scheme, which avoids the problem of the signatures leaking information about the secret key by using *Gaussian sampling* in order to generate signatures. Indeed, a crucial contribution of their work was a sampling algorithm (known as GPV sampling) which allows using a short basis as a trapdoor for generating short lattice vectors.

Concretely, imagine the signer wants to sign a message  $M$ . The public key of the signature scheme is a lattice represented by a matrix  $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$  and the secret key is a trapdoor for this matrix  $\mathbf{A}$  (basically a short basis of the lattice). Then, Gaussian sampling consists to sample a point following a Gaussian distribution discretized on the lattice points and centered on the point  $H(M)$ . With Gaussian sampling, signatures don't leak any information about the short basis which was used.

This tool has been widely used in lattice-based cryptography ever since.

**Lattice-Based Encryption Schemes.** Regev introduced in [Reg05] a public-key encryption scheme whose security was based on the hardness of the LWE problem. Basically, the public key of the scheme is a couple  $(\mathbf{A}, \mathbf{b}^T = \mathbf{s}^T \mathbf{A} + \mathbf{e}^T \text{ mod } q)$  and the secret key is the vector  $\mathbf{s}$ . In order to encrypt a message  $\mu \in \{0, 1\}$ , the user chooses  $r$  uniformly random in  $\{0, 1\}^m$  and computes the ciphertext  $(\mathbf{C} = \mathbf{A}r, C' = \mathbf{b}^T r + \lfloor q/2 \rfloor \cdot \mu)$ . Then the recipient can find the message using his knowledge of the secret key  $\mathbf{s}$  by computing  $C' - \mathbf{s}^T \mathbf{C} = \mathbf{e}^T r + \lfloor q/2 \rfloor \cdot \mu$ . If the error term  $\mathbf{e}^T r$  is small enough, it allows the recipient to recover the message  $\mu$  by checking if  $\mathbf{e}^T r + \lfloor q/2 \rfloor \cdot \mu$  is closer to 0 or  $\lfloor q/2 \rfloor$ .

**Identity-Based Encryption Schemes.** An Identity-Based Encryption (IBE) scheme is an advanced public-key encryption scheme in which an identity, such as a username, an email address, or a social security number, acts as the public-key. In such a scheme, the sender encrypts a message with the unique identity of the recipient, and the recipient then decrypts the ciphertext with their private-key to obtain the original message. By doing so, one party can send an encrypted message to any other party without requesting the recipient's public key beforehand. Indeed, the pair "identity" and "associated secret key" acts as a classical public-key and secret key-pair in a classical public key encryption scheme.

The notion of identity-based schemes was proposed by Shamir in [Sha84]. The idea

was to eliminate the need for a public certificate across email systems. Indeed, these schemes allow secure communication without exchanging user keys. Shamir presented a solution for an identity-based signature scheme but the first IBE constructions appeared only in 2001 in [BF01; Coc01] and were based respectively on bilinear maps and quadratic residue based assumptions. However, these schemes are vulnerable to quantum attacks due to Shor's algorithm [Sho94].

In [GPV08], Gentry, Peikert and Vaikuntanathan described the first lattice-based IBE, relying on the Dual-Regev encryption scheme. The Gaussian sampler they introduce was used to construct a lattice-based IBE scheme, proven adaptively secure against chosen-plaintext attacks in the random oracle model. However, the master public key and user secret keys had large sizes in  $O(n^2)$  bits. Later on, a construction of a Hierarchical IBE (HIBE) scheme in the standard model was proposed in [CHK<sup>+</sup>10] based on a new mechanism for users' keys delegation. This IBE scheme was proven secure in the selective model. In this model, the adversary needs to target an identity beforehand. In 2010, Agrawal et al. [ABB10a] proposed a **LWE**-based IBE scheme with a trapdoor structure and with performance comparable to the **GPV** scheme. Their construction viewed an identity as a sequence of bits and then assigned a matrix to each bit. It used a sampling algorithm to obtain a basis with a low Gram-Schmidt norm for the master secret key and formed a lattice family with two associated trapdoors to generate short vectors; one for all lattices in the family and the other one for all but one.

The first **Ring-LWE** based IBE scheme has been proposed by Ducas, Lyubashevsky and Prest [DLP14] (**DLP-IBE**), which is still considered the most efficient scheme to date due to smaller key sizes. The use of the ring variant improved efficiency by reducing the public key size and ciphertext size to  $O(n)$ . The security of their scheme holds in the random oracle model and is related to the **NTRU** hardness assumption. An efficient C implementation of this scheme and a detailed performance analysis was provided in [MSO17]. In 2017, Campbell and Grover introduced a HIBE scheme [CG17], called **Latte**, which can be viewed as a combination of the **DLP-IBE** scheme with the delegation mechanism from [CHK<sup>+</sup>10]. An optimized implementation and refined analysis of **Latte**, has recently been proposed in [ZMS<sup>+</sup>21].

The work from [BFR<sup>+</sup>18] constructed an IBE using the notion of gadget-based trapdoors in the ring setting, introduced by [MP12]. Such trapdoors can be seen as linear transformations mapping hard instances of cryptographic problems on some lattices to easy instances on a lattice defined by a public "gadget matrix". The IBE from [BFR<sup>+</sup>18]

also made use of the efficient Gaussian preimage sampling algorithms from [GM18] to propose an implementation of their scheme. In [ZMS<sup>+</sup>21], the authors proposed new efficient gadget sampling algorithms which didn't need floating-point arithmetic, and as fast as the original [GM18] sampler.

All those constructions generally make use of dedicated trapdoors, needed by the authority to generate the secret key of a user. In that case, building the trapdoor and sampling particular short vectors are quite costly, and represent the main bottleneck in the efficiency of such schemes.

## Contributions

This manuscript presents the work done during my PhD. I mainly worked on the constructions and implementations of efficient cryptographic schemes, especially identity-based encryption schemes, focusing on improving the efficiency of trapdoor-based algorithms in the design of these protocols. A quick overview of my contributions is presented below.

**Lattice Trapdoors on Modules.** I worked on the development and implementation of Gaussian preimage sampling techniques on module lattices, which rely on the works of Micciancio and Peikert in 2012 [MP12], and Micciancio and Genise in 2018 [GM18].

The objective was to design efficient trapdoor-based cryptographic schemes. While early constructions based on GPV trapdoors [GPV08] were never instantiated in practice because of their inefficiency, I worked with another notion of trapdoors introduced in [MP12], called gadget trapdoors, and recently improved in 2018 in [GM18] that I adapted to the module setting.

This objective of constructing efficient algorithms for gadget-based cryptographic schemes in the module setting seemed interesting because module lattices possess additional algebraic structure, relying on an underlying polynomial ring which leads to a more compact representation and an improved running time. While the ring setting and ideal lattices [PR06; SST<sup>+</sup>09; LPR10], usually based on rings of the form  $\mathcal{R}_q = \mathbb{Z}_q[X]/\langle X^n + 1 \rangle$ , are often the first choice for efficient lattice-based constructions, module lattices [LS15], based on modules of the form  $\mathcal{R}_q^d$ , lie somewhere between ideal lattices and unstructured ones. Constructions in the module setting are as efficient as ring-based ones, and have other advantages for practical schemes. Indeed, compared to the ring setting, module



lattices have a weaker algebraic structure which translates in practice to a more flexible choice of parameters for cryptographic schemes and more control over the trade-off between efficiency and security.

Thus, I worked on adapting different algorithms based on gadget trapdoors, such as trapdoor generation and Gaussian sampling algorithms, in the module setting. In order to assess the performance of these algorithms, I also worked on a C implementation from scratch. This implementation is really modular: it does not require external dependencies, is easy to modify if needed and blocks can be easily swapped out (it is possible to change the arithmetic over the ring  $\mathcal{R}_q$ , the pseudorandom number generator and the (constant-time) sampler of discrete Gaussian distributions can be swapped for other instantiations, ...). Therefore, I believe this implementation could be useful as a tool to implement various lattice-based cryptographic constructions using gadget trapdoors.

Relying on these tools, as applications, I worked on two instantiations and implementations of proven trapdoor-based signature schemes in the module setting: GPV in the random oracle model and a variant of it in the standard model. To the best of my knowledge, this is the first efficient implementation of a lattice-based signature scheme in the standard model. Relying on that last signature, I also worked on the implementation of a standard model IBE in the module setting. While the resulting schemes may not be competitive with the most efficient schemes selected by the NIST, they are practical and run on a standard laptop in acceptable time, which paves the way for practical advanced trapdoor-based constructions.

This work is the main object of the Chapter 2 of this manuscript.

- [BEP<sup>+</sup>21] Pauline Bert, Gautier Eberhart, Lucas Prabel, Adeline Roux-Langlois, Mohamed Sabt: Implementation of Lattice Trapdoors on Modules and Applications. PQCrypto 2021: 195-214

**Approximate Trapdoors and Applications.** Afterward, the rest of my work has naturally been directed towards the search for new methods to improve the efficiency of cryptographic schemes making use of gadget trapdoors.

Based on my work briefly presented in the previous paragraph, I worked on improving the efficiency of the identity-based encryption scheme. I study the possibility of using approximate trapdoors, with SIS and LWE assumptions, rather than the exact trapdoors of [MP12], in order to obtain a more efficient scheme, without drastically reducing the intrinsic security.

Approximate trapdoors were introduced by Chen, Genise and Mukherjee in [CGM19]. In their paper, they introduce the Approximate Inhomogeneous Short Integer Solution (AISIS) problem, which asks, given  $\mathbf{A} \in \mathcal{R}_q^{d \times m}$  and  $\mathbf{y} \in \mathcal{R}_q^d$ , to find a vector  $\mathbf{x} \in \mathcal{R}^m$  such that  $\|\mathbf{x}\| \leq \beta$  and such that there is a vector  $\mathbf{z} \in \mathcal{R}^d$  satisfying:  $\|\mathbf{z}\| \leq \alpha$  and  $\mathbf{Ax} = \mathbf{y} + \mathbf{z} \bmod q$ .

They showed that this problem is as hard as the original ISIS problem for some sets of parameters, and introduce the concept of approximate gadget trapdoors, related to this AISIS problem. In practice, approximate trapdoors are generated by dropping  $\ell$  entries corresponding to the small powers of  $b$  from the gadget matrix  $\mathbf{G}$  to get the following matrix:  $\mathbf{F} = \mathbf{I}_d \otimes \mathbf{f}^T = \mathbf{I}_d \otimes \begin{bmatrix} b^\ell & b^{\ell+1} & b^{\ell+2} & \dots & b^{k-1} \end{bmatrix} \in \mathcal{R}^{d \times d(k-\ell)}$ .

Doing so allows to reduce the sizes of the public matrix, the trapdoor and the preimage used in cryptographic constructions by a factor of 2 without losing too much on the concrete security of those schemes.

However, the use of approximate trapdoors leads to the appearance of error terms which must be taken care of in the decryption phase of encryption schemes. I worked on making use of those approximate trapdoors to design an identity-based encryption scheme that can handle this error term while using tag matrices. The resulting scheme I got is IND-sID-CPA secure in the standard model and resulted in an implementation showing that it is more efficient than the counterpart IBE using exact trapdoors.

- [IPR23] Malika Izabachène, Lucas Prabel and Adeline Roux-Langlois: Identity-Based Encryption From Lattices Using Approximate Trapdoors, ACISP 2023

**NTRU Lattices.** During my PhD, I was also interested in the study of NTRU lattices, in the efficiency (in terms of parameter sizes and running time) of cryptographic schemes making use of such a family of lattices, in implementation perspectives and in the concrete security of cryptographic schemes based on a variant of this hardness assumption.

This variant of NTRU, called iNTRU, was introduced in [GGH<sup>+</sup>19]. It can be seen as an inhomogeneous variant of NTRU. Basically, the matrix version of iNTRU asks to distinguish between a matrix  $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$  defined as  $\mathbf{A} = \mathbf{S}^{-1}(\mathbf{G} - \mathbf{E}) \bmod q$  (where  $\mathbf{S} \in \mathbb{Z}_q^{n \times n}$  is a random invertible matrix,  $\mathbf{E}$  is a low-norm matrix and  $\mathbf{G}$  is the gadget matrix) from a matrix chosen uniformly at random in  $\mathbb{Z}_q^{n \times m}$ .

The use of gadget trapdoors not being straightforwardly compatible with the NTRU hardness assumption, I used that particular variant of NTRU in order to build an identity-based encryption scheme based on the [MP12] paradigm, in order to obtain a more efficient

IBE in terms of timings and parameter sizes. This new scheme, whose security has been proven in the random oracle model, also resulted in an implementation, making it possible to assess its timings performance and to compare it with my previous IBE.

I was also interested in the concrete security of cryptographic schemes based on the iNTRU hardness assumption, as the literature on this subject is relatively new, this problem having been introduced recently. I was more particularly interested in overstretched attacks. Those attacks were introduced in [KF17] in the case of NTRU lattices, and applied to instances that need a large modulus  $q$  compared to the dimension  $2n$  of the lattice underlying the scheme.

In addition, I have also been interested more recently in the resolutions of NTRU equations, which are involved in particular in several cryptosystems such as Falcon [FHK<sup>+</sup>17] or BAT [FKP<sup>+</sup>22], during the key generation phase. My goal was to investigate techniques for effectively solving NTRU equations, when working with number fields that don't admit a subfields tower. In this case, solving these equations efficiently as presented in [PP19] using the field norm and the subfield structure is no longer possible, and other solutions must be found. I was also interested in the possibility of avoiding floating point arithmetic during the reduce phase of the NTRU equation solving algorithms, in order to get an efficient implementation.

- [IPR23] Malika Izabachène, Lucas Prabel and Adeline Roux-Langlois: Identity-Based Encryption From Lattices Using Approximate Trapdoors, ACISP 2023

Part of this work has also been published in the article [IPR23] cited above, and another part is still ongoing and has not yet been submitted for publication.

# PRELIMINARIES

---

In this first chapter, we introduce different mathematical and cryptographic notions, through some definitions and properties, on which we will rely multiple times in the rest of this thesis. In particular, we define what a lattice is, state some of its properties, the hard problems on which lattice-based cryptography is built, and introduce several tools that will be useful later in this manuscript. Next, we recall some mathematical definitions and properties in probability theory. Finally, we present some basic cryptographic notions and primitives which will reappear several times later, when presenting advanced cryptographic constructions.

## Contents

---

<b>1.1. Notations</b> .....	44
<b>1.2. Lattices</b> .....	45
1.2.1 First Definitions .....	45
1.2.2 Structured Lattices .....	48
1.2.3 Computational Problems .....	48
1.2.3.1 Worst-case Problems .....	49
1.2.3.2 Average-case Problems .....	50
1.2.3.3 Structured Variants .....	51
1.2.3.4 Worst-Case to Average-Case Reductions .....	53
<b>1.3. Probabilities</b> .....	54
1.3.1 Basic Definitions .....	54
1.3.2 Discrete Gaussians .....	55
1.3.2.1 Definitions .....	55
1.3.2.2 Gaussian Tailcut .....	56
1.3.2.3 The Gram-Schmidt Orthogonalization .....	56
1.3.2.4 Smoothing Parameter .....	57
1.3.2.5 Gaussian Samplers .....	57

1.3.3 Leftover Hash Lemma.....	59
<b>1.4. Cryptographic Primitives.....</b>	<b>59</b>
1.4.1 Basic Public-Key Primitives.....	60
1.4.1.1 Public-Key Encryption Scheme.....	60
1.4.1.2 Public-Key Digital Signature Scheme.....	62
1.4.2 More advanced primitives.....	64
1.4.2.1 Identity-Based Encryption Scheme.....	64
1.4.2.2 Encoding Messages with Full-Rank Differences.....	67

---

## 1.1 Notations

**Sets.** We denote by  $\mathbb{Z}$  and  $\mathbb{Z}_q$  the rings of integers and integers modulo  $q$ , by  $\mathbb{R}$  and  $\mathbb{C}$  the fields of real and complex numbers. Elements of these sets are denoted by standard lowercase letters, (column) vectors by bold lowercase letters, and matrices by bold uppercase letters.

**Norms.** The Euclidean inner product of two vectors  $\mathbf{x}, \mathbf{y} \in \mathbb{C}^n$  is defined as  $\langle \mathbf{x}, \mathbf{y} \rangle = \sum_{i=1}^n x_i y_i$ . The norm  $\|\cdot\|_2$  (or just  $\|\cdot\|$ ) denotes the Euclidean norm, induced by the Euclidean inner product  $\langle \cdot, \cdot \rangle$ . The norm of a vector over  $\mathbb{Z}_q$  is the norm of the corresponding vector over  $\mathbb{Z}$  with entries in  $\{-\lfloor q/2 \rfloor, \dots, \lfloor q/2 \rfloor\}$ , the norm of a polynomial  $a = \sum_{i=0}^{n-1} a_i X^i$  is the norm of the vector  $(a_0, \dots, a_{n-1})$ , and the norm of a matrix is the maximum norm of its column vectors. We also denote  $\|\cdot\|_\infty$  the infinity norm, defined as  $\|\mathbf{x}\|_\infty = \max_i |x_i|$  for a vector  $\mathbf{x}$  and  $\|\mathbf{A}\|_\infty = \max_{i,j} |a_{i,j}|$  for a matrix  $\mathbf{A} = (a_{i,j})_{i,j}$ .

**Asymptotic Notations.** We use the standard Landau notations. A function  $f$  is negligible when  $f(n) = o(n^{-c})$  for all  $c > 0$  as  $n \rightarrow \infty$ . An event happens with overwhelming probability if its probability of not happening is negligible.

**Distributions.** If  $x$  is sampled from a distribution  $D$ , we write  $x \leftarrow D$ . The uniform distribution over a finite set  $S$  is denoted  $\mathcal{U}(S)$  and if  $x$  is sampled uniformly from a set  $S$ , we write  $x \xleftarrow{\$} S$ . Two distributions  $D_0$  and  $D_1$  over the same countable domain  $\Omega$  are said to be statistically indistinguishable if their statistical distance  $\Delta(D_0, D_1) = \frac{1}{2} \sum_{\omega \in \Omega} |D_0(\omega) - D_1(\omega)|$  is negligible. They are computationally indistinguishable if no

probabilistic polynomial time algorithm can distinguish them with non-negligible advantage.

**Matrices.** We denote  $\mathbf{I}_n$  (or just  $\mathbf{I}$  if the size is clear from context) the identity matrix of size  $n \times n$ . A symmetric matrix  $\mathbf{M} \in \mathbb{R}^{n \times n}$  is said to be positive definite (resp. positive semidefinite) if for all nonzero  $\mathbf{x} \in \mathbb{R}^n$  we have  $\mathbf{x}^T \mathbf{M} \mathbf{x} > 0$  (resp.  $\mathbf{x}^T \mathbf{M} \mathbf{x} \geq 0$ ), in which case we write  $\mathbf{M} \succ 0$  (resp.  $\mathbf{M} \succeq 0$ ). Positive semidefiniteness induces a partial order on  $\mathbb{R}^{n \times n}$ , as we say that  $\mathbf{M} \succeq \mathbf{N}$  when  $\mathbf{M} - \mathbf{N} \succeq 0$ . For simplicity of notation, we write  $\mathbf{M} \succeq \eta \mathbf{I}$  instead of  $\mathbf{M} \succeq \eta \mathbf{I}$ , for a real  $\eta \geq 0$ .

## 1.2 Lattices

This section contains general definitions and properties concerning Euclidean and structured lattices. It will also present some computational problems based on lattices.

### 1.2.1 First Definitions

Informally speaking, a lattice  $\Lambda$  can be seen as a periodic arrangement of points forming a "grid" (see Figure 1.1). More formally, it is a discrete subgroup of  $\mathbb{R}^m$ .

**Definition 1 (Lattice)** Given a set of linearly independent vectors  $\mathbf{B} = \{\mathbf{b}_1, \dots, \mathbf{b}_k\} \subset \mathbb{R}^m$ , we call lattice with basis  $\mathbf{B}$  the following set:

$$\mathcal{L}(\mathbf{B}) := \left\{ \sum_{i=1}^k \lambda_i \mathbf{b}_i, \lambda_i \in \mathbb{Z} \right\}.$$

The dimension of the lattice is  $m$  and its rank is  $k$ .

When  $k = m$ , the lattice is said to be full-rank.

For  $m \geq 2$ , a lattice  $\Lambda$  admits infinitely many bases. Indeed, if  $\mathbf{B}$  is a basis of the  $m$ -dimensional lattice  $\Lambda$ , then any  $\mathbf{UB}$  where  $\mathbf{U} \in \mathbb{Z}^{m \times m}$  is unimodular is also a basis of  $\Lambda$ .

**Definition 2 (Sublattice)** Let  $\Lambda, \Lambda' \subset \mathbb{R}^m$  be lattices. We say that  $\Lambda'$  is a sublattice of  $\Lambda$  if  $\Lambda' \subset \Lambda$ .

In cryptographic applications, we will mostly work with  $q$ -ary lattices, meaning that

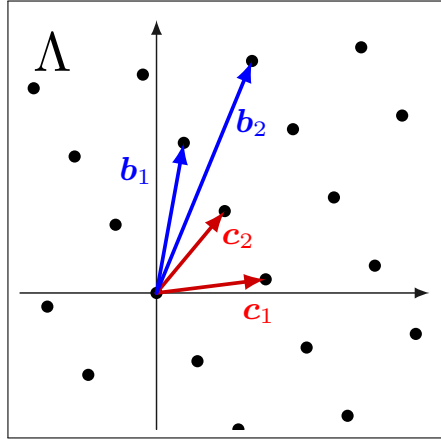


Figure 1.1 – A 2-dimensional lattice  $\Lambda$  with two of its basis:  $\mathbf{B} = (\mathbf{b}_1, \mathbf{b}_2)$  and  $\mathbf{B}' = (\mathbf{c}_1, \mathbf{c}_2)$ .

they have  $q\mathbb{Z}^m = \{q\mathbf{x} \mid \mathbf{x} \in \mathbb{Z}^m\}$  as a sublattice. In practice, this enables us to carry out all of our computations using integers modulo  $q$ . For  $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$  and  $\mathbf{u} \in \mathbb{Z}_q^n$ , we will regularly make use of the following  $m$ -dimensional  $q$ -ary lattices:

$$\Lambda_q^\perp(\mathbf{A}) = \{ \mathbf{x} \in \mathbb{Z}^m \mid \mathbf{A}\mathbf{x} = \mathbf{0} \pmod{q} \}$$

and its coset

$$\Lambda_q^{\mathbf{u}}(\mathbf{A}) = \{ \mathbf{x} \in \mathbb{Z}^m \mid \mathbf{A}\mathbf{x} = \mathbf{u} \pmod{q} \}.$$

**Definition 3 (Dual of a lattice)** Given a lattice  $\Lambda \subset \mathbb{R}^m$ , the dual lattice of  $\Lambda$  is the lattice

$$\Lambda^* = \{ \mathbf{x} \in \mathbb{R}^m \mid \forall \mathbf{y} \in \Lambda, \langle \mathbf{x}, \mathbf{y} \rangle \in \mathbb{Z} \}.$$

Moreover, if  $\mathbf{B}$  is a basis of the lattice  $\Lambda$ , then  $\mathbf{B}(\mathbf{B}^T \mathbf{B})^{-1}$  is a basis of the dual lattice  $\Lambda^*$ .

**Definition 4 (First minimum)** Given a lattice  $\Lambda \subset \mathbb{R}^m$ , the first minimum  $\lambda_1(\Lambda)$  of the lattice  $\Lambda$  is the length of a shortest non-zero vector in  $\Lambda$ :

$$\lambda_1(\Lambda) = \min_{\mathbf{x} \in \Lambda \setminus \{\mathbf{0}\}} \|\mathbf{x}\|_2.$$

We can also define the  $i$ -th successive minimum of a lattice (see Figure 1.2), which is the smallest real  $r$  such that the lattice  $\Lambda$  has  $i$  linearly independent vectors of norms at most  $r$ .

**Definition 5 (Successive Minimum)** Given a lattice  $\Lambda \subset \mathbb{R}^m$ , the  $i$ -th successive minimum  $\lambda_i(\Lambda)$  of the lattice  $\Lambda$  is the following quantity:

$$\lambda_i(\Lambda) = \min\{r \mid \dim(\text{span}(\Lambda \cap \mathcal{B}_r)) \geq i\}.$$

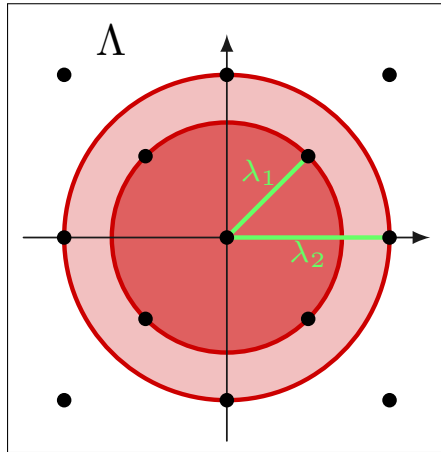


Figure 1.2 – The first two successive minima of a 2-dimensional lattice  $\Lambda$ .

The determinant  $\det(\Lambda)$  of a lattice  $\Lambda$ , defined below, is independent of the choice of the basis  $\mathbf{B}$ , and thus constitutes an invariant of  $\Lambda$ .

**Definition 6 (Determinant of a lattice)** Let  $\Lambda = \mathcal{L}(\mathbf{B}) \subset \mathbb{R}^m$  be a lattice of rank  $m$ . The determinant of  $\Lambda$  is the following quantity:

$$\det(\Lambda) = \sqrt{\det(\mathbf{B}\mathbf{B}^T)}.$$

Minkowski's theorem allows to bound the value of the first minimum of a lattice given its determinant:

**Theorem 1 (Minkowski's theorem)** Let  $\Lambda = \mathcal{L}(\mathbf{B}) \subset \mathbb{R}^m$  be a lattice of rank  $m$ . Then  $\lambda_1(\Lambda) \leq \sqrt{m} \det(\Lambda)^{1/m}$ .



## 1.2.2 Structured Lattices

In the early days of lattice-based cryptography, the schemes were not practical due to their inefficiency. Compared to more recent schemes, they required significantly larger sizes and computational times for equivalent security. To address this problem, cryptosystems have leveraged structured lattices in order to gain in efficiency. These lattices have an additional algebraic structure, which allows for a more compact representation in terms of storage and faster operations. By using structured lattices, lattice-based cryptographic schemes have become much more practical and efficient.

More precisely, ideal lattices (introduced in [PR06; SST<sup>+</sup>09; LPR10]) and module lattices (introduced in [LS15; HPS98; SS13]) are particular lattices that have a polynomial structure. Briefly, if  $\mathcal{K}$  is a number field of degree  $n$  with  $\mathcal{R}$  its ring of integers, the canonical embedding  $\sigma$  defines a field homomorphism  $\sigma : \mathcal{K} \rightarrow \mathbb{R}^n$ . Then, a  $\mathcal{R}$ -module  $M \subset \mathcal{K}^d$  of rank  $d$  defines a module lattice  $\sigma(M) \subset \mathbb{R}^{dn}$ . Similarly, any ideal  $I$  over  $\mathcal{R}$ , which is a module of rank 1, defines an ideal lattice  $\sigma(I) \subset \mathbb{R}^n$ . When constructing cryptographic schemes, we will mostly consider the ones that are based on the rings  $\mathcal{R} = \mathbb{Z}[X]/\langle X^n + 1 \rangle$  and  $\mathcal{R}_q = \mathbb{Z}_q[X]/\langle X^n + 1 \rangle$ , where  $n$  is a power of two and  $q$  is a prime. They are sublattices of the full lattice  $\mathcal{R}^m$ , itself isomorphic to the integer lattice  $\mathbb{Z}^{nm}$ .

Consequently, when working with structured lattices, using efficient polynomial multiplication algorithms such as the Fast Fourier Transform (FFT) allows performing matrix-matrix multiplication in  $\tilde{O}(m^2)$  (instead of  $O(m^3)$  with unstructured lattices) and  $\tilde{O}(m)$  (instead of  $O(m^2)$  with unstructured lattices) arithmetic operations.

Therefore, working with lattices that possess a polynomial structure allows to effectively speed up computations and to reduce storage costs compared to constructions relying on Euclidean lattices.

## 1.2.3 Computational Problems

Lattice-based cryptography now constitutes a viable candidate for replacing number theoretic cryptography in the future. Hardness assumptions on lattices are conjecturally quantum resistant, whereas the discrete logarithm and factorization problems are known to be solvable in polynomial time in a quantum setting [Sho94]. Worst-case to average-case reductions from fundamental lattice problems (relaxations of NP-hard problems) also provide strong theoretical security guarantees for lattice-based primitives.

### 1.2.3.1 Worst-case Problems

We begin by defining some important problems which naturally appear when studying lattices. Those problems are hard to solve on classical computers, and it remains still unknown if quantum computers can solve them substantially faster. This is in contrast to problems such as discrete logarithms and factoring, which can be solved in polynomial time by quantum computers [Sho94].

**Definition 7 (Shortest Vector Problem (SVP))** Let  $\Lambda$  be a lattice. The Shortest Vector Problem (SVP) asks to find a lattice vector  $\mathbf{x} \in \Lambda$  such that  $\|\mathbf{x}\| = \lambda_1(\Lambda)$ .

The SVP problem is illustrated in Figure 1.3 for the two-dimensional case.

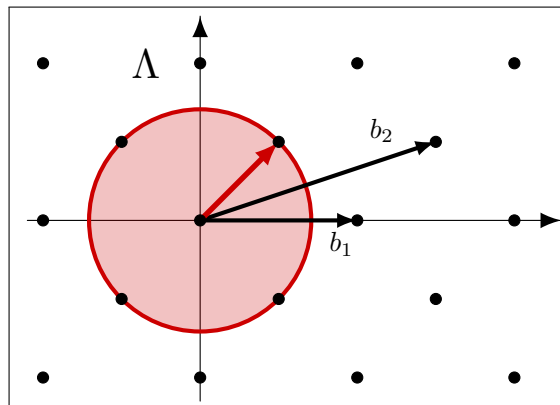


Figure 1.3 – A 2-dimensional lattice  $\Lambda$  with basis  $\mathbf{B} = (\mathbf{b}_1, \mathbf{b}_2)$  and a **shortest vector**.

This problem admits approximate and decisional variants.

**Definition 8 (Approximate Shortest Vector Problem ( $\text{SVP}_\gamma$ ))** Let  $\Lambda$  be a lattice and  $\gamma \geq 1$  the approximate factor. The Approximate Shortest Vector Problem ( $\text{SVP}_\gamma$ ) asks to find a lattice vector  $\mathbf{x} \in \Lambda$  such that  $\|\mathbf{x}\| \leq \gamma \cdot \lambda_1(\Lambda)$ .

**Definition 9 (Decisional Approximate Shortest Vector Problem ( $\text{GapSVP}_\gamma$ ))**

Let  $\Lambda$  be a lattice,  $\delta > 0$  be a real number and  $\gamma \geq 1$  the approximate factor. The Decisional Approximate Shortest Vector Problem ( $\text{GapSVP}_\gamma$ ) asks to distinguish between the two following cases:

- $\lambda_1(\Lambda) \leq \delta$ ;
- $\lambda_1(\Lambda) > \gamma \cdot \delta$ .

In the case where  $\delta < \lambda_1(\Lambda) \leq \gamma \cdot \delta$ , any answer is correct.

**1.2.3.2 Average-case Problems**

However, problems such as  $\text{SVP}$  or  $\text{SVP}_\gamma$  are worst-case problems, meaning they are hard to solve in the worst-case, and not for any instance of a lattice. Therefore, they are not particularly well-suited for designing cryptographic protocols. On the contrary, average-case problems will be preferred when it comes to building cryptographic schemes, because with such problems, random instances are hard to solve.

Therefore, let's begin by defining some average-case problems, which naturally appear when studying cryptographic constructions. The Short Integer Solution (SIS) problem was introduced in [Ajt96] and the Learning With Error (LWE) problem was introduced in [Reg05].

**Definition 10 (Short Integer Solution ( $\text{SIS}_{n,q,m,\beta}$ ))** Given a uniformly random matrix  $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ , the Short Integer Solution ( $\text{SIS}_{n,q,m,\beta}$ ) problem asks to find a non-zero vector  $\mathbf{x} \in \mathbb{Z}^m$  such that  $\mathbf{Ax} = \mathbf{0} \pmod{q}$ .

**Definition 11 (Learning With Error (Decisional-LWE $_{n,q,m,D_{\mathbb{Z},\alpha q}$ }))** Given a uniformly random matrix  $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$  and a vector  $\mathbf{b}^T = \mathbf{s}^T \mathbf{A} + \mathbf{e}^T$  where  $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$  and  $\mathbf{e} \leftarrow D_{\mathbb{Z}^m, \alpha q}$ :

- The Decisional Learning With Error (Decisional-LWE $_{n,q,m,D_{\mathbb{Z},\alpha q}}$ ) problem asks to distinguish between  $(\mathbf{A}, \mathbf{b}^T)$  and  $(\mathbf{A}, \mathbf{u})$  where  $\mathbf{u}$  is uniformly distributed over  $\mathbb{Z}_q^m$ .
- The Search Learning With Error (LWE $_{n,q,m,D_{\mathbb{Z},\alpha q}}$ ) problem asks to find  $\mathbf{s}$ .

SIS and LWE do benefit from worst-case to average-case reductions, and are thus also believed to be quantum resistant. Those reductions state that if the security of a

cryptosystem is based on an average-case problem, attacking random instances of this cryptosystem is at least as hard as solving all instances of the underlying worst-case lattice problem. Thus, average-case problems such as SIS and LWE are particularly well-suited for designing cryptographic schemes.

Another standard problem in lattice-based cryptography is the NTRU hardness assumption [HPS98].

**Definition 12 (NTRU)** Given an integer polynomial  $\phi$  defining the ring  $\mathcal{R} = \mathbb{Z}[X]/(\phi)$ , a modulus  $q \in \mathbb{Z}$  and a polynomial  $h = f \cdot g^{-1} \pmod{q}$  where  $f, g \in \mathcal{R}$  are polynomials with small coefficients, the NTRU problem asks to recover the pair  $(f, g)$ .

We don't know any worst-case to average-case reduction for the NTRU problem, as opposed to the SIS and LWE, but the NTRU hardness assumption having been widely studied since the original article [HPS98], it is now seen as a standard lattice problem. For particular polynomials  $\phi$  (such as powers-of-two cyclotomic polynomials for example), the NTRU problem is believed to be hard to solve for a quantum computer.

### 1.2.3.3 Structured Variants

A cryptographic scheme whose security is based on the hardness of the SIS and LWE problems defined in Section 1.2.3.2 however presents the drawback of being usually less efficient in terms of parameter sizes than the classical counterpart of this scheme, whose hardness is based on the discrete logarithm of factorization problems. Indeed, in such a scheme, the public key usually includes a matrix  $\mathbf{A}$ , whose size is quadratic in the dimension  $n$ , with  $n$  at least as large as the security parameter of the scheme. The most used operations in lattice-based cryptosystems being matrix-matrix multiplication, whose running time is in  $O(n^3)$  and matrix-vector multiplication, whose running time is in  $O(n^2)$ , this has a major impact on the efficiency of the cryptosystem.

However, to address this efficiency concern, structured variants of LWE and SIS have been introduced and widely studied, such as Ring-SIS [PR06], Ring-LWE [SST<sup>+</sup>09; LPR10] or Module-SIS and Module-LWE [LS15]. These problems deal with structured lattices (see Section 1.2.2), such as *ideal* or *module* lattices. Ideal lattices are a special subset of lattices that possess interesting computational properties. Basically, when working with ideal lattices, we work in a ring of polynomials  $\mathcal{R}$ , often chosen as  $\mathcal{R} = \mathbb{Z}[X]/(X^n + 1)$  instead of working over the ring of integers. Therefore, vectors of ideal lattices correspond to

polynomials and a polynomial, modulo  $X^n + 1$ , can be seen as a vector of dimension  $n$ . An ideal lattice is then represented by the skew-circulant matrix of multiplication by a polynomial  $a \in \mathbb{Z}[X]/(X^n + 1)$ . Using efficient polynomial multiplication algorithms such as the Fast Fourier Transform (FFT) then allows performing matrix-matrix multiplication in  $\tilde{O}(n^2)$  (instead of  $O(n^3)$  with unstructured lattices) and  $\tilde{O}(n)$  (instead of  $O(n^2)$  with unstructured lattices) arithmetic operations.

Therefore, as in most practical lattice-based constructions [ADP<sup>+</sup>16; BFR<sup>+</sup>18; DKL<sup>+</sup>18; FHK<sup>+</sup>17], we consider rings of the form  $\mathcal{R} = \mathbb{Z}[X]/\langle X^n + 1 \rangle$  and  $\mathcal{R}_q = \mathbb{Z}_q[X]/\langle X^n + 1 \rangle$ , where  $n$  is a power of two and  $q$  a prime modulus. The polynomial  $X^n + 1$  is the cyclotomic polynomial of order  $2n$ , and  $\mathcal{R}$  is the corresponding cyclotomic ring.

Let us begin by defining the Ring-SIS and Ring-LWE structured variants of SIS and LWE.

**Definition 13 (Ring-SIS <sub>$n,m,q,\beta$</sub> )** Given a uniformly random  $\mathbf{a} \in \mathcal{R}_q^m$ , find a vector  $\mathbf{x} \in \mathcal{R}^m$  such that  $\mathbf{a}^T \mathbf{x} = \mathbf{0} \pmod{q}$ , and  $0 < \|\mathbf{x}\| \leq \beta$ .

**Definition 14 (Decision Ring-LWE <sub>$n,q,\sigma$</sub> )** Given a uniformly random  $\mathbf{a} \in \mathcal{R}_q^m$  and the vector  $\mathbf{b} = s\mathbf{a} + \mathbf{e} \pmod{q}$ , where  $s \xleftarrow{\$} \mathcal{R}_q$  and  $\mathbf{e} \leftarrow D_{\mathcal{R}^m, \sigma}$ , distinguish the distribution of  $(\mathbf{a}, \mathbf{b})$  from the uniform distribution over  $\mathcal{R}_q^m \times \mathcal{R}_q^m$ .

The module variants, generalizing Ring-SIS and Ring-LWE, were introduced in [LS15]. The parameter  $d$  corresponds to the rank of the module, and  $nd$  is the dimension of the corresponding module lattice.

**Definition 15 (Module-SIS <sub>$n,d,m,q,\beta$</sub> )** Given a uniformly random  $\mathbf{A} \in \mathcal{R}_q^{d \times m}$ , find a vector  $\mathbf{x} \in \mathcal{R}^m$  such that  $\mathbf{A}\mathbf{x} = \mathbf{0} \pmod{q}$ , and  $0 < \|\mathbf{x}\| \leq \beta$ .

**Definition 16 (Decision Module-LWE <sub>$n,d,q,\sigma$</sub> )** Given a uniform  $\mathbf{A} \in \mathcal{R}_q^{m \times d}$  and the vector  $\mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e} \pmod{q}$ , where  $\mathbf{s} \xleftarrow{\$} \mathcal{R}_q^d$  and  $\mathbf{e} \leftarrow D_{\mathcal{R}^m, \sigma}$ , distinguish the distribution of  $(\mathbf{A}, \mathbf{b})$  from the uniform distribution over  $\mathcal{R}_q^{m \times d} \times \mathcal{R}_q^m$ .

If we specialized the previous definitions of the module variants with  $d = 1$ , we get the Ring-LWE and Ring-SIS problems.

We also rely on variants of these problems, which have been proven to be as hard. The inhomogeneous variant of Module-SIS, called Module-ISIS, consists in finding a small  $\mathbf{x}$  such that  $\mathbf{A}\mathbf{x} = \mathbf{u}$ , given the vector  $\mathbf{u}$ . In the normal form of Module-LWE, the secret

$\mathbf{s}$  follows the same distribution  $D_{\mathcal{R}^m, \sigma}$  as the error  $\mathbf{e}$ .

#### 1.2.3.4 Worst-Case to Average-Case Reductions

One of the major points of interest in lattice-based cryptography is the existence of worst-case to average-case reductions. Indeed, Ajtai showed in [Ajt96] that it is possible to construct cryptographic schemes whose security is based on the worst-case hardness of lattice problems. Basically, if the security of a cryptosystem is based on an average-case problem (such as SIS or LWE), then attacking this cryptosystem will be as hard as solving all instances of the underlying worst-case lattice problem. That makes intermediate lattice problems such as SIS and LWE, which benefit from the existence of worst-case to average-case reductions, particularly well-suited for designing cryptographic schemes. Therefore, those reductions provide strong security guarantees for lattice-based cryptography.

First, concerning the SIS problem, a wide range of articles have progressively shown that several standard worst-case problems over Euclidean lattices reduce to the SIS problem ([Ajt96; MR07; GPV08]). We give a simplified result below.

**Theorem 2** For any  $m = \text{poly}(n)$ ,  $\beta > 0$  and  $q \geq \beta \cdot \text{poly}(n)$ , solving  $\text{SIS}_{n,q,m,\beta}$  with non-negligible probability is as least as hard as solving  $\text{GapSVP}_\gamma$  on arbitrary  $n$ -dimensional lattices with overwhelming probability for some  $\gamma = \beta \cdot \text{poly}(n)$ .

Similarly, such reductions also exist for the LWE problem. Indeed, this problem has been shown to enjoy both quantum [Reg05] and classical [Pei09; BLP<sup>+</sup>13] worst-case to average-case reductions for appropriate parameters. We state the result from [Reg05] below.

**Theorem 3 ([Reg05])** For any  $m = \text{poly}(n)$ ,  $q \leq 2^{\text{poly}(n)}$  and  $\alpha q \geq 2\sqrt{n}$ , solving  $\text{LWE}_{n,q,m,D_{\mathbb{Z},\alpha q}}$  is as least as hard as quantumly solving  $\text{GapSVP}_\gamma$  on arbitrary  $n$ -dimensional lattices for some  $\gamma = \tilde{O}(n/\alpha)$ .

While this result only shows that having an efficient algorithm (classical or quantum) solving LWE implies having an efficient quantum algorithm solving  $\text{GapSVP}_\gamma$ , a classical reduction was first given in [Pei09] (but requiring an exponentially large modulus  $q$ ), and another classical reduction was shown in [BLP<sup>+</sup>13], only requiring polynomially large modulus.

Finally, structured variants of SIS and LWE also enjoy worst-case to average-case reductions for suitable parameter choices, where the underlying lattice problems are specialized

to ideal (for ring variants) and module (for module variants) lattices (see [LPR10; LS15]).

## 1.3 Probabilities

This section introduces different notions and tools of probability theory, through some definitions and properties, which we will use extensively throughout this thesis.

### 1.3.1 Basic Definitions

We begin by introducing the statistical distance, a tool which allows to tell how different two distributions are from one another.

**Definition 17** Let  $P$  and  $Q$  be two distributions over a countable domain  $E$ . The statistical distance between  $P$  and  $Q$ , denoted  $\Delta(P, Q)$  is defined as:

$$\Delta(P, Q) = \frac{1}{2} \sum_{x \in E} |P(x) - Q(x)|.$$

The statistical distance  $\Delta$  is a distance, and it is widely used in cryptography. We recall some of its basic properties here.

**Proposition 1** Let  $P_i$  and  $Q_i$  be some distributions over a countable domain  $E$  for  $i \in \{1, 2\}$ . The statistical distance verifies the following properties:

- Sub-additivity:  $\Delta(P_0 \times P_1, Q_0 \times Q_1) = \Delta(P_0, Q_0) + \Delta(P_1, Q_1)$ .
- Preservation under any transformation: for every function  $f$ , it holds that  $\Delta(f(P_1), f(Q_1)) \leq \Delta(P_1, Q_1)$ .
- For any measurable event  $X$ , we have  $Q_1(X) \geq P_1(X) - \Delta(P_1, Q_1)$ .

The last property shows that if a given cryptographic scheme has a distribution  $P$  as input and if we call  $X$  the event of an attacker successfully breaking the scheme, then if  $\Delta(P, Q)$  is negligible, the scheme will be as hard to break when using distribution  $Q$  instead of  $P$ . This will be extensively used later in security proofs of our constructions.

Another distance, called the *max-log* distance, which also measures the closeness between two distributions was introduced in [MW17]. This metric is also quite simple to use and compute, and we will make use of it in Chapter 3 and 4 to get sharper security

estimates than if we had used the statistical distance. We define the max-log distance in Definition 18.

**Definition 18** Let  $P$  and  $Q$  be two distributions over a countable domain  $E$ . The max-log distance between  $P$  and  $Q$ , denoted  $\Delta_{\text{ML}}(P, Q)$  is defined as:

$$\Delta_{\text{ML}}(P, Q) = \max_{x \in E} |\log P(x) - \log Q(x)|.$$

We can show that the max-log distance is indeed a distance. This metric also verifies the sub-additivity and the preservation under any transformation properties.

## 1.3.2 Discrete Gaussians

### 1.3.2.1 Definitions

The discrete Gaussian distribution of center  $\mathbf{c} \in \mathbb{R}^n$  and parameter  $\sigma > 0$  over a full-rank lattice  $\Lambda \subset \mathbb{Z}^n$  is denoted  $D_{\Lambda, \sigma, \mathbf{c}}$ . It is the probability distribution over  $\Lambda$  such that each  $\mathbf{x} \in \Lambda$  is assigned a probability proportional to  $\rho_{\sigma, \mathbf{c}}(\mathbf{x}) = \exp(-\frac{\pi \|\mathbf{x} - \mathbf{c}\|^2}{\sigma^2})$ .

For a positive definite matrix  $\Sigma \in \mathbb{R}^{n \times n}$ , we also define the (skewed) density  $\rho_{\sqrt{\Sigma}, \mathbf{c}}(\mathbf{x}) = \exp(-\pi(\mathbf{x} - \mathbf{c})^T \Sigma^{-1}(\mathbf{x} - \mathbf{c}))$ , and the corresponding discrete Gaussian distribution of center  $\mathbf{c}$  and covariance  $\Sigma$  denoted  $D_{\Lambda, \sqrt{\Sigma}, \mathbf{c}}$ .

We represent in Figure 1.4 the graph of a continuous and discrete Gaussian in dimension  $n = 1$ .

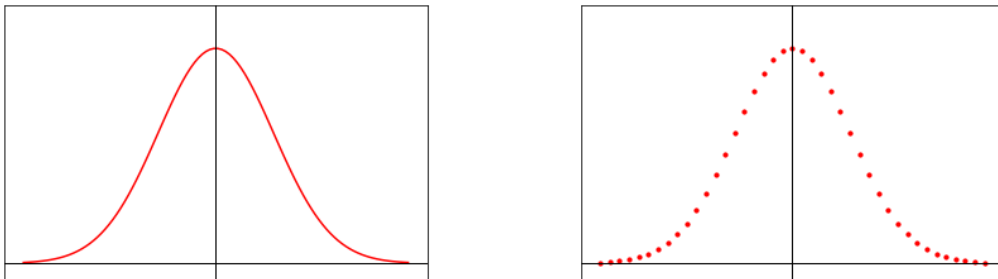


Figure 1.4 – Continuous and discrete Gaussian distributions in dimension  $n = 1$ .



### 1.3.2.2 Gaussian Tailcut

We denote by  $t$  the tailcut of the discrete Gaussian of parameter  $\sigma$ . It is a positive number such that samples from  $D_{\mathbb{Z},\sigma}$  land outside of  $[-t\sigma, t\sigma]$  only with negligible probability. We choose it using the fact that  $\Pr_{x \leftarrow D_{\mathbb{Z},\sigma}} [|x| > t\sigma] \leq \operatorname{erfc}(t/\sqrt{2})$ , where  $\operatorname{erfc}(x) = 1 - \frac{2}{\pi} \int_0^x \exp^{-u^2} du$ . This generalizes to higher dimensions using the following lemma.

**Lemma 1 ([MR07, Lemma 4.4])** For any  $n$ -dimensional lattice  $\Lambda$ , vector  $\mathbf{c} \in \mathbb{R}^n$ , reals  $0 < \varepsilon < 1$  and  $\sigma \geq \eta_\varepsilon(\Lambda)$ , if  $x$  is distributed according to  $D_{\Lambda,\sigma,\mathbf{c}}$ , then we have  $\Pr [\|\mathbf{x} - \mathbf{c}\| > \sigma\sqrt{n}] \leq \frac{1+\varepsilon}{1-\varepsilon} \cdot 2^{-n}$ .

### 1.3.2.3 The Gram-Schmidt Orthogonalization

Having an orthogonal basis is useful to decompose a vector. Therefore, when working with a basis  $\mathbf{B}$ , it is often desirable to compute an orthogonal basis  $\tilde{\mathbf{B}}$  which generates the same linear subspace. The Gram-Schmidt Orthogonalization (GSO) is an algorithm that allows to do so. Concerning lattices, if  $\mathbf{B}$  is the basis of a lattice  $\Lambda$ ,  $\tilde{\mathbf{B}}$  will not necessarily be a basis of the same lattice  $\Lambda$ . Nonetheless, the Gram-Schmidt of a lattice basis is still a valuable analysis tool, as we will see multiple times in this manuscript.

We give the definition of the Gram-Schmidt Orthogonalization (GSO) of a matrix in Definition 19.

#### Definition 19 (Gram-Schmidt Orthogonalization)

Let  $\mathbf{B} = [\mathbf{b}_1, \dots, \mathbf{b}_d] \in \mathbb{R}^{m \times d}$  be a matrix composed of  $d$  linearly independent vectors. We call Gram-Schmidt Orthogonalization (GSO) of  $\mathbf{B}$  the unique matrix  $\tilde{\mathbf{B}} = [\tilde{\mathbf{b}}_1, \dots, \tilde{\mathbf{b}}_d] \in \mathbb{R}^{m \times d}$  verifying:

$$\begin{aligned} \tilde{\mathbf{b}}_1 &= \mathbf{b}_1, \\ \tilde{\mathbf{b}}_i &= \pi_{\{\mathbf{b}_1, \dots, \mathbf{b}_{i-1}\}^\perp}(\mathbf{b}_i) \quad \text{for } 2 \leq i \leq d. \end{aligned}$$

For  $1 \leq i \leq d$ , we also write  $\pi_i$  the orthogonal projection over  $\{\mathbf{b}_1, \dots, \mathbf{b}_{i-1}\}^\perp$ .

Thus,  $\tilde{\mathbf{b}}_i$  is the  $i$ -th Gram-Schmidt vector of  $\tilde{\mathbf{B}}$ , which is defined as  $\tilde{\mathbf{b}}_i = \pi_i(\mathbf{b}_i)$ .

We also denote  $\mathbf{B}_{[l:r]}$  the projected block  $[\pi_l(\mathbf{b}_l), \dots, \pi_l(\mathbf{b}_{r-1})]$  and  $\mathcal{L}_{[l:r]} := \mathcal{L}(\mathbf{B}_{[l:r]})$  the lattice spanned by  $\mathbf{B}_{[l:r]}$ , whose rank is  $r - l$ . The  $\tilde{\mathbf{b}}_i$  will also be denoted  $\mathbf{b}_i^*$ .

The following Algorithm 1 computes the GSO  $\tilde{\mathbf{B}}$  of a basis  $\mathbf{B}$ .

---

**Algorithm 1** GramSchmidtOrthogonalization( $\mathbf{B}$ )

---

**Input:**  $\mathbf{B} = \{\mathbf{b}_1, \dots, \mathbf{b}_n\}$  a basis**Output:**  $\tilde{\mathbf{B}}$  the GSO of  $\mathbf{B}$ 

```

1: for  $i = 1, \dots, n$  do
2:   Let  $\tilde{\mathbf{b}}_i \leftarrow \mathbf{b}_i$ 
3:   for  $j = 1, \dots, i - 1$  do
4:     Compute  $\mu_{i,j} = \frac{\langle \mathbf{b}_i, \tilde{\mathbf{b}}_j \rangle}{\|\tilde{\mathbf{b}}_j\|^2}$ 
5:     Let  $\tilde{\mathbf{b}}_i \leftarrow \tilde{\mathbf{b}}_i - \mu_{i,j} \tilde{\mathbf{b}}_j$ 
6: return  $\tilde{\mathbf{B}} = \{\tilde{\mathbf{b}}_1, \dots, \tilde{\mathbf{b}}_n\}$ 

```

---

**1.3.2.4 Smoothing Parameter**

The smoothing parameter  $\eta_\varepsilon(\Lambda)$  of a lattice  $\Lambda$  was introduced in [MR07]. Informally, this quantity represents the smallest Gaussian parameter  $\sigma > 0$  such that the discrete Gaussian distribution  $D_{\Lambda, \sigma, c}$  "behaves" like a continuous Gaussian.

**Definition 20** ([MR07, Definition 3.1]) For a  $n$ -dimensional lattice  $\Lambda$  and a real  $\varepsilon > 0$ , the smoothing parameter  $\eta_\varepsilon(\Lambda)$  of  $\Lambda$  is defined as the smallest real  $s > 0$  such that  $\rho_{1/s}(\Lambda^* \setminus \{\mathbf{0}\}) \leq \varepsilon$ .

The following lemma allows to find a lower bound of this smoothing parameter.

**Lemma 2** ([GPV08, Lemma 3.1]) Let  $\Lambda \subset \mathbb{R}^n$  be a lattice with basis  $\mathbf{B}$ , and  $\tilde{\mathbf{B}}$  the Gram-Schmidt orthogonalization of  $\mathbf{B}$ . Then, for any  $\varepsilon > 0$ , we have

$$\eta_\varepsilon(\Lambda) \leq \|\tilde{\mathbf{B}}\| \cdot \sqrt{\log(2n(1 + 1/\varepsilon))/\pi}.$$

**1.3.2.5 Gaussian Samplers**

In this section, we will briefly review some basic Gaussian samplers which aim at sampling discrete Gaussian distribution over a lattice.

**The Klein Sampler.** Klein introduced in [Kle00] a Gaussian sampler, whose goal was to solve a variant of the closest vector problem. This sampler was used in another context in [GPV08] to construct a signature scheme. This sampling algorithm is a randomized variant

of Babai *nearest-plane algorithm* [Bab86], a deterministic algorithm that, given a lattice  $\Lambda$  and a point  $\mathbf{c}$ , aims at finding a point  $\mathbf{x} \in \Lambda$  close to  $\mathbf{c}$ . The only difference between the two algorithms comes from the rounding step, which is replaced by a randomized rounding in the case of the Klein's algorithm, thus turning the Babai's solver into a Gaussian sampler. Klein's sampler is described in Algorithm 2.

---

**Algorithm 2** KleinSampler( $\mathbf{B}, \mathbf{c}, \sigma$ ) for sampling  $\mathbf{x} \leftarrow D_{\mathcal{L}(\mathbf{B}), \sigma, \mathbf{c}}$

---

```

1: function KleinSampler( $\mathbf{B} = \{\mathbf{b}_1, \dots, \mathbf{b}_n\} \in \mathbb{Z}^{m \times n}, \mathbf{c} \in \mathbb{R}^m, \sigma > 0$ )
2:    $\mathbf{c}_n \leftarrow \mathbf{c}, \mathbf{x}_n \leftarrow \mathbf{0}$ 
3:   for  $i = n$  to 1 do
4:      $d_i \leftarrow \frac{\langle \mathbf{c}_i, \tilde{\mathbf{b}}_i \rangle}{\|\tilde{\mathbf{b}}_i\|^2}$   $\triangleright \tilde{\mathbf{B}} = \{\tilde{\mathbf{b}}_1, \dots, \tilde{\mathbf{b}}_n\}$  is the GSO of  $\mathbf{B}$ .
5:      $\sigma_i \leftarrow \frac{\sigma}{\|\tilde{\mathbf{b}}_i\|}$ 
6:      $z_i \leftarrow D_{\mathbb{Z}, \sigma_i, d_i}$ 
7:      $\mathbf{c}_{i-1} \leftarrow \mathbf{c}_i - z_i \tilde{\mathbf{b}}_i$ 
8:      $\mathbf{x}_{i-1} \leftarrow \mathbf{x}_i + z_i \tilde{\mathbf{b}}_i$ 
9:   return  $\mathbf{v}_0$ 

```

---

We state the correctness of KleinSampler in the following Theorem 4.

**Theorem 4 ([DN12a, Theorem 1])** For positive integers  $m \geq n$ , a real  $\epsilon > 0$  and  $\sigma \geq \eta_\epsilon(\mathbb{Z}) \cdot \|\tilde{\mathbf{B}}\|$ , for any basis  $\mathbf{B} \in \mathbb{Z}^{m \times n}$  and vector  $\mathbf{c} \in \mathbb{R}^m$ , the statistical distance between the output distribution of KleinSampler( $\mathbf{B}, \mathbf{c}, \sigma$ ) and the discrete Gaussian distribution  $D_{\mathcal{L}(\mathbf{B}), \sigma, \mathbf{c}}$  is less than  $2^{-\lambda}$ .

**The Peikert Sampler.** Another widely used Gaussian sampler is the Peikert sampler, introduced in [Pei10]. It is a randomized variant of the round-off algorithm, a deterministic algorithm which offers an easy way, when given a lattice  $\Lambda$  a point  $\mathbf{c}$ , to find a close  $\mathbf{x} \in \Lambda$  close to  $\mathbf{c}$ . We describe the Peikert sampler in Algorithm 3.

---

**Algorithm 3** PeikertSampler( $\mathbf{B}, \mathbf{c}, r, \Sigma$ ) for sampling  $\mathbf{x} \leftarrow D_{\mathcal{L}(\mathbf{B}), \sqrt{\Sigma}, \mathbf{c}}$

---

```

1: function PeikertSampler( $\mathbf{B} \in \mathbb{Z}^{n \times n}, \mathbf{c} \in \mathbb{R}^n, r = \omega(\sqrt{\log n}), \Sigma > r^2 \mathbf{B} \mathbf{B}^T$ .)
2:    $\mathbf{y} \leftarrow D_1 \cdot \sqrt{\Sigma - r^2 \mathbf{B} \mathbf{B}^T}$ 
3:   return  $\mathbf{x} \leftarrow D_{\mathbb{Z}^n, r, (\mathbf{c} - \mathbf{y}) \mathbf{B}^{-1}} \mathbf{B}$ 

```

---

We state the correctness of this algorithm in the following Theorem 5.

**Theorem 5 ([Pei10, Theorem 3.1])** Let  $\Sigma_1, \Sigma_2 > \mathbf{0}$  with  $\Sigma_1 = r^2 \mathbf{B} \mathbf{B}^T$  (where  $r = \omega(\sqrt{\log n})$ ) which define  $\Sigma = \Sigma_1 + \Sigma_2 > \mathbf{0}$ . For  $\Lambda$  a lattice such that  $\sqrt{\Sigma_1} \geq \eta_\epsilon(\Lambda)$ , for  $\epsilon < \frac{1}{2}$  and for  $\mathbf{c} \in \mathbb{R}^n$ , the statistical distance between the output distribution of  $\text{PeikertSampler}(\mathbf{B}, \mathbf{c}, r, \Sigma)$  and  $D_{\Lambda, \sqrt{\Sigma}, \mathbf{c}}$  is less than  $8\epsilon$ .

Both Klein and Peikert samplers present pros and cons. While Klein's algorithm is sequential and doesn't benefit from efficiency improvement when working on structured lattices, Peikert's algorithm on the other hand is parallelizable and enjoys some speed-up when used in ring-based cryptosystems. However, the Peikert sampler generates longer vectors compared to the Klein sampler.

### 1.3.3 Leftover Hash Lemma

We give below two particular cases of the Leftover Hash Lemma ([HIL<sup>+</sup>99]), a standard result in cryptography that will appear repeatedly as an argument in security proofs. Indeed, in our cryptographic constructions, this lemma will allow us to argue that the distribution of our ciphertexts is statistically indistinguishable from the uniform distribution.

**Lemma 3 ([GPV08, Corollary 5.4])** Let  $q$  be a prime integer and  $n, m$  be positive integers such that  $m \geq 2n \log q$ . Let  $\mathbf{x} \leftarrow D_{\mathbb{Z}^m, \sigma}$  for  $\sigma \geq \omega(\sqrt{\log m})$  and  $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$ . Then, the distribution of  $\mathbf{u} = \mathbf{A} \mathbf{x} \bmod q$  is statistically close to uniform over  $\mathbb{Z}_q^n$ . Besides, when given  $\mathbf{u}$ , the conditional distribution of  $\mathbf{x}$  is  $D_{\Lambda_q^{\mathbf{u}(\mathbf{A})}, \sigma}$ .

**Lemma 4 ([ABB10a, Lemma 13])** Let  $q > 2$  be a prime integer and  $n, m, l$  be integers such that  $m \geq (n+1) \log q + \omega(\log n)$  and  $l = \text{poly}(n)$ . Let  $\mathbf{R} \leftarrow \mathcal{U}(\{-1, 1\}^{m \times l})$ ,  $\mathbf{A} \leftarrow \mathcal{U}(\mathbb{Z}_q^{n \times m})$  and  $\mathbf{B} \leftarrow \mathcal{U}(\mathbb{Z}_q^{n \times l})$ . Then, for all  $\mathbf{w} \in \mathbb{Z}_q^m$ , the distribution of  $(\mathbf{A}, \mathbf{A} \mathbf{R}, \mathbf{R}^T \mathbf{w})$  is statistically close to the distribution of  $(\mathbf{A}, \mathbf{B}, \mathbf{R}^T \mathbf{w})$ .

## 1.4 Cryptographic Primitives

In this section, we introduce some cryptographic primitives.

## 1.4.1 Basic Public-Key Primitives

### 1.4.1.1 Public-Key Encryption Scheme

**Definition.** In a public-key encryption scheme, one can use a public key  $pk$  to *encrypt* a message such that the recipient can *decrypt* it using its associated private key  $sk$ .

We give a more formal definition below.

**Definition 21** A public-key encryption scheme for a message space  $\mathcal{M}$  is a tuple  $(\text{KeyGen}, \text{Encrypt}, \text{Decrypt})$  of probabilistic polynomial-time algorithms which verify:

$\text{KeyGen}(1^n) \rightarrow (pk, sk)$ : takes as input the security parameter and outputs the public key  $pk$  and the secret key  $sk$ .

$\text{Encrypt}(pk, \mu) \rightarrow C$ : takes as input the public key  $pk$  and a message  $\mu \in \mathcal{M}$  and outputs a ciphertext  $C$  of the message  $\mu$ .

$\text{Decrypt}(sk, C) \rightarrow \{\mu, \perp\}$ : takes as input the secret key  $sk$ , a ciphertext  $C$  and outputs either the message  $\mu$  or an error  $\perp$ .

**Correctness.** The correctness of a public-key encryption scheme requires that for all  $(pk, sk) \leftarrow \text{KeyGen}(1^n)$  and  $\mu \in \mathcal{M}$ ,

$$\text{Decrypt}(sk, \text{Encrypt}(pk, \mu)) = \mu$$

with overwhelming probability.

**Security.** The most common security required for a public key encryption scheme is called *Ciphertext Indistinguishability under Chosen-Plaintext Attack* (IND-CPA), whose definition is given below.

**Definition 22** A public-key encryption scheme  $\Pi = (\text{KeyGen}, \text{Encrypt}, \text{Decrypt})$  is said to be IND-CPA secure if, for all PPT adversaries  $\mathcal{A}$ :

$$\Pr(\text{IND-CPA}_{\Pi}^{\mathcal{A}}(n) = 1) < \frac{1}{2} + \text{negl}(n),$$

where  $\text{IND-CPA}_{\Pi}^{\mathcal{A}}$  is the security game defined in Figure 1.5.

---

IND-CPA<sub>II</sub><sup>A</sup>

---

```

1 :  $b \xleftarrow{\$} \{0, 1\}$ 
2 :  $(pk, sk) \leftarrow \text{KeyGen}(1^n)$ 
3 :  $(\mu_0, \mu_1) \leftarrow \mathcal{A}(1^n, pk)$ 
4 :  $c \leftarrow \text{Encrypt}(pk, \mu_b)$ 
5 :  $b' \leftarrow \mathcal{A}(1^n, pk, c)$ 
6 : return  $b = b'$ 

```

Figure 1.5 – The IND-CPA security game

**The Dual-Regev Encryption Scheme.** We present a public-key encryption scheme, which was introduced in [GPV08] and is called the *Dual-Regev encryption* scheme. This scheme is a variant of the original Regev’s encryption scheme presented in [Reg05], in which the key generation and encryption algorithms are swapped. It represents an important tool for the construction of identity-based encryption, as we will see later. Let’s describe the three PPT algorithms composing this scheme.

- $\text{KeyGen}(1^n) \rightarrow (pk, sk)$ . The secret key  $sk$  is a vector  $\mathbf{x} \in \mathbb{Z}^m$  sampled from a discrete Gaussian distribution:  $\mathbf{x} \leftarrow D_{\mathbb{Z}^m, \sigma}$ . The public key  $pk$  is then composed of a uniformly random matrix  $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$  and of the vector  $\mathbf{u} = \mathbf{A}\mathbf{x} \in \mathbb{Z}_q^n$ .
- $\text{Encrypt}(sk, b) \rightarrow C$ . To encrypt a bit  $b \in \{0, 1\}$ , a vector  $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$  is sampled uniformly at random, and error vectors  $\mathbf{e} \leftarrow D_{\mathbb{Z}^m, \tau}$  and  $e' \leftarrow D_{\mathbb{Z}, \tau}$  are sampled from discrete Gaussian distributions of standard deviation  $\tau$ . The ciphertext is then defined as  $C = (\mathbf{b}, c) \in \mathbb{Z}_q^m \times \mathbb{Z}_q$ , where:

$$\begin{aligned} \mathbf{b} &= \mathbf{A}^T \mathbf{s} + \mathbf{e} \in \mathbb{Z}_q^m, \\ c &= \mathbf{u}^T \mathbf{s} + e' + b \cdot \lfloor q/2 \rfloor \in \mathbb{Z}_q. \end{aligned}$$

- $\text{Decrypt}(sk, C) \rightarrow \{b, \perp\}$ . To recover the message  $b$ , the recipient computes

$$res = c - \mathbf{b}^T \mathbf{x} = e' - \mathbf{e}^T \mathbf{x} + b \cdot \lfloor q/2 \rfloor \in \mathbb{Z}_q.$$

Then,  $b = 0$  if  $res$  is closer to 0 than  $\lfloor q/2 \rfloor$ , and  $b = 1$  otherwise.

The Dual-Regev encryption scheme is correct when the Euclidean norm of the error

term of  $res$  verifies  $\|e' - e^T \mathbf{x}\| \leq \lfloor q/4 \rfloor$ .

The scheme is also proven to be IND-CPA secure in [GPV08], under the hardness of  $\text{LWE}_{n,q,m,D_{\mathbb{Z}},\tau}$ .

### 1.4.1.2 Public-Key Digital Signature Scheme

**Definition.** In a public-key digital signature scheme, a signer  $S$  with public key  $pk$  can *sign* a message using its associated private key  $sk$  such that any other party who knows  $pk$  (and knows that this public key was established by  $S$ ) can check if this message has been sent by  $S$  without having been modified.

We give a more formal definition below.

**Definition 23** A public-key digital signature scheme for a message space  $\mathcal{M}$  is a tuple (KeyGen, Sign, Verify) of probabilistic polynomial-time algorithms which verify:

KeyGen( $1^n$ )  $\longrightarrow$  ( $pk, sk$ ): takes as input the security parameter and outputs the public key  $pk$  and the secret key  $sk$ .

Sign( $sk, \mu$ )  $\longrightarrow \nu$ : takes as input the secret key  $sk$  and a message  $\mu \in \mathcal{M}$  and outputs a signature  $\nu$  associated to the message  $\mu$ .

Verify( $pk, \mu, \nu$ )  $\longrightarrow \{0, 1\}$ : takes as input the public key  $pk$ , a message  $\mu$  and a signature  $\nu$  and outputs 1 (accept) if the signature is valid and 0 (reject) otherwise.

**Correctness.** The correctness of a public-key digital signature scheme requires that for all  $(pk, sk) \leftarrow \text{KeyGen}(1^n)$  and  $\mu \in \mathcal{M}$ ,

$$\text{Verify}(pk, \mu, \text{Sign}(sk, \mu)) = 1$$

with overwhelming probability.

**Security.** The most common security required for a public key digital signature scheme is called *Existential Unforgeability under an Adaptive Chosen-Message Attack* (EUF-CMA). A *forgery* is a message  $\mu$  along with a valid signature  $\nu$ , with  $\mu$  not having been previously signed by the signer  $S$ . The EUF-CMA security of a signature scheme then requires that an adversary  $\mathcal{A}$  is unable to output a forgery even after getting signatures for numerous other messages of its choice. We give a formal definition below.

**Definition 24** A public-key signature scheme  $\Pi = (\text{KeyGen}, \text{Sign}, \text{Verify})$  is said to be EUF-CMA secure if, for all PPT adversaries  $\mathcal{A}$ :

$$\Pr(\text{EUF-CMA}_{\Pi}^{\mathcal{A}} = 1)(n) < \text{negl}(n),$$

where  $\text{EUF-CMA}_{\Pi}^{\mathcal{A}}$  is the security game defined in Figure 1.6.

EUF-CMA<sub>Π</sub><sup>ℳ</sup>

- 1 :  $(pk, sk) \leftarrow \text{KeyGen}(1^n)$   
 $(\mu, \nu) \leftarrow \mathcal{A}^{\text{Sign}(sk, \cdot)}(1^n, pk)$
- 2 : Let  $\mathcal{Q}$  denote the set of queries that  $\mathcal{A}$  asked to the oracle  $\text{Sign}(sk, \cdot)$
- 3 : **return**  $\begin{cases} 1 & \text{if } \text{Verify}(pk, \mu, \nu) = 1 \text{ and } \mu \notin \mathcal{Q} \\ 0 & \text{otherwise} \end{cases}$

Figure 1.6 – The EUF-CMA security game

**The GPV Signature Scheme.** The earliest constructions for proven lattice-based signature schemes were presented in 2008. In [GPV08], Gentry, Peikert and Vaikuntanathan proposed a hash-and-sign signature scheme which they proved secure in the Random Oracle Model (ROM).

The GPV signature scheme [GPV08] was the first of a family of proven trapdoor-based signature schemes. In this scheme, the public key is a matrix  $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$  that defines the  $q$ -ary lattice  $\Lambda_q^\perp(\mathbf{A}) = \{ \mathbf{x} \in \mathbb{Z}^m \mid \mathbf{A}\mathbf{x} = \mathbf{0} \text{ mod } q \}$ . The secret key is defined by a trapdoor for  $\mathbf{A}$ , which is a short basis  $\mathbf{T} \in \mathbb{Z}^{m \times m}$  of the lattice  $\Lambda_q^\perp(\mathbf{A})$ . Then, to sign a message  $M \in \{0, 1\}^*$ , the signer first hashes it to a vector  $\mathbf{u} = \mathcal{H}(M) \in \mathbb{Z}_q^n$ , and then computes a small preimage of  $\mathbf{u}$  under the function  $f_{\mathbf{A}} : \mathbf{x} \mapsto \mathbf{A}\mathbf{x}$ .

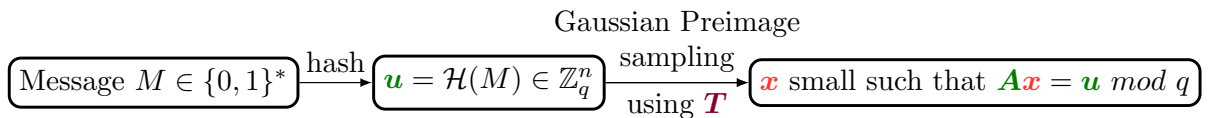


Figure 1.7 – The [GPV08] signature algorithm.

This operation, known as Gaussian preimage sampling, is made possible by knowledge



of the trapdoor: using  $\mathbf{T}$ , one can sample a vector  $\mathbf{x} \in \mathbb{Z}^m$  following a narrow discrete Gaussian distribution and which is such that  $\mathbf{Ax} = \mathbf{u} \bmod q$ . Verification simply consists in checking that  $\mathbf{Ax} = \mathcal{H}(M) \bmod q$  and that  $\mathbf{x}$  is sufficiently short. This scheme admits strong EU-CMA security in the ROM, under the hardness of the SIS problem.

There are several direct constructions of lattice-based signatures in the standard model [CHK<sup>+</sup>10; Boy10; MP12], which are often similar to identity-based encryption schemes [CHK<sup>+</sup>10; ABB10b]. In these schemes, a message  $M$  is encoded into a lattice  $\Lambda_q^\perp(\mathbf{A}_M)$ , where  $\mathbf{A}_M$  is a matrix that depends on the public key and  $M$ . Signing  $M$  then consists in sampling Gaussian preimages on  $\Lambda_q^\perp(\mathbf{A}_M)$ , similarly to [GPV08]. In [Boy10],  $\mathbf{A}_M = [\mathbf{A} \mid \mathbf{A}_0 + \sum_i M_i \mathbf{A}_i]$ , where the  $M_i$  are the bits of  $M$ , and the  $\mathbf{A}_i$  are part of the public key. This results in very large public keys. In [BFR<sup>+</sup>18],  $\mathbf{A}_M = \mathbf{A} + [\mathbf{0} \mid H(M)\mathbf{G}]$ , where  $H$  is a function with a strong injectivity property and  $\mathbf{G}$  is the structured gadget matrix of [MP12]. This yields much lighter public keys, and combines particularly well with the trapdoors from [MP12]. As far as we know, [BFR<sup>+</sup>18] provides the previously only implementation of a lattice-based standard model signature.

## 1.4.2 More advanced primitives

### 1.4.2.1 Identity-Based Encryption Scheme

The concept of *Identity Based Encryption* (IBE) was defined by Shamir in [Sha84]. The first IBE constructions were based respectively on bilinear maps and on quadratic residue assumptions. The first supposedly post-quantum IBE scheme was introduced in [GPV08] and was based on hard lattice problems. It was then followed by many improvements [CHK<sup>+</sup>10; ABB10a; DLP14; Yam16]. Note that both [DLP14] and more recently [ZMS<sup>+</sup>21] provide an implementation of an IBE scheme based on NTRU lattices.

**Definition.** Let's begin by giving a formal definition of an *Identity-Based Encryption* scheme.

**Definition 25** An IBE scheme is composed of 4 algorithms:

$\text{Setup}(1^n) \rightarrow (mpk, msk)$ : takes as input the security parameter and outputs the master public key  $mpk$  and the master secret key  $msk$ .

$\text{Extract}(1^n, mpk, msk, id) \rightarrow sk_{id}$ : takes as input the security parameter, the master keys  $mpk$  and  $msk$  and an identity  $id \in \mathcal{ID}$  and outputs a private key  $sk_{id}$  associated to the identity  $id$ .

$\text{Encrypt}(1^n, mpk, id, M) \rightarrow C$ : takes as input the security parameter, the master public key  $mpk$ , an identity  $id$  and a message  $M$  and outputs a cyphertext  $C$ .

$\text{Decrypt}(1^n, sk_{id}, C) \rightarrow \{M, Error\}$ : takes as input the security parameter, the master public key  $mpk$ , a private key  $sk_{id}$  associated to the identity  $id$  and a cyphertext  $C$  and outputs either a message  $M$  or the word "Error" if the cyphertext is invalid.

**Correctness.** The correctness of an identity-based encryption scheme requires that for all messages  $M$ , all identities  $id$ ,  $(mpk, msk) \leftarrow \text{Setup}(1^n)$  and  $sk_{id} \leftarrow \text{Extract}(1^n, mpk, msk, id)$ ,

$$\text{Decrypt}(1^n, sk_{id}, \text{Encrypt}(1^n, mpk, id, M)) = M$$

with overwhelming probability.

**Security.** The most common security required for an identity-based encryption scheme is called *Indistinguishability of Ciphertexts under a Selective-Identity Chosen-Plaintext Attack* (IND-sID-CPA). In the selective security model, the adversary  $\mathcal{A}$  has to choose its target identity  $id^*$  at the beginning of the game. In the security game, he has access to a key extraction oracle  $\text{Extract}(1^n, mpk, msk, \cdot)$  which allows him to get secret keys  $sk_{id}$  associated to identities  $id \neq id^*$ . We give a formal definition below.

**Definition 26** An identity-based encryption scheme  $\Pi = (\text{Setup}, \text{Extract}, \text{Encrypt}, \text{Decrypt})$  is said to be IND-sID-CPA secure if, for all PPT adversaries  $\mathcal{A}$ :

$$\Pr(\text{IND-sID-CPA}_{\Pi}^{\mathcal{A}} = 1)(n) < \frac{1}{2} + \text{negl}(n),$$

where  $\text{IND-sID-CPA}_{\Pi}^{\mathcal{A}}$  is the security game defined in Figure 1.8.

$\text{IND-sID-CPA}_{\Pi}^{\mathcal{A}}$
1 : $\text{id}^* \leftarrow \mathcal{A}(1^n)$
2 : $(\text{mpk}, \text{msk}) \leftarrow \text{Setup}(1^n)$
3 : $(\mu_0, \mu_1) \leftarrow \mathcal{A}^{\text{Extract}(1^n, \text{mpk}, \text{msk}, \cdot)}(1^n, \text{pk})$
4 : $b \xleftarrow{\$} \{0, 1\}$
5 : $C^* \leftarrow \text{Encrypt}(1^n, \text{mpk}, \text{id}^*, \mu_{b^*})$
6 : $b \leftarrow \mathcal{A}^{\text{Extract}(1^n, \text{mpk}, \text{msk}, \cdot)}(1^n, C^*)$
7 : <b>return</b> $b = b^*$

Figure 1.8 – The IND-sID-CPA security game

In the stronger security variant of *Indistinguishability of Ciphertexts under an Adaptive-Identity Chosen-Plaintext Attack* (IND-ID-CPA), the adversary has first access to the master public key  $\text{mpk}$  and is allowed to ask queries to the oracle  $\text{Extract}(1^n, \text{mpk}, \text{msk}, \cdot)$  before choosing the target identity  $\text{id}^*$ .

**The GPV Identity-Based Encryption Scheme.** In [GPV08], the authors also present an identity-based encryption scheme, based on their hash-and-sign signature scheme mentioned above and using the Dual-Regev encryption scheme for the encryption part of the IBE. In this IBE, the master public key  $\text{mpk}$  corresponds to the public key  $\text{pk}$  of the signature scheme and the master public key  $\text{msk}$  to the signing key  $\text{sk}$ . Then, identities  $\text{id}$  correspond to messages  $\mu$  in the signature scheme and a user private key  $\text{sk}_{\text{id}}$  corresponds to a signature  $\nu$ . Let's finally describe the resulting IBE.

- $\text{Setup}(1^n) \rightarrow (\text{mpk}, \text{msk})$ . The master public key  $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$  is a uniformly random matrix and the master secret key  $\mathbf{T}_{\mathbf{A}} \in \mathbb{Z}^{m \times m}$  is a trapdoor associated to  $\mathbf{A}$ .

- $\text{Extract}(1^n, \mathbf{A}, \mathbf{T}_A, \text{id}) \rightarrow sk_{\text{id}}$ . The private secret key  $sk_{\text{id}}$  associated to the identity  $\text{id}$  is a short preimage sample  $\mathbf{x} \in \mathbb{Z}^m$  of  $H(\text{id})$  under the function  $f_A : \mathbf{x} \mapsto \mathbf{A}\mathbf{x}$ , computed thanks to  $\mathbf{T}_A$ .
- $\text{Encrypt}(1^n, \mathbf{A}, \text{id}, M) \rightarrow C$ . A ciphertext  $C$  is computed using  $\text{Encrypt}((\mathbf{A}, H(\text{id})), M)$  from the Dual-Regev encryption scheme.
- $\text{Decrypt}(1^n, sk_{\text{id}}, C) \rightarrow \{M, \text{Error}\}$ . Similarly,  $\text{Decrypt}(sk_{\text{id}}, C)$  from the Dual-Regev encryption scheme is used to recover the message  $M$ .

This IBE is proven to be IND-ID-CPA secure in the random oracle model ([GPV08, Theorem 7.2]).

#### 1.4.2.2 Encoding Messages with Full-Rank Differences

We begin by describing the notion of encoding with Full-Rank Differences (FRD). The original definition comes from [ABB10b] and Definition 27 extends it by requiring  $H(m)$  to be invertible.

**Definition 27 (adapted from [ABB10b])** An encoding with full-rank differences from the set  $\mathcal{M}$  to a ring  $\mathcal{R}$  is a map  $H : \mathcal{M} \rightarrow \mathcal{R}$  such that:

- for any  $m \in \mathcal{M}$ ,  $H(m)$  is invertible,
- for any  $m_1, m_2 \in \mathcal{M}$  such that  $m_1 \neq m_2$ ,  $H(m_1) - H(m_2)$  is invertible,
- $H$  is computable in polynomial time.

Before constructing an FRD encoding in the module setting (that is, taking values in  $\mathcal{R}_q^{d \times d}$ ), we first construct one in the ring setting (taking values in  $\mathcal{R}_q$ ). Our construction is based on the following result from [LS18], which allows us to find invertible elements in  $\mathcal{R}_q$ .

**Theorem 6 ([LS18, Corollary 1.2])** Let  $n \geq r > 1$  be powers of 2, and  $q$  a prime such that  $q \equiv 2r + 1 \pmod{4r}$ . Then the cyclotomic polynomial  $X^n + 1$  factors in  $\mathbb{Z}_q[X]$  as  $X^n + 1 = \prod_{i=1}^r (X^{n/r} - s_i)$ , for some distinct  $s_i \in \mathbb{Z}_q^*$  such that the  $(X^{n/r} - s_i)$  are irreducible in  $\mathbb{Z}_q[X]$ . Moreover, any  $f \in \mathcal{R}_q$  such that  $0 < \|f\|_\infty < q^{1/r}/\sqrt{r}$  or  $0 < \|f\| < q^{1/r}$  is invertible.

This result can be used to build two different types of FRD encodings. The first one is a "low-degree" FRD, described in Proposition 2.

**Proposition 2** Let  $n \geq r > 1$  be powers of 2,  $q$  a prime such that  $q \equiv 2r + 1 \pmod{4r}$ , and  $\mathcal{M} = \mathbb{Z}_q^{n/r} \setminus \{\mathbf{0}\}$  the set of messages. Then the following map  $H : \mathcal{M} \rightarrow \mathcal{R}_q$  is an FRD encoding.

$$H : \mathcal{M} \rightarrow \mathcal{R}_q$$

$$(m_0, \dots, m_{n/r-1}) \mapsto \sum_{i=0}^{n/r-1} m_i X^i$$

*Proof.* According to Theorem 6 on the factorization of  $X^n + 1$  and the Chinese remainder theorem, we have that  $\mathcal{R}_q = \frac{\mathbb{Z}_q[X]}{\langle X^n + 1 \rangle} \simeq \prod_{i=1}^r \frac{\mathbb{Z}_q[X]}{\langle X^{n/r} - s_i \rangle}$ , with the  $X^{n/r} - s_i$  being irreducible, meaning that the  $r$  rings of the product are actually fields (isomorphic to  $\mathbb{F}_{q^{n/r}}$ ).

Under the canonical map from the Chinese Remainder Theorem (CRT), a nonzero polynomial  $f$  of degree less than  $n/r$  is sent to  $(f \bmod X^{n/r} - s_1, \dots, f \bmod X^{n/r} - s_r) = (f, \dots, f)$ . Each coordinate of the image vector is then invertible (since nonzero) in  $\mathbb{F}_{q^{n/r}}$ , which makes  $f$  itself invertible in  $\mathcal{R}_q$ . This proves that all the  $H(m)$  are invertible.

For distinct  $m_1, m_2 \in \mathcal{M}$ ,  $H(m_1) - H(m_2)$  is a nonzero polynomial of degree less than  $n/r$ , so it is invertible in  $\mathcal{R}_q$  by the previous reasoning. Finally,  $H$  is clearly computable in polynomial time.  $\square$

But one could also encode messages as polynomials of infinity norm smaller than  $\frac{q^{1/r}}{2\sqrt{r}}$  with an injective map and thus obtain this way a "small-norm" FRD, rather than the previous "low-degree" one. We present such an FRD encoding in the following Proposition 3.

**Proposition 3** Let  $n \geq r > 1$  be powers of 2,  $q$  a prime such that  $q \equiv 2r + 1 \pmod{4r}$  and  $1 \leq D \leq \frac{q^{1/r}}{2\sqrt{r}}$  an integer. We define  $\mathcal{M} = \{-D, \dots, D\}^n \setminus \{(0, \dots, 0)\}$  the set of identities. Then the following map  $H : \mathcal{M} \rightarrow \mathcal{R}_q$ , such that  $H_M(m_0, \dots, m_{n-1}) = \sum_{i=0}^{n-1} m_i X^i$  is an FRD encoding.

$$H : \mathcal{M} \rightarrow \mathcal{R}_q$$

$$(m_0, \dots, m_{n-1}) \mapsto \sum_{i=0}^{n-1} m_i X^i$$

*Proof.* We have  $\|H_M(m)\|_\infty \leq D < \frac{q^{1/r}}{2\sqrt{r}}$  for all  $m$  because of the choice of  $\mathcal{M}$ . So according to Lemma 6,  $H_M(m)$  is invertible. For all  $m_1, m_2 \in \mathcal{M}$ , we also have

$\|H_M(m_1) - H_M(m_2)\|_\infty \leq 2D < \frac{q^{1/r}}{\sqrt{r}}$  so  $H_M(m_1) - H_M(m_2)$  is invertible. Finally,  $H_M$  is an FRD encoding.  $\square$   $\square$

Finally, we can build an FRD encoding in the module setting using an existing FRD encoding in the ring setting  $H_R : \mathcal{M} \rightarrow \mathcal{R}_q$  by constructing:

$$H_M : \mathcal{M} \rightarrow \mathcal{R}_q^{d \times d}$$

$$m \mapsto H_R(m) \cdot \mathbf{I}_d = \begin{bmatrix} H_R(m) & & \\ & \ddots & \\ & & H_R(m) \end{bmatrix},$$

where  $\mathbf{I}_d \in \mathcal{R}_q^{d \times d}$  is the identity matrix.

**Lemma 5** If  $H_R$  is an FRD (in the ring setting) from  $\mathcal{M}$  to  $\mathcal{R}_q$ , then  $H_M$  as constructed above is an FRD (in the module setting) from  $\mathcal{M}$  to  $\mathcal{R}_q^{d \times d}$ .



# LATTICE TRAPDOORS ON MODULES: IMPLEMENTATION AND APPLICATIONS

---

This chapter’s content is mainly based on a joint work [BEP<sup>+</sup>21] with Pauline Bert, Gautier Eberhart, Adeline Roux-Langlois, and Mohamed Sabt. The corresponding implementation is available at [https://github.com/lucasprabel/module\\_gaussian\\_lattice](https://github.com/lucasprabel/module_gaussian_lattice).

## Contents

---

<b>2.1. Introduction</b> .....	72
2.1.1 Presentation .....	72
2.1.2 Contributions: A Technical Overview .....	73
2.1.2.1 Preimage Sampling .....	73
2.1.2.2 Applications .....	75
2.1.2.3 Comparison with Previous Works .....	75
<b>2.2. Gaussian Preimage Sampling on Module Lattices</b> .....	76
2.2.1 Module $\mathbf{G}$ -Lattice .....	77
2.2.2 $\mathbf{G}$ -Sampling .....	79
2.2.2.1 How to Perform $\mathbf{G}$ -Sampling .....	79
2.2.2.2 What to Do with $\mathbf{G}$ -Samples .....	80
2.2.3 Perturbation Sampling .....	80
2.2.3.1 Schur Complements .....	80
2.2.3.2 An Efficient Algorithm for Sampling Perturbations .....	82
2.2.3.3 Correctness .....	82
2.2.3.4 Gaussian Parameters for Perturbation Sampling .....	85
2.2.4 Sampling Gaussian Preimages .....	86
2.2.5 Implementation and Performance .....	86
2.2.5.1 Implementation Choices .....	86



2.2.5.2 Timings.....	87
<b>2.3. A GPV Signature Scheme on Modules .....</b>	<b>88</b>
2.3.1 The Scheme.....	88
2.3.2 Parameters and Security .....	90
2.3.3 Implementation and Performance.....	91
2.3.3.1 Comparison Between Rings and Modules.....	92
2.3.3.2 Comparison with [GPR <sup>+</sup> 19].....	92
<b>2.4. A Standard Model Signature Scheme on Modules .....</b>	<b>93</b>
2.4.1 The Scheme.....	93
2.4.2 Parameters and Security .....	94
2.4.3 Implementation and Performance.....	95
<b>2.5. An Identity-Based Encryption Scheme on Modules.....</b>	<b>96</b>
2.5.1 Implementation and Performance.....	98

---

## 2.1 Introduction

### 2.1.1 Presentation

As explained in Section 1.2.3.3, ideal lattices [PR06; SST<sup>+</sup>09; LPR10], usually based on rings of the form  $\mathcal{R}_q = \mathbb{Z}_q[X]/\langle X^n + 1 \rangle$ , are often the first choice for efficient lattice-based constructions, preferred to unstructured lattices. On the other hand, module lattices [LS15], based on modules of the form  $\mathcal{R}_q^d$ , lie somewhere between ideal lattices and unstructured ones. Constructions in the module setting are (almost) as efficient as ring-based ones, and have other advantages for practical schemes.

Typically, module schemes fix a modulus  $q$  and a degree  $n$  for all parameter sets, and the security parameter is the rank  $d$  of the module. This leads to a more flexible choice of parameters, and potentially easier optimization (since one only has to optimize arithmetic in the base ring  $\mathcal{R}_q$  to obtain a faster arithmetic for all parameter sets). Additionally, fundamental problems on module lattices might not suffer from the same structural weaknesses as on ideal lattices (see [PHS19]). As an example of the interest of module lattices, several NIST candidates of the 3rd round of the post-quantum cryptography standardization process relied on them [DKL<sup>+</sup>18; DKR<sup>+</sup>18], and a recent result [CPS<sup>+</sup>19] proposed a module variant of the Falcon signature scheme [FHK<sup>+</sup>17].

Thus, module lattices seem to constitute an interesting subject to meet the needs of efficiency and parameters' flexibility of recent constructions. Therefore, developing tools

to build efficient lattice-based schemes constitutes an important objective.

In this perspective, this chapter presents the development and the implementations of efficient Gaussian preimage sampling techniques on module lattices, which rely on the works of Micciancio and Peikert in 2012 [MP12], and Micciancio and Genise in 2018 [GM18]. Our implementation has a major benefit in terms of modularity, which makes it practical to use for signature schemes, but also for more advanced constructions that rely on trapdoors such as identity-based encryption. Specifically, it is easy to use in the ring or module setting, and easy to modify the arithmetic on  $\mathcal{R}_q$  (as different schemes have different conditions on  $q$ ).

Relying on these tools, we also present two instantiations and implementations of proven trapdoor-based signature schemes in the module setting: GPV in the random oracle model and a variant of it in the standard model presented in [BFR<sup>+</sup>18]. Concerning this last scheme, we address a security issue and correct obsolescence problems in the implementation of [BFR<sup>+</sup>18] by building ours from scratch. To the best of our knowledge, this is the first efficient implementation of a lattice-based signature scheme in the standard model. Relying on that last signature, we also present the implementation of a standard model IBE in the module setting. We show that while the resulting schemes may not be competitive with the most efficient NIST candidates, they are practical and run on a standard laptop in acceptable time, which paves the way for practical advanced trapdoor-based constructions.

## 2.1.2 Contributions: A Technical Overview

Our main contribution is the development and the implementation of efficient Gaussian preimage sampling techniques on module lattices together with an implementation that offers the advantages of being constant-time and modular, thereby making it practical for both signature schemes and more complex constructions that make use of trapdoors. Our resulting C implementation is public and open-source, available at [https://github.com/lucasprabel/module\\_gaussian\\_lattice](https://github.com/lucasprabel/module_gaussian_lattice).

### 2.1.2.1 Preimage Sampling

Gaussian preimage sampling is a crucial operation in trapdoor-based schemes, but no method adapted to module lattices existed previously. We develop efficient algorithms for trapdoor generation and Gaussian preimage sampling in the module setting, by general-

izing existing tools in the unstructured and ring settings [MP12; GM18]. This adaptation has to be done carefully to correctly work over modules. In particular, the perturbation sampling step does not directly adapt, and we resort to our own algorithm, using some subroutines from [GM18]. We also provide a detailed description of those algorithms and of the conditions needed to choose their parameters. Our contributions can serve as an important building block for advanced trapdoor-based constructions, such as identity-based encryption, attribute-based encryption, or group signature.

Our implementation requires no external dependencies, and is easily customizable as needed. Its modularity is a key feature, achieved through the use of several interchangeable basic building blocks as illustrated in Figure 2.1: the arithmetic over  $\mathbb{Z}_q$  and  $\mathcal{R}_q$ , a pseudorandom number generator, and a (constant-time) sampler of discrete Gaussian distributions over  $\mathbb{Z}$ . Additionally, our two signature schemes require different structures for the ring  $\mathcal{R}_q$ , making use of distinct arithmetic operations. These modifications are easily achieved given the modularity of our implementation.

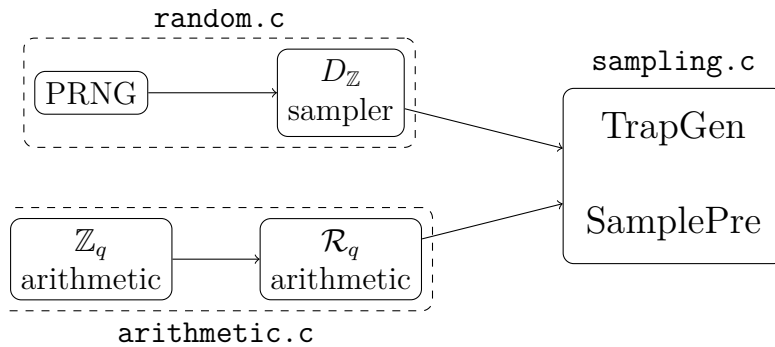


Figure 2.1 – Basic structure of our implementation and relationships between the blocks.

Table 2.1 – Overview of the performances of our trapdoor tools and cost of sampling scalar Gaussians for degree  $n = 256$  and rank  $d = 4$ .

Phase	TrapGen	SamplePre		
		Perturb. sampling	$\mathbf{G}$ -sampling	Global
Running time	36.56 ms	5.48 ms	8.28 ms	14.87 ms
Sampling $D_{\mathbb{Z}}$	74%	60%	84%	70%

In Table 2.1 we give an overview of the running times of our trapdoor algorithms on an Intel i7-8650U CPU running at 1.90 GHz. More specifically, we emphasize the proportion of time taken to sample Gaussians over  $\mathbb{Z}$  and observe that having a fast and effective sampler is crucial, as it constitutes the largest part of the computational costs.

### 2.1.2.2 Applications

As an application, we propose an implementation of two trapdoor-based signature schemes and of an identity-based encryption scheme. The GPV signature is the simplest trapdoor-based scheme one can think of, since key generation is exactly the trapdoor generation algorithm, and signing essentially consists in Gaussian preimage sampling. As such, it makes for a natural way of evaluating trapdoor tools and techniques. Our second signature scheme, proven secure in the standard model, is a variation on GPV, and has been constructed by adapting the scheme from [BFR<sup>+</sup>18] to the module setting. The original construction was using an encoding function that should satisfy a strong injectivity property but it was not the case in practice. To avoid this issue, we propose a construction for this encoding using a result of [LS18], which allows us to find invertible elements in  $R_q$ , and which needs a specific  $q$  as a consequence. Relying on this signature scheme, we also implement the standard model IBE scheme from [BFR<sup>+</sup>18], which was inspired by the IBE [ABB10a], in the module setting.

Our GPV implementation relies on our trapdoor tools, as well as a Number Theoretic Transform for fast multiplication in  $\mathcal{R}_q$ , adapted from CRYSTALS-Kyber [DKL<sup>+</sup>18]. In our standard model schemes, the particular structure of the ring, due to the particular choice of  $q$ , is incompatible with the NTT. As such, the main difference with GPV in terms of implementation is the use of a partial NTT inspired by [LS18], instead of a full one. An example of the performances of our signatures is given in Table 2.2. For this set of parameters, the public key has size 508kB, the private key 5.06MB and the signature 131kB.

Table 2.2 – Performances of our signatures implementation and comparison with previous GPV implementation (96-bit security parameter sets, lattice dimension 1024, modulus  $q \approx 2^{30}$ ).

Scheme	KeyGen	Sign	Verify
GPV ([GPR <sup>+</sup> 19])	5.86 ms	32.42 ms	0.28 ms
GPV (this work)	8.94 ms	13.08 ms	0.29 ms
BFRS (this work)	9.46 ms	15.66 ms	1.19 ms

### 2.1.2.3 Comparison with Previous Works

From a theoretical point of view, the adaptation of the algorithms from [MP12; GM18] to the module setting is quite direct but has to be done carefully, in particular concerning

the perturbation sampling algorithm which is an important step in those algorithms. This algorithm over rings is iteratively sampling vectors with a covariance matrix of dimension  $2 \times 2$  over  $\mathcal{R}$ , whereas in our case, the matrix has size  $2d \times 2d$ , where  $d$  is the module rank. As a consequence, we have to decompose the covariance matrix into blocks of different sizes at each iteration instead of simply updating ring elements.

We chose to only compare our GPV implementation with the recent work of Gür *et al.* [GPR<sup>+</sup>19], as it already outperforms previous implementations of Gaussian preimage sampling [BB13; GPR<sup>+</sup>18]

We provide a new encoding function for the signature and the IBE schemes (presented in 1.4.2.2), which allows correcting a security issue in the corresponding schemes in [BFR<sup>+</sup>18]. Our implementation does not rely on the [BFR<sup>+</sup>18] one and does not use the NTL library. We do not compare the original implementation of the BFRS scheme [BFR<sup>+</sup>18] with our corrected version, as the former’s limited security would make the comparison irrelevant.

We also present a public and open-source implementation of a standard model IBE scheme in the module setting, relying on our standard model signature scheme.

## 2.2 Gaussian Preimage Sampling on Module Lattices

Our implementation builds upon the notion of trapdoors introduced in [MP12], which is a crucial ingredient for efficient trapdoor-based schemes such as the two signatures and the IBE we implemented. Compared to the short bases of [GPV08], these trapdoors are more compact and enjoy faster algorithms, both asymptotically and in practice. They were later generalized to ideal lattices in [LCC14], and an efficient instantiation of the associated algorithms was given in [GM18]. However, to the best of our knowledge, neither the trapdoors nor their algorithms had been adapted to the module setting until our work.

In this section, we generalize in detail these constructions to module lattices, following the ideas from [MP12], by accomplishing two goals:

- We derive an algorithm TrapGen from [MP12, Section 5.2]. It generates a uniform matrix  $\mathbf{A} \in \mathcal{R}_q^{d \times m}$  along with its trapdoor  $\mathbf{T} \in \mathcal{R}^{2d \times dk}$ , where  $k = \lceil \log_b q \rceil$  and  $m = d(k+2)$ . The trapdoor  $\mathbf{T}$  is sampled from a Gaussian distribution of parameter  $\sigma$ . The matrix  $\mathbf{A}$  defines hard module SIS and ISIS problems.
- We give an algorithm SamplePre, that uses  $\mathbf{T} \in \mathcal{R}^{2d \times dk}$  to perform efficient Gaussian preimage sampling with parameter  $\zeta$ , effectively solving the module SIS and ISIS

problems.

Gaussian preimage sampling consists in sampling from a spherical discrete Gaussian distribution on cosets of the lattice  $\Lambda_q^\perp(\mathbf{A})$  (that is, the sets  $\Lambda_q^{\mathbf{u}}(\mathbf{A})$  for  $\mathbf{u} \in \mathcal{R}^d$ ) using  $\mathbf{T}$ . The standard deviation  $\zeta$  of this distribution should be small (so that it is hard to sample from it without knowing  $\mathbf{T}$ ), and the produced vectors should not leak any information about  $\mathbf{T}$ . To this end, we follow the method introduced in [MP12] where sampling from  $D_{\Lambda_q^{\mathbf{u}}(\mathbf{A}), \zeta}$  is divided into two complementary phases:

- *$\mathbf{G}$ -sampling* of parameter  $\alpha$ , which ensures that our samples actually lie in the good coset.
- *Perturbation sampling* with parameters  $\zeta$  and  $\alpha$ , which conceals the information about  $\mathbf{T}$  in the output distribution.

### 2.2.1 Module $\mathbf{G}$ -Lattice

We recall that we work with the rings  $\mathcal{R} = \mathbb{Z}[X]/\langle X^n + 1 \rangle$  and  $\mathcal{R}_q = \mathbb{Z}_q[X]/\langle X^n + 1 \rangle$ , where  $n$  is a power of two and  $q$  a prime modulus.

We first define a specific lattice named the module  $\mathbf{G}$ -lattice, where the **Module-SIS** problem is easy. Then, we describe how we can generate simultaneously a random-looking lattice and its trapdoor, which is a way of mapping the module  $\mathbf{G}$ -lattice to it. Finally, we show how to use this trapdoor to solve **Module-SIS** on the random lattice.

In the case of ideal lattices, the construction starts with the ring *gadget vector*

$$\mathbf{g}^T = [1 \quad b \quad b^2 \quad \dots \quad b^{k-1}] \in \mathcal{R}_q^{1 \times k}, \quad \text{where } k = \lceil \log_b q \rceil,$$

whose entries are constant power-of- $b$  polynomials, and the ring  $\mathbf{G}$ -lattice  $\Lambda_q^\perp(\mathbf{g}^T) \subset \mathcal{R}^k$ . Typically, we take  $b = 2$  (which allows us to use efficient bitwise operations in our implementation) but choosing larger values for  $b$  is possible in our implementation and could bring some interesting trade-offs in terms of efficiency. In the module setting, we use these to create the matrix

$$\mathbf{G} = \mathbf{I}_d \otimes \mathbf{g}^T = \begin{bmatrix} \mathbf{g}^T & & & \\ & \mathbf{g}^T & & \\ & & \ddots & \\ & & & \mathbf{g}^T \end{bmatrix} \in \mathcal{R}^{d \times dk},$$

and the associated module  $\mathbf{G}$ -lattice  $\Lambda_q^\perp(\mathbf{G}) \subset \mathcal{R}^{dk}$ , which is isomorphic to  $d$  copies of  $\Lambda_q^\perp(\mathbf{g}^T)$ . Having introduced the  $\mathbf{G}$ -lattice, we can now define trapdoors.

**Definition 28** A trapdoor for a matrix  $\mathbf{A} \in \mathcal{R}_q^{d \times m}$  is a matrix  $\mathbf{T} \in \mathcal{R}^{(m-dk) \times dk}$  such that

$$\mathbf{A} \begin{bmatrix} \mathbf{T} \\ \mathbf{I}_{dk} \end{bmatrix} = \mathbf{H}\mathbf{G}$$

for some invertible  $\mathbf{H} \in \mathcal{R}_q^{d \times d}$ , called the *tag* of  $\mathbf{A}$ .

We now describe the algorithm TrapGen, which outputs a matrix  $\mathbf{A}$  along with its associated trapdoor  $\mathbf{T}$ , given a tag  $\mathbf{H}$ . So that  $\mathbf{A}$ 's uniformity is guaranteed by a Module-LWE instance, we instantiate TrapGen with  $m = d(k + 2)$ .

---

**Algorithm 4** TrapGen( $\mathbf{H}, \sigma$ ) for the generation of a matrix  $\mathbf{A}$  and its trapdoor  $\mathbf{T}$

---

1: <b>function</b> TrapGen( $\mathbf{H} \in \mathcal{R}_q^{d \times d}, \sigma > 0$ )	
2: $\hat{\mathbf{A}} \leftarrow \mathcal{U}(\mathcal{R}_q^{d \times d})$	▷ $\hat{\mathbf{A}} \in \mathcal{R}_q^{d \times d}$
3: $\mathbf{A}' \leftarrow [\mathbf{I}_d \mid \hat{\mathbf{A}}]$	▷ $\mathbf{A}' \in \mathcal{R}_q^{d \times 2d}$
4: $\mathbf{T} \leftarrow D_{\mathcal{R}^{2d \times dk}, \sigma}$	▷ $\mathbf{T} \in \mathcal{R}^{2d \times dk}$
5: $\mathbf{A} \leftarrow [\mathbf{A}' \mid \mathbf{H}\mathbf{G} - \mathbf{A}'\mathbf{T}]$	▷ $\mathbf{A} \in \mathcal{R}^{d \times m}$
6: <b>return</b> ( $\mathbf{A}, \mathbf{T}$ )	

---

We state the correctness of the trapdoor generation algorithm in the following Lemma 6.

**Lemma 6** Let  $q, d$  be positive integers,  $k = \lceil \log_b q \rceil$ ,  $m = d(k + 2)$ ,  $\mathbf{H} \in \mathcal{R}_q^{d \times d}$ ,  $\sigma > 0$ , and  $(\mathbf{A}, \mathbf{T}) \leftarrow \text{TrapGen}(\mathbf{H}, \sigma)$ . Then, the following points hold:

- $\mathbf{T}$  is a trapdoor for  $\mathbf{A}$  with tag  $\mathbf{H}$ ;
- $\mathbf{A}$  is computationally indistinguishable from uniform (excluding the identity submatrix in the first  $d$  columns) based on the hardness of the decision version of  $\text{Module-LWE}_{n,d,q,\sigma}$ .

*Proof.* Following the structure of the matrix  $\mathbf{A}$  returned by the algorithm TrapGen, the equality  $\mathbf{A} \begin{bmatrix} \mathbf{T} \\ \mathbf{I}_{dk} \end{bmatrix} = \mathbf{H}\mathbf{G}$  is verified and thus  $\mathbf{T}$  is a trapdoor for  $\mathbf{A}$ .

Moreover, writing  $\mathbf{T} = \begin{bmatrix} \mathbf{T}_1 \\ \mathbf{T}_2 \end{bmatrix}$  we have  $\mathbf{A}'\mathbf{T} = \mathbf{T}_1 + \hat{\mathbf{A}}\mathbf{T}_2$ . Therefore,  $[\mathbf{A}' \mid -\mathbf{A}'\mathbf{T}]$  is an instance (ignoring the identity submatrix in the first  $d$  columns) of the decision version of  $\text{Module-LWE}_{n,d,q,\sigma}$  in its normal form (the secret and the error following the same distribution).  $\square$

These trapdoors are useful as they allow one to compute small vectors on any coset of a random lattice if they can do so on the  $\mathbf{G}$ -lattice. Indeed, a small vector on the  $\mathbf{G}$ -lattice can be mapped to a small vector on the random lattice using its trapdoor.

Now that we know how to generate trapdoors, we need to understand how to use them to perform Gaussian preimage sampling. Gaussian preimage sampling consists in sampling from a spherical discrete Gaussian distribution on cosets of the lattice  $\Lambda_q^\perp(\mathbf{A})$  (that is, the sets  $\Lambda_q^\mathbf{u}(\mathbf{A})$  for  $\mathbf{u} \in \mathcal{R}^d$ ) using the trapdoor  $\mathbf{T}$ . The standard deviation  $\zeta$  of this distribution should be as small as possible (so that it is hard to sample from it without the knowledge of  $\mathbf{T}$ ), and the produced vectors should not leak any information about  $\mathbf{T}$ . To this end, we follow the method introduced in [MP12] where sampling from  $D_{\Lambda_q^\perp(\mathbf{A}),\zeta}$  is divided into two complementary phases: the  *$\mathbf{G}$ -sampling*, which ensures that our samples actually lie in the good coset, and the *perturbation sampling*, which conceals the information about  $\mathbf{T}$  in the output distribution.

## 2.2.2 $\mathbf{G}$ -Sampling

We now describe one of the two main steps of the Gaussian preimage sampling. Given a target vector  $\mathbf{v} \in \mathcal{R}^d$ , the goal of  $\mathbf{G}$ -sampling is to compute a vector  $\mathbf{z}$  following a discrete Gaussian of parameter  $\alpha$  over the coset  $\Lambda_q^\mathbf{v}(\mathbf{G})$  of the module  $\mathbf{G}$ -lattice defined by  $\mathbf{G} \in \mathcal{R}^{d \times dk}$ .

### 2.2.2.1 How to Perform $\mathbf{G}$ -Sampling

We make use of an efficient algorithm [GM18, Section 3] designed for  $\mathbf{G}$ -sampling in the unstructured setting, running in  $\mathcal{O}(k)$  time. It samples on cosets of the scalar  $\mathbf{G}$ -lattice  $\Lambda_q^\perp(\mathbf{g}_s^T) \subset \mathbb{Z}^k$ , defined by the scalar gadget vector  $\mathbf{g}_s^T = [1 \ b \ b^2 \ \dots \ b^{k-1}] \in \mathbb{Z}_q^{1 \times k}$  used in the construction of trapdoors on unstructured lattices. While we do not go into detail about the algorithm, and treat it as a black box, we need to take into account the following constraint for its instantiation. According to [GM18, Corollary 3.1 and Proposition 3.1],



we need to take  $\alpha \geq \sqrt{2b} \cdot (2b + 1) \cdot \sqrt{\log(2k(1 + 1/\varepsilon))/\pi}$  and  $\alpha \geq (b + 1)\eta_\varepsilon(\mathbb{Z})$  for the algorithm to be correct.

To sample a vector  $\mathbf{z}_r \in \mathcal{R}^k$  from a coset of the ring  $\mathbf{G}$ -lattice, one can do the following: independently sample  $n$  column vectors from the scalar  $\mathbf{G}$ -lattice, arrange them into an  $k \times n$  matrix, and form each of the  $k$  entries of  $\mathbf{z}_r$  by reading the rows in order. In turn, module  $\mathbf{G}$ -sampling consists of  $d$  independent operations of ring  $\mathbf{G}$ -sampling. In summary, we make  $nd$  calls to the scalar  $\mathbf{G}$ -sampling algorithm, leading to a procedure for module  $\mathbf{G}$ -sampling running in optimal  $\mathcal{O}(ndk)$  time.

### 2.2.2.2 What to Do with $\mathbf{G}$ -Samples

Now that we are able to sample such vectors, let us explain how to use them, along with the trapdoor  $\mathbf{T}$ , to compute a Gaussian vector  $\mathbf{x} \in \Lambda_q^u(\mathbf{A})$  for a given  $\mathbf{u} \in \mathcal{R}^d$ . Letting  $\mathbf{v} = \mathbf{H}^{-1}\mathbf{u}$ , we sample  $\mathbf{z} \leftarrow D_{\Lambda_q^v(\mathbf{G}), \alpha}$ , and output  $\mathbf{x} = \begin{bmatrix} \mathbf{T} \\ \mathbf{I} \end{bmatrix} \mathbf{z}$ . One can check that such an  $\mathbf{x}$  then lies in the desired coset. Nevertheless, its distribution is degenerate (the support is not full-rank) and skewed (the covariance matrix is  $\alpha^2 \begin{bmatrix} \mathbf{T} \\ \mathbf{I} \end{bmatrix} [\mathbf{T}^T \ \mathbf{I}]$ ), which leaks information about the trapdoor. To solve this problem, we make use of perturbation vectors, as first described in [Pei10].

## 2.2.3 Perturbation Sampling

Perturbation sampling aims at sampling vectors following the Gaussian distribution over  $\mathcal{R}^m$  of covariance  $\Sigma_p = \zeta^2 \mathbf{I} - \alpha^2 \begin{bmatrix} \mathbf{T} \\ \mathbf{I} \end{bmatrix} [\mathbf{T}^T \ \mathbf{I}]$ . In a way, this covariance matrix is complementary to the one of  $\begin{bmatrix} \mathbf{T} \\ \mathbf{I} \end{bmatrix} \mathbf{z}$ , where  $\mathbf{z}$  is the output of the  $\mathbf{G}$ -sampling. This is so that when we sum the perturbation  $\mathbf{p}$  and  $\begin{bmatrix} \mathbf{T} \\ \mathbf{I} \end{bmatrix} \mathbf{z}$ , the final covariance matrix  $\Sigma_p + \alpha^2 \begin{bmatrix} \mathbf{T} \\ \mathbf{I} \end{bmatrix} [\mathbf{T}^T \ \mathbf{I}] = \zeta^2 \mathbf{I}$  does not leak any information about the trapdoor  $\mathbf{T}$ .

### 2.2.3.1 Schur Complements

Internally, perturbation sampling takes place in the ring  $\mathcal{P} = \mathbb{R}[X]/\langle X^n + 1 \rangle$  rather than the usual ring  $\mathcal{R}$ . As in most discrete Gaussian sampling algorithms, computations are done with real numbers even if the final result is composed of integers only. Since  $\mathcal{R}$  can naturally be embedded in  $\mathcal{P}$ , we can consider  $\mathbf{T}$  and covariance matrices to have entries in  $\mathcal{P}$ .

We denote by  $\phi$  the ring homomorphism from  $\mathcal{P} = \mathbb{R}[X]/\langle X^n + 1 \rangle$  to  $\mathbb{R}^{n \times n}$  that maps

a polynomial  $a = \sum_{i=0}^{n-1} a_i X^i$  to the matrix

$$\phi(a) = \begin{bmatrix} a_0 & -a_{n-1} & \cdots & -a_1 \\ a_1 & a_0 & & \vdots \\ \vdots & & \ddots & -a_{n-1} \\ a_{n-1} & \cdots & a_1 & a_0 \end{bmatrix},$$

which represents multiplication by  $a$  in  $\mathcal{P}$  when polynomials are represented as vectors over  $\mathbb{R}$ . This definition is naturally extended to matrices over  $\mathcal{P}$  by component-wise application.

Sampling a vector of  $m$  polynomials with covariance  $\Sigma \in \mathcal{P}^{m \times m}$  is then equivalent to sampling a vector of  $nm$  scalars with covariance  $\phi(\Sigma) \in \mathbb{R}^{nm \times nm}$ . While our algorithm is described in terms of matrices over  $\mathcal{P}$ , it is easier to understand it when viewing the covariance matrix as a real matrix.

As a reminder, for a symmetric matrix  $\mathbf{M} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B}^T & \mathbf{D} \end{bmatrix}$  over  $\mathbb{R}$  where the lower-right block  $\mathbf{D}$  is invertible, the Schur complement of  $\mathbf{D}$  is defined as  $\mathbf{M}/\mathbf{D} := \mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{B}^T$ . Then,  $\mathbf{M}$  is positive definite if and only if both  $\mathbf{D}$  and  $\mathbf{M}/\mathbf{D}$  are positive definite themselves. If we write out the blocks of a structured scalar covariance matrix

$$\phi(\Sigma) = \begin{bmatrix} \phi(\mathbf{A}) & \phi(\mathbf{B}) \\ \phi(\mathbf{B})^T & \phi(\mathbf{D}) \end{bmatrix} \in \mathbb{R}^{nm \times nm},$$

then we can consider  $\phi(\Sigma)/\phi(\mathbf{D}) = \phi(\mathbf{A}) - \phi(\mathbf{B})\phi(\mathbf{D}^{-1})\phi(\mathbf{B})^T$ . For a similar definition to make sense in terms of matrices over  $\mathcal{P}$  (that is, we want to define the Schur complement  $\Sigma/\mathbf{D} = \mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{B}^T$ ), we first need to construct a matrix  $\mathbf{B}^T$  such that  $\phi(\mathbf{B}^T) = \phi(\mathbf{B})^T$ .

Since our perturbation sampling algorithm deals with Schur complements, and that computing a Schur complement in a symmetric matrix involves matrix transposition, we need to take a closer look at transposing structured matrices. More precisely, let  $\phi(\mathbf{B})$  be a structured matrix over  $\mathbb{R}$ . Because of its structure, it can be described by some polynomials  $b_{ij} \in \mathcal{P}$ . We want to find out the polynomials with which  $\phi(\mathbf{B})^T$  can be described.

For any polynomial  $a = \sum_{i=0}^{n-1} a_i X^i \in \mathcal{P}$ , we define its *transpose* to be  $a^T = a_0 - \sum_{i=1}^{n-1} a_{n-i} X^i \in \mathcal{P}$ . It is the polynomial such that  $\phi(a^T) = \phi(a)^T$ . The transpose of a matrix  $\mathbf{B}$  over  $\mathcal{P}$  is the matrix  $\mathbf{B}^T$  such that  $\mathbf{B}^T[i, j] = \mathbf{B}[j, i]^T$ . We then have that  $\phi(\mathbf{B}^T) = \phi(\mathbf{B})^T$ , which is what we needed to be able to define Schur complements of matrices over  $\mathcal{P}$ . For a real  $\eta \geq 0$ , we will write  $\Sigma \succeq \eta$  instead of  $\phi(\Sigma) \succeq \eta$  for ease of

notation.

### 2.2.3.2 An Efficient Algorithm for Sampling Perturbations

Genise and Micciancio [GM18] made the sampling perturbations operation efficient in the ring setting. In particular, they describe an algorithm `SampleFz` which takes as input a covariance polynomial  $f$  and a center  $c$ , and returns a sample from the corresponding Gaussian distribution over  $\mathcal{R}$ . While their method cannot be applied directly to the module setting, we use the same general ideas, some of their lemmas, and `SampleFz` to derive an algorithm that samples perturbation vectors in the module setting. Their method cannot be applied directly to the module setting because of the additional rank module parameter  $d$ . Instead of having to sample vectors with a covariance matrix of dimension  $2 \times 2$  over  $\mathcal{R}$  and with a center  $(c_0, c_1) \in \mathcal{R}^2$  as in [GM18], we have to work with a covariance matrix  $\Sigma \in \mathcal{P}^{2d \times 2d}$  and a center  $\mathbf{c} \in \mathcal{P}^{2d}$ . However, by using [GM18, Lemma 4.3] and the `SampleFz` algorithm, we wisely decompose the covariance matrices into blocks of different sizes at each iteration and update our center, allowing us to iteratively sample the perturbations  $p_i \in \mathcal{R}$ .

We now give a description of the algorithm `SamplePerturb` which, given the trapdoor  $\mathbf{T}$  and the Gaussian parameters  $\zeta$  and  $\alpha$ , returns a vector  $\mathbf{p}$  sampled from the centered discrete Gaussian over  $\mathcal{R}^m$  of covariance  $\Sigma_p = \zeta^2 \mathbf{I} - \alpha^2 \begin{bmatrix} \mathbf{T} \\ \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{T}^T & \mathbf{I} \end{bmatrix}$ . This algorithm does not explicitly use  $\Sigma_p \in \mathcal{P}^{m \times m}$ , but only a much smaller matrix  $\Sigma \in \mathcal{P}^{2d \times 2d}$ , which can be computed in advance. It uses the algorithm `SampleFz` [GM18, Section 4] to sample from discrete Gaussians over  $\mathcal{R}$ .

Note that in lines 6 and 7 of Algorithm 5, no computation is actually performed: different parts of the variables  $\Sigma$  and  $\mathbf{c}$  are just given names, for a clearer understanding.

Algorithm 5 has a complexity of  $\Theta(d^2 n \log n)$  scalar operations, if we ignore the updates to  $\Sigma$  (which only depend on  $\mathbf{T}$  and can actually be precomputed in  $\Theta(d^3 n \log n)$  in the trapdoor generation). This stems from the fact that multiplication in  $\mathcal{P}$  and `SampleFz` both take  $\Theta(n \log n)$  time.

### 2.2.3.3 Correctness

The correctness of `SamplePerturb` relies on the two following lemmas. The first one is a convolution lemma, which justifies that the output distribution is correct.

---

**Algorithm 5** SamplePerturb( $\mathbf{T}, \zeta, \alpha$ ) for sampling a perturbation vector
 

---

```

1: function SamplePerturb( $\mathbf{T} \in \mathcal{P}^{2d \times dk}, \zeta > 0, \alpha > 0$ )
2:    $\mathbf{p}_s \leftarrow D_{\mathcal{R}^{dk}, \zeta^2 - \alpha^2}$   $\triangleright \mathbf{p}_s \in \mathcal{R}^{dk}$ 
3:    $\mathbf{c} \leftarrow -\frac{\alpha^2}{\zeta^2 - \alpha^2} \mathbf{T} \mathbf{p}_s$   $\triangleright \mathbf{c} \in \mathcal{P}^{2d}$ 
4:    $\Sigma \leftarrow \zeta^2 \mathbf{I} - (\alpha^{-2} - \zeta^{-2})^{-1} \mathbf{T} \mathbf{T}^T$   $\triangleright \Sigma \in \mathcal{P}^{2d \times 2d}$ 
5:   for  $i = 2d - 1 \dots 0$  do
6:      $\Sigma = \left[ \begin{array}{c|c} \mathbf{A} & \mathbf{B} \\ \hline \mathbf{B}^T & f \end{array} \right]$   $\triangleright \mathbf{A} \in \mathcal{P}^{i \times i}, \mathbf{B} \in \mathcal{P}^{i \times 1}, f \in \mathcal{P}$ 
7:      $\mathbf{c} = (\mathbf{c}', c_i)$   $\triangleright \mathbf{c}' \in \mathcal{P}^i, c_i \in \mathcal{P}$ 
8:      $p_i \leftarrow D_{\mathcal{R}, \sqrt{f}, c_i}$   $\triangleright p_i \in \mathcal{R}$ 
9:      $\mathbf{c} \leftarrow \mathbf{c}' + f^{-1} \mathbf{B} (p_i - c_i)$   $\triangleright \mathbf{c} \in \mathcal{P}^i$ 
10:     $\Sigma \leftarrow \mathbf{A} - f^{-1} \mathbf{B} \mathbf{B}^T$   $\triangleright \Sigma \in \mathcal{P}^{i \times i}$ 
11:     $\mathbf{p} \leftarrow (p_0, \dots, p_{2d-1}, \mathbf{p}_s)$   $\triangleright \mathbf{p} \in \mathcal{R}^m$ 
12:  return  $\mathbf{p}$ 
    
```

---

**Lemma 7 ([GM18, Lemma 4.3])** For any real  $0 < \varepsilon \leq 1/2$ , positive integers  $r$  and  $s$ , vector  $\mathbf{c} = (\mathbf{c}_0, \mathbf{c}_1) \in \mathbb{R}^{r+s}$ , positive definite  $\Sigma = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B}^T & \mathbf{D} \end{bmatrix} \in \mathbb{R}^{(r+s) \times (r+s)}$  composed of blocks  $\mathbf{A} \in \mathbb{R}^{r \times r}$ ,  $\mathbf{B} \in \mathbb{R}^{r \times s}$ ,  $\mathbf{D} \in \mathbb{R}^{s \times s}$ , we define the following random process:

- $\mathbf{x}_1 \leftarrow D_{\mathbb{Z}^s, \sqrt{\mathbf{D}}, \mathbf{c}_1}$ ;
- $\mathbf{x}_0 \leftarrow D_{\mathbb{Z}^r, \sqrt{\Sigma/\mathbf{D}}, \mathbf{c}_0 + \mathbf{B} \mathbf{D}^{-1} (\mathbf{x}_1 - \mathbf{c}_1)}$ .

If  $\Sigma \succeq \eta_\varepsilon^2(\mathbb{Z}^{r+s})$ , then this process outputs a vector  $\mathbf{x} = (\mathbf{x}_0, \mathbf{x}_1) \in \mathbb{Z}^{r+s}$  whose distribution is statistically indistinguishable from  $D_{\mathbb{Z}^{r+s}, \sqrt{\Sigma}, \mathbf{c}}$ .

The second one ensures that all covariance matrices manipulated during our algorithm are "positive definite enough" so that we can rigorously apply Lemma 7 at each step of the algorithm.

**Lemma 8 ([GM18, Lemma 4.2])** Let  $\varepsilon > 0$ ,  $r$  and  $s$  be positive integers, and  $\Sigma = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B}^T & \mathbf{D} \end{bmatrix} \in \mathbb{R}^{(r+s) \times (r+s)}$  be a positive definite matrix made out of blocks  $\mathbf{A} \in \mathbb{R}^{r \times r}$ ,  $\mathbf{B} \in \mathbb{R}^{r \times s}$ ,  $\mathbf{D} \in \mathbb{R}^{s \times s}$ .

If  $\Sigma \succeq \eta_\varepsilon^2(\mathbb{Z}^{r+s})$ , then  $\mathbf{D} \succeq \eta_\varepsilon^2(\mathbb{Z}^s)$  and  $\Sigma/\mathbf{D} \succeq \eta_\varepsilon^2(\mathbb{Z}^r)$ .

The correctness of SamplePerturb is then stated in the following Theorem.

**Theorem 7** Let  $\mathbf{T} \in \mathcal{P}^{2d \times dk}$ ,  $\zeta, \alpha > 0$ , and  $\Sigma_p = \zeta^2 \mathbf{I} - \alpha^2 \begin{bmatrix} \mathbf{T} \\ \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{T}^T & \mathbf{I} \end{bmatrix} \in \mathcal{P}^{m \times m}$  be the derived perturbation covariance matrix.

If  $\Sigma_p \succeq \eta_\varepsilon^2(\mathbb{Z}^{nm})$ , then  $\text{SamplePerturb}(\mathbf{T}, \zeta, \alpha)$  returns a vector  $\mathbf{p} \in \mathcal{R}^m$  whose distribution is statistically indistinguishable from  $D_{\mathcal{R}^m, \sqrt{\Sigma_p}}$ .

*Proof.* From a high-level point of view, our algorithm works in the following way, sampling a vector  $\mathbf{p} \in \mathcal{R}^m$  from right to left. First, it samples the last  $dk$  entries of  $\mathbf{p}$  from a centered spherical discrete Gaussian. Then, it samples the remaining  $2d$  polynomials one by one, from  $p_{2d-1}$  down to  $p_0$ , each time updating the covariance matrix and the center before sampling the next one. The sampling of a single polynomial with given covariance and center is done using the algorithm  $\text{SampleFz}$  from [GM18]. First, let us observe that  $\Sigma_p$  has a particular structure. Indeed,

$$\Sigma_p = \zeta^2 \mathbf{I} - \alpha^2 \begin{bmatrix} \mathbf{T} \\ \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{T}^T & \mathbf{I} \end{bmatrix} = \left[ \begin{array}{c|c} \mathbf{A} & -\alpha^2 \mathbf{T} \\ \hline -\alpha^2 \mathbf{T}^T & (\zeta^2 - \alpha^2) \mathbf{I} \end{array} \right],$$

where  $\mathbf{A} = \zeta^2 \mathbf{I} - \alpha^2 \mathbf{T} \mathbf{T}^T \in \mathcal{P}^{2d \times 2d}$ . This allows us to use Lemma 7 to first sample a centered vector  $\mathbf{p}_s \in \mathcal{R}^{dk}$  with covariance  $(\zeta^2 - \alpha^2) \mathbf{I}$  (which is easy since the distribution is spherical), and then sample a vector with covariance  $\Sigma_p / (\zeta^2 - \alpha^2) \mathbf{I}$  and center  $\mathbf{c} = -\frac{\alpha^2}{\zeta^2 - \alpha^2} \mathbf{T} \mathbf{p}_s$ .

However,  $\zeta^2 \mathbf{I} - (\alpha^{-2} - \zeta^{-2})^{-1} \mathbf{T} \mathbf{T}^T$  is precisely the Schur complement of  $(\zeta^2 - \alpha^2) \mathbf{I}$  in  $\Sigma_p$ . The goal of what comes after sampling  $\mathbf{p}_s$  is sampling a vector  $(p_0, \dots, p_{2d-1}) \in \mathcal{R}^{2d}$  with this covariance and an updated center  $\mathbf{c}$ .

To this end, we make use of Lemma 7 iteratively during the loop. More precisely, at each iteration of the loop, we sample the rightmost remaining entry of  $\mathbf{p}$  using  $\text{SampleFz}$ . This entry's covariance is the lower-right entry of  $\Sigma$ , and its center is the last entry of  $\mathbf{c}$ . We then update both  $\Sigma$  and  $\mathbf{c}$  according to Lemma 7, and proceed with the sampling of the other entries in the same way, until we have sampled all of  $\mathbf{p}$ .

The only missing argument is that we can actually apply Lemma 7 at each step of the algorithm. Lemma 8 takes care of that, assuring that at each iteration of the loop we have  $\Sigma \succeq \eta_\varepsilon^2(\mathbb{Z}^{i+1})$ .  $\square$

### 2.2.3.4 Gaussian Parameters for Perturbation Sampling

As seen in Theorem 7, we need to have  $\Sigma_p \succeq \eta_\varepsilon^2(\mathbb{Z}^{nm})$  for SamplePerturb to be correct. Let us show how this leads to a lower bound on  $\zeta$ .

**Lemma 9** Let  $\mathbf{T} \in \mathcal{P}^{2d \times dk}$ ,  $\zeta, \alpha > 0$ , and  $\Sigma_p = \zeta^2 \mathbf{I} - \alpha^2 \begin{bmatrix} \mathbf{T} \\ \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{T}^T & \mathbf{I} \end{bmatrix}$ . For any positive real  $0 < \varepsilon \leq 1/2$ , if  $\zeta$  is such that  $\zeta > \sqrt{(\alpha^2 + 1)s_1^2(\mathbf{T}) + \eta_\varepsilon^2(\mathbb{Z}^{nm})}$  and  $\zeta > \sqrt{\alpha^2 + \eta_\varepsilon^2(\mathbb{Z}^{nm})}$ , then we have  $\Sigma_p \succ \eta_\varepsilon^2(\mathbb{Z}^{nm})$ .

*Proof.* For the sake of simplicity, we will write  $\eta = \eta_\varepsilon(\mathbb{Z}^{nm})$ . We want the matrix  $\Sigma_p - \eta^2 \mathbf{I}$  to be positive definite. Since

$$\Sigma_p - \eta^2 \mathbf{I} = \left[ \begin{array}{c|c} \mathbf{M} & -\alpha^2 \mathbf{T} \\ \hline -\alpha^2 \mathbf{T}^T & (\zeta^2 - \alpha^2 - \eta^2) \mathbf{I} \end{array} \right],$$

where  $\mathbf{M} = (\zeta^2 - \eta^2) \mathbf{I} - \alpha^2 \mathbf{T} \mathbf{T}^T \in \mathcal{R}^{2d \times 2d}$ , this condition is equivalent to having both the block  $(\zeta^2 - \alpha^2 - \eta^2) \mathbf{I}$  and its Schur complement be positive definite. The former trivially is if  $\zeta^2 - \alpha^2 - \eta^2 > 0$ ; for the latter we need to take a closer look at the Schur complement in question.

This Schur complement is equal to  $(\zeta^2 - \eta^2) \mathbf{I} - \alpha^2 \frac{\zeta^2 - \eta^2}{\zeta^2 - \alpha^2 - \eta^2} \mathbf{T} \mathbf{T}^T$ . It is positive definite if and only if  $\mathbf{I} - \frac{\alpha^2}{\zeta^2 - \alpha^2 - \eta^2} \mathbf{T} \mathbf{T}^T$  is. A sufficient condition for that is having  $\frac{\alpha^2}{\zeta^2 - \alpha^2 - \eta^2} s_1^2(\mathbf{T}) < 1$ , which in turn yields  $\zeta^2 > (\alpha^2 + 1)s_1^2(\mathbf{T}) + \eta^2$ .  $\square$

To instantiate concretely SamplePerturb, we then need to determine an upper bound on  $s_1(\mathbf{T})$ , knowing that  $\mathbf{T}$  follows a Gaussian distribution over  $\mathcal{P}^{2d \times dk}$  of parameter  $\sigma$ . If  $\mathbf{T}$  was a  $2nd \times ndk$  unstructured matrix over  $\mathbb{R}$ , one could apply Lemma 2.9 of [MP12], it would state that  $s_1(\mathbf{T}) < C\sigma(\sqrt{2nd} + \sqrt{ndk} + c)$  for a certain universal constant  $C > 0$ , except with probability at most  $2\exp(-\pi c^2)$ . While in the case where  $\mathbf{T}$  is structured, there is no similar result that we know of, we can assume that such a property still holds in our case, which we confirmed empirically.

We choose  $c = 4.7$  to ensure that the probability  $2\exp(-\pi c^2)$  is less than  $2^{-100}$ . During our experiments, we found the constant  $C$  to be less than 1.1. We generated matrices  $\mathbf{T}$  with the same structure and size as the ones used in our signature schemes (see Table 2.5 for concrete values for  $n, k, d$  and  $\sigma$ ), computed their exact spectral norm and their expected spectral norm, and deduced the constant  $C$ .

## 2.2.4 Sampling Gaussian Preimages

Once we are able to perform both  $\mathbf{G}$ -sampling and perturbation sampling, we combine these two operations to sample from the spherical Gaussian of parameter  $\zeta$  on a coset  $\Lambda_q^u(\mathbf{A})$ . The method is adapted from [MP12] to the module setting, yielding Algorithm 6 called SamplePre.

We remind the following conditions on  $\alpha$  and  $\zeta$  for the correctness of the algorithm:

- $\alpha \geq \sqrt{2b} \cdot (2b + 1) \cdot \sqrt{\log(2k(1 + 1/\varepsilon))}/\pi$ ,
- $\zeta > \sqrt{(\alpha^2 + 1)s_1^2(\mathbf{T}) + \eta_\varepsilon^2(\mathbb{Z}^{nm})}$ ,  
 knowing that  $s_1(\mathbf{T}) < 1.1\sigma(\sqrt{2nd} + \sqrt{ndk} + 4.7)$  with high probability.

Under those conditions, the preimage sampling algorithm, described in Algorithm 6 below, is correct.

---

**Algorithm 6** SamplePre( $\mathbf{A}, \mathbf{T}, \mathbf{H}, \mathbf{u}, \zeta, \alpha$ ) for sampling from a discrete Gaussian of parameter  $\zeta$  over  $\Lambda_q^u(\mathbf{A})$

---

```

1: function SamplePre( $\mathbf{A} \in \mathcal{R}_q^{d \times m}, \mathbf{T} \in \mathcal{R}^{2d \times dk}, \mathbf{H} \in \mathcal{R}_q^{d \times d}, \mathbf{u} \in \mathcal{R}_q^d, \zeta > 0, \alpha > 0$ )
2:    $\mathbf{p} \leftarrow \text{SamplePerturb}(\mathbf{T})$   $\triangleright \mathbf{p} \in \mathcal{R}^m$ 
3:    $\mathbf{v} \leftarrow \mathbf{H}^{-1}(\mathbf{u} - \mathbf{A}\mathbf{p})$   $\triangleright \mathbf{v} \in \mathcal{R}_q^d$ 
4:    $\mathbf{z} \leftarrow D_{\Lambda_q^u(\mathbf{G}), \alpha}$   $\triangleright \mathbf{z} \in \mathcal{R}^{dk}$ 
5:    $\mathbf{x} \leftarrow \mathbf{p} + \begin{bmatrix} \mathbf{T} \\ \mathbf{I} \end{bmatrix} \mathbf{z}$   $\triangleright \mathbf{x} \in \mathcal{R}^m$ 
6:   return  $\mathbf{x}$ 
    
```

---

**Theorem 8** Let  $\mathbf{T} \in \mathcal{R}^{2d \times dk}$  be a trapdoor for the matrix  $\mathbf{A} \in \mathcal{R}_q^{d \times m}$  with tag  $\mathbf{H} \in \mathcal{R}_q^{d \times d}$ ,  $\mathbf{u} \in \mathcal{R}_q^d$  be a target vector, and  $\zeta$  and  $\alpha$  be Gaussian parameters with the above constraints. Then, SamplePre( $\mathbf{A}, \mathbf{T}, \mathbf{H}, \mathbf{u}, \zeta, \alpha$ ) outputs a vector whose distribution is statistically close to  $D_{\Lambda_q^u(\mathbf{A}), \zeta}$ .

## 2.2.5 Implementation and Performance

### 2.2.5.1 Implementation Choices

To generate our specific discrete Gaussian distributions, we make use of the following building blocks: the AES-based pseudorandom number generator from [MN17] (implemented using AES-NI instructions for x86 architectures), and a sampler of discrete Gaussians over  $\mathbb{Z}$  similar to Karney’s sampler [Kar16]. We chose this sampler as it can

generate samples in constant time, independently of the center, Gaussian parameter, and output value. All of the computations that deal with non-integers are carried out with floating-point operations that do not involve subnormal numbers.

To obtain an efficient arithmetic in  $\mathcal{P} = \mathbb{R}[X]/\langle X^n + 1 \rangle$  we used the Chinese Remainder Transform (CRT, as defined in [LPR13]), as in several other works [DP16; GM18; GPR<sup>+</sup>18]. It is a kind of fast Fourier transform that evaluates a polynomial  $f \in \mathcal{P}$  at the complex primitive  $2n^{\text{th}}$  roots of unity, the  $n$  points of the form  $\omega_i = e^{\frac{ki\pi}{n}}$  for  $i \in \{1, 3, \dots, 2n - 1\}$ , in time  $\Theta(n \log n)$ . As explained in [GM18, Section 4.1], this CRT transform combines especially well with the algorithm `SampleFz` whose recursive structure is similar to that of an FFT.

Also, the matrix  $\Sigma$  is not actually updated during a run of `SamplePerturb`. Instead, we precompute (during the trapdoor generation) all of the  $2d$  values that it would take during the execution of the algorithm, and store them in a single  $2d \times 2d$  triangular matrix by "stacking" them. This is possible because at each iteration of the loop,  $\Sigma$  is an  $i \times i$  matrix of which we only use the last line and column. This comes with an additional storage cost of  $d(2d + 1)$  elements from  $\mathcal{P}$  in the secret key, and Table 2.3 quantifies the time gains in practice.

Our implementation is constant-time, assuming the compiler produces constant-time code for reduction modulo  $q$  and basic operations such as integer division and multiplication. Indeed, our algorithms do not require branching or memory access that depend on secret values. In particular, our sampler of discrete Gaussians over  $\mathbb{Z}$ 's running time is independent of both the input parameters and the output value.

### 2.2.5.2 Timings

We now present running times for our trapdoor generation and preimage sampling algorithms, and the cost of their different components. Our experimentations were carried out with  $n = 256$ ,  $k = \lceil \log_b q \rceil = 30$ , and values of  $d$  up to 10. We ran them on an Intel i7-8650U CPU running at 1.90 GHz.

In Table 2.3, we see how the trapdoor generation is divided into three main operations: sampling from  $D_{\mathbb{Z}, \sigma}$  for the construction of  $\mathbf{T}$ , the precomputations concerning the covariance matrices, and arithmetic, which is mainly computing the matrix product.

Table 2.4 concerns the algorithm `SamplePre`. We also measured that sampling from discrete Gaussians over  $\mathbb{Z}$  constitutes 57-64% of the perturbation sampling (decreasing with  $d$ ) and about 85% of the  $\mathbf{G}$ -sampling, for a total of 67-72% of the whole presam-



Table 2.3 – Running time of the TrapGen algorithm.

$d$	4	6	8	10
$D_{\mathbb{Z},\sigma}$ sampling	27.17 ms (74%)	56.37 ms (72%)	100.22 ms (67%)	159.10 ms (64%)
$\Sigma$ computations	7.03 ms (19%)	17.11 ms (22%)	39.40 ms (26%)	71.92 ms (29%)
Arithmetic	2.34 ms (6%)	1.11 ms (1%)	2.60 ms (2%)	5.25 ms (2%)
Total	36.56 ms	78.52 ms	149.57 ms	248.09 ms

pling. Gaussian sampling over  $\mathbb{Z}$  makes up most of the running times of both TrapGen and SamplePre. As such, it is important for efficiency to use a fast sampler of discrete Gaussians over  $\mathbb{Z}$  as a building block. As a reminder, in our implementation, this sampler can easily be swapped out for another if needed.

Table 2.4 – Running time of the SamplePre algorithm.

$d$	4	6	8	10
Perturb. sampling	4.73 ms (36%)	6.63 ms (38%)	9.43 ms (38%)	12.03 ms (39%)
$\mathbf{G}$ -sampling	7.48 ms (57%)	9.83 ms (56%)	13.29 ms (54%)	16.43 ms (53%)
Arithmetic	0.82 ms (6%)	1.20 ms (7%)	1.98 ms (8%)	2.64 ms (8%)
Total	13.28 ms	17.66 ms	24.70 ms	31.10 ms

## 2.3 A GPV Signature Scheme on Modules

In 2008, Gentry, Peikert and Vaikuntanathan [GPV08] proposed a generic framework to construct lattice-based hash-and-sign signature schemes originally described with general lattices. In [BB13], the authors proposed an instantiation of the GPV framework using the trapdoor from [MP12] both for general and for ring lattices. We propose to instantiate the GPV signature scheme using [MP12; GM18] but adapted to the module setting.

### 2.3.1 The Scheme

A direct application of our Gaussian preimage sampling techniques on module lattices is the GPV signature [GPV08] in the module setting. It was originally formulated on unstructured lattices, and has previously been implemented using improved trapdoors and algorithms [MP12; GM18] in the ring setting [BB13; GPR<sup>+</sup>18; GPR<sup>+</sup>19].

Our goal here is not to obtain a competitive signature scheme, but rather to show the relevance of the tools we developed.

In this hash-and-sign scheme, key generation is simply trapdoor generation, and signing consists in hashing a message to a point in space and then sampling a Gaussian preimage of that point.

We base ourselves on the probabilistic GPV signature [GPV08, Section 6.2], where a random salt is appended to a message before hashing it. Then, if the same message  $M$  is signed several times, one does not obtain multiple samples from the same coset  $\Lambda_q^u(\mathbf{A})$ , which would compromise the security of the scheme.

**Parameters.** The parameters are:  $n$  a power of two,  $q$  a prime modulus such that  $q \equiv 1 \pmod{2n}$ ,  $k = \lceil \log_b q \rceil$  its size in the base- $b$  expansion,  $d$  a positive integer,  $m = d(k+2)$ ,  $\sigma$ ,  $\zeta$ , and  $\alpha$  Gaussian parameters,  $t$  the Gaussian tailcut, and  $s$  the length of the salt.

**Description.** We describe the resulting scheme below, where  $\mathcal{H} : \{0, 1\}^* \rightarrow \mathcal{R}_q^d$  is a collision-resistant hash function. We added a random salt to the message before hashing it, as described in [GPV08, Section 6.2]. This is so that if the same message  $M$  is hashed multiple times, one does not obtain multiple samples from the same coset  $\Lambda_q^u(\mathbf{A})$ , which is needed for the security proof.

- $\text{KeyGen}(1^n) \rightarrow (vk, sk)$ 
  - $(\mathbf{A}, \mathbf{T}) \leftarrow \text{TrapGen}(\mathbf{I}, \sigma)$
  - $vk \leftarrow \mathbf{A} \in \mathcal{R}_q^{d \times m}$
  - $sk \leftarrow \mathbf{T} \in \mathcal{R}^{2d \times dk}$
- $\text{Sign}(1^n, sk, M) \rightarrow \nu$ 
  - $S \leftarrow \mathcal{U}(\{0, 1\}^s)$
  - $\mathbf{u} \leftarrow \mathcal{H}(M \parallel S)$
  - $\mathbf{x} \leftarrow \text{SamplePre}(\mathbf{A}, \mathbf{T}, \mathbf{I}, \mathbf{u}, \zeta, \alpha)$
  - $\nu \leftarrow (\mathbf{x}, S) \in \mathcal{R}^m \times \{0, 1\}^s$
- $\text{Verify}(1^n, vk, M, \nu) \rightarrow \{\text{accept}, \text{reject}\}$ 
  - $\mathbf{u} \leftarrow \mathcal{H}(M \parallel S)$
  - Accept if  $\mathbf{A}\mathbf{x} = \mathbf{u} \pmod{q}$  and  $0 < \|\mathbf{x}\| \leq t\zeta\sqrt{mn}$

**Correctness.** The following Theorem 9 states the correctness of this signature scheme.

**Theorem 9** Let  $n, q, k, d, m, \sigma, \zeta, \alpha, t, s$  be the parameters of the scheme as above. Then, the scheme is correct, and admits EU-CMA security in the random oracle model under the hardness of  $\text{Module-LWE}_{n,d,q,\sigma}$  and  $\text{Module-SIS}_{n,d,m,q,\beta}$ , with  $\beta = 2\zeta\sqrt{mn}$ .

### 2.3.2 Parameters and Security

We describe how we chose the parameters for our GPV signature scheme, estimating the difficulty of the underlying lattice problems. Our Gaussian sampling algorithms also impose constraints on our parameters.

**Length of the Salt.** To make sure the probability of collision (two messages landing in the same coset) is negligible, we follow [FHK<sup>+</sup>17] and take the length of the random salt to be  $s \geq \lambda + \log q_s$ , where  $\lambda$  is the security parameter and  $q_s$  the number of allowed signing queries.

**Key Recovery.** The computational instantiation of the trapdoors we use is based on the hardness of  $\text{Module-LWE}_{n,d,q,\sigma}$ , where we choose the Gaussian parameter  $\sigma$  to be at least the smoothing parameter  $\eta_\varepsilon(\mathbb{Z}^{nd})$  where  $\varepsilon$  is taken as  $2^{-\lambda}$  where  $\lambda$  is the target security parameter of the scheme. To get an idea of the security provided by such a  $\text{Module-LWE}$  instance, we use the estimator of [APS15]<sup>1</sup> and approximate it by an instance of an unstructured LWE problem in dimension  $nd$ .

**Signature Forgery.** Forging a signature consists in finding a short vector  $\mathbf{x} \in \mathcal{R}^m$  of norm less than  $\beta = t\zeta\sqrt{mn}$  satisfying the relation  $\mathbf{A}\mathbf{x} = \mathbf{u} \bmod q$  for some  $\mathbf{u} \in \mathcal{R}_q^n$  that depends on the message. This is exactly an instance of  $\text{Module-ISIS}_{n,d,m,q,\beta}$ , which we approximate by an instance of unstructured  $\text{ISIS}_{nd, nm, q, \beta}$ . This ISIS problem can be solved by finding a short vector in  $\Lambda_q^{\mathbf{u}}(\mathbf{A}')$  for some sub-matrix  $\mathbf{A}'$  of  $\mathbf{A}$ . To estimate the cost of computing such a solution, we look at the cost of running the BKZ algorithm [SE94] to reduce a basis of  $\Lambda_q^{\mathbf{u}}(\mathbf{A}')$  in order to get a sufficiently short vector.

**BKZ Reduction Cost Model.** The BKZ algorithm reduces a basis by calling an exact SVP oracle in a smaller dimension  $b$ , called the BKZ blocksize. We follow the (very pessimistic) core-SVP hardness introduced in [ADP<sup>+</sup>16], where the cost of a BKZ

1. <https://bitbucket.org/malb/lwe-estimator> (commit a2296b8)

algorithm with blocksize  $b$  is taken to be the cost of only one call to an SVP oracle in dimension  $b$ , rather than a polynomial number of calls. We consider the sieving algorithm as our exact SVP oracle, its complexity in dimension  $b$  is estimated to  $2^{cb}$ , where  $c = \log \sqrt{3/2} \approx 0.292$  in the classical setting, and  $c = \log \sqrt{13/9} \approx 0.265$  in the quantum setting.

**Estimating Security and Choosing Parameters.** In Table 2.5, we propose four parameter sets and corresponding security estimates, taking the prime modulus  $q = 1073738753$  of bitsize  $k = 30$ . The sets I and IV correspond to the ring setting, where  $n$  is a power of two and  $d = 1$ . The sets II and III are intermediate using the module setting.

Table 2.5 – Suggested parameter sets for our instantiation of the GPV signature.

Parameter set	I	II	III	IV
$nd$	1024	1280	1536	2048
$n$	1024	256	512	2048
$k$	30	30	30	30
$d$	1	5	3	1
$\sigma$	7.00	5.55	6.15	6.85
$\alpha$	48.34	54.35	60.50	67.40
$\zeta$	83832	83290	112522	160778
BKZ blocksize $b$ to break LWE	367	478	614	896
Classical security	107	139	179	262
Quantum security	97	126	163	237
BKZ blocksize $b$ to break SIS	364	482	583	792
Classical security	106	140	170	231
Quantum security	96	127	154	210

### 2.3.3 Implementation and Performance

Our GPV implementation relies on our trapdoor tools, as well as a Number Theoretic Transform for fast multiplication in  $\mathcal{R}_q$ , adapted from CRYSTALS-Kyber [DKL<sup>+</sup>18].

We now present in Table 2.6 the running times for our implementation of the GPV signature scheme. While it is practical and runs on a standard laptop in acceptable time, the comparison with lattice-based NIST candidates given in Table 2.11 shows that it is not competitive.

Table 2.6 – Performances of our GPV signature.

Parameter set	KeyGen	Sign	Verify
I	7.48 ms	11.47 ms	0.73 ms
II	51.34 ms	15.25 ms	1 ms
III	36.49 ms	17.45 ms	1.12 ms
IV	15.55 ms	22.64 ms	1.48 ms

### 2.3.3.1 Comparison Between Rings and Modules

As already explained, one goal of using a module variant instead of a ring variant is to be more flexible in the choice of parameters. The comparison between the different levels of security shows that the running time for signing and verifying is increasing with  $nd$ , and then that having intermediate levels allow to be faster to sign and verify.

On the other hand, the KeyGen algorithm does not depend only on  $nd$  but is slower for larger  $d$ . We give a more concrete example of this in Table 2.7. When  $nd$  is constant, so is the estimated security provided. With a higher  $n$  and a lower  $d$  ( $d = 1$  being the ring setting), the underlying lattices have a stronger structure and the signature is more efficient. With a lower  $n$  and a higher  $d$  (the extreme being  $n = 1$  in the unstructured setting), the lattices have less structure, leading to increased flexibility at the cost of efficiency.

Table 2.7 – Cost of KeyGen, Sign and Verify depending of the parameter  $d$  for  $nd = 1024$ .

$(n, d)$	KeyGen	Sign	Verify
(1024, 1)	$7.62 + 1.32 = 8.94$ ms	13.08 ms	0.79 ms
(512, 2)	$15.32 + 2.81 = 18.13$ ms	13.20 ms	0.79 ms
(256, 4)	$29.53 + 7.03 = 36.56$ ms	13.36 ms	0.74 ms

### 2.3.3.2 Comparison with [GPR<sup>+</sup>19]

In Table 2.8, we compare our timings with the work of [GPR<sup>+</sup>19]. In their paper, the authors present some software implementations of the GPV digital signature scheme. Their implementation benefits in particular from the use of bases larger than 2 for the definition of the gadget vector. We also did use larger bases in our implementation, which allows better performance for the execution times and storage requirements of our algorithms.

Their parameter set where  $(n, k) = (1024, 27)$  is compared with ours where  $(nd, k) = (1024, 30)$ , which provides approximately the same security.

Table 2.8 – Comparison of GPV implementations.

Implementation		KeyGen	Sign	Verify
[GPR <sup>+</sup> 19]	$n = 1024$	5.86 ms	32.42 ms	0.28 ms
This work	$(n, d) = (1024, 1)$	$7.62 + 1.32 = 8.94$ ms	13.08 ms	0.79 ms

## 2.4 A Standard Model Signature Scheme on Modules

The second application of our tools is an implementation of a signature scheme that is proven secure in the standard model, as opposed to the GPV signature and the NIST schemes.

This scheme is the signature from [BFR<sup>+</sup>18], which is a variant of GPV, adapted to the module setting. For the security proof to hold, the encoding must fulfil a strong injectivity property. However, the original encoding described in [BFR<sup>+</sup>18] did not meet these requirements, leading to a limited security. We propose a modified version of this scheme: we translated it to the module setting, and instantiated it with a correct encoding.

We give a complete description of our scheme and state its correctness and security in this section.

### 2.4.1 The Scheme

The proposed signature scheme is a module adaptation of the scheme from [BFR<sup>+</sup>18] with a different instantiation of the FRD encoding.

**Parameters.** We have:  $n \geq r > 1$  two powers of 2,  $q$  a prime modulus such that  $q \equiv 2r + 1 \pmod{4r}$ ,  $k = \lceil \log_b q \rceil$ ,  $d$  a positive integer,  $m = d(k + 2)$ ,  $\sigma$ ,  $\zeta$ , and  $\alpha$  Gaussian parameters, and  $t$  the tailcut for the Gaussian  $D_{\mathbb{Z}, \sigma}$ .

**Description.** We describe this scheme below, where  $H : \mathcal{M} \rightarrow \mathcal{R}_q^{d \times d}$  is an FRD encoding instantiated as explained above. Note that during the trapdoor generation, we use the tag  $\mathbf{H} = \mathbf{0}$ , meaning that  $\mathbf{T}$  is not technically a trapdoor for  $\mathbf{A}$  since  $\mathbf{H}$  is singular. Still, the scheme is correct because, for any message  $M \in \mathcal{M}$ ,  $\mathbf{T}$  is a trapdoor for  $\mathbf{A}_M = \mathbf{A} + [\mathbf{0}_{d, 2d} \mid H(M)\mathbf{G}]$  with tag  $H(M)$ .

- $\text{KeyGen}(1^n) \longrightarrow (vk, sk)$ 
  - $(\mathbf{A}, \mathbf{T}) \leftarrow \text{TrapGen}(\mathbf{0}, \sigma)$
  - $vk \leftarrow \mathbf{A} \in \mathcal{R}_q^{d \times m}$
  - $sk \leftarrow \mathbf{T} \in \mathcal{R}_q^{2d \times dk}$
- $\text{Sign}(1^n, sk, M) \longrightarrow \boldsymbol{\nu}$ 
  - $\mathbf{H}_M \leftarrow H(M)$
  - $\mathbf{A}_M \leftarrow \mathbf{A} + \begin{bmatrix} \mathbf{0}_{d,2d} & | & \mathbf{H}_M \mathbf{G} \end{bmatrix}$
  - $\boldsymbol{\nu} \leftarrow \text{SamplePre}(\mathbf{A}_M, \mathbf{T}, \mathbf{H}_M, \mathbf{0}, \zeta, \alpha)$
- $\text{Verify}(1^n, vk, M, \boldsymbol{\nu}) \longrightarrow \{\text{accept}, \text{reject}\}$ 
  - $\mathbf{H}_M \leftarrow H(M)$
  - $\mathbf{A}_M \leftarrow \mathbf{A} + \begin{bmatrix} \mathbf{0}_{d,2d} & | & \mathbf{H}_M \mathbf{G} \end{bmatrix}$
  - Accept if  $\mathbf{A}_M \boldsymbol{\nu} = \mathbf{0} \pmod q$  and  $0 < \|\boldsymbol{\nu}\| \leq t\zeta\sqrt{mn}$

**Correctness.** We state the correctness of this scheme in the following Theorem 10.

**Theorem 10** Let  $n, r, q, k, d, m, \sigma, \zeta, \alpha, t$  be the parameters of the scheme as above. Then, the scheme is correct, and is SU-CMA secure under the hardness of  $\text{Module-LWE}_{n,d,q,\sigma}$  and  $\text{MSIS}_{n,d,2d,q,\beta}$ , where  $\beta = (1 + s_1(\mathbf{T}))t\zeta\sqrt{mn}$ .

## 2.4.2 Parameters and Security

Concerning the choice of parameters, we take into account the best-known attacks rather than the proof of security, as is the case in most schemes [ABB<sup>+</sup>19; DKL<sup>+</sup>18]. In our case, it consists in finding a vector of norm at most  $\zeta$  in  $\Lambda_q^\perp(\mathbf{A}_M)$ , given  $\mathbf{A}$  and  $M \in \mathcal{M}$ . As such, the analysis of security is exactly the same as the one we did for GPV in Section 2.3.

The only factor influencing the choice of  $r$  is the number of bits of security we aim at. For a given level of security of  $\lambda$  bits, we want to be able to encode at least  $2^\lambda$  messages to prevent brute-force attacks. Since the message space  $\mathcal{M}$  is of size  $q^{n/r}$ , this yields the relation  $r < \frac{n \log q}{\lambda}$ . We would like to take the biggest  $r$  possible to have an efficient arithmetic in  $\mathcal{R}_q$ .

This scheme's suggested parameter sets are then the same as those of our ROM scheme (described in Table 2.5), except for  $r$  and  $q$ . For the parameter sets I and II we take

$(r, q) = (64, 1073741441)$ , and for set III we take  $(r, q) = (32, 1073740609)$ . The prime moduli are chosen so that they fulfill the congruence condition involving  $r$ .

### 2.4.3 Implementation and Performance

The main point that differs from our ROM scheme in the implementation is the arithmetic over  $\mathcal{R}_q$ . While without having  $q \equiv 1 \pmod{2n}$  one cannot use the NTT, we can still make use of the structure of our ring to speed up the multiplication of polynomials. Described at a high level, what we perform is a "partial NTT". To multiply polynomials, we first reduce them modulo all the  $X^{n/r} - s_i$  in  $\Theta(n \log r)$  operations (see Section 1.4.2.2). Then, we multiply them in the smaller rings  $\mathbb{Z}_q[X]/\langle X^{n/r} - s_i \rangle$  by using the Karatsuba multiplication algorithm, and reducing them both modulo  $q$  and modulo the  $X^{n/r} - s_i$ . The result can then be mapped back to the ring  $\mathcal{R}_q$  in time  $\Theta(n \log r)$  using an inverse transform. These ideas were formulated in [LS18].

As such, the main difference with GPV in terms of implementation is the use of a partial NTT inspired by [LS18], instead of a full one. An example of performances of our signatures is given in Table 2.9. For this set of parameters, the public key has size 508kB, the private key 5.06MB and the signature 131kB.

Table 2.9 – Performances of our signatures and comparison with previous GPV implementation (96-bit security parameter sets, lattice dimension 1024, modulus  $q \approx 2^{30}$ ).

Scheme	KeyGen	Sign	Verify
GPV ([GPR <sup>+</sup> 19])	5.86 ms	32.42 ms	0.28 ms
GPV (this work)	8.94 ms	13.08 ms	0.29 ms
BFRS (this work)	9.46 ms	15.66 ms	1.19 ms

In Table 2.10, we present the performance of our implementation of this standard model scheme and, in particular, we highlight the additional cost compared to our ROM scheme.

Table 2.10 – Performances of our standard model signature.

Parameter set	KeyGen	Sign	Verify
I	9.46 ms (+27%)	15.66 ms (+37%)	1.19 ms (+63%)
II	73.41 ms (+43%)	21.92 ms (+44%)	2.23 ms (+123%)
III	51.79 ms (+42%)	29.11 ms (+66%)	2.37 ms (+112%)



We do not give a comparison with the implementation of [BFR<sup>+</sup>18] as it would not be relevant, given the limited security provided by their instantiation of the FRD encoding.

Here we compare our signature schemes with the three lattice-based signatures of the second round of the NIST standardization process: Dilithium [DKL<sup>+</sup>18], qTESLA [ABB<sup>+</sup>19], and Falcon [FHK<sup>+</sup>17]. The timings are shown in Table 2.11. They were obtained using an Intel i7-8650U CPU running at 1.90 GHz. We also explain why GPV is much less efficient than those, which is not only because our implementation is not as optimized.

Table 2.11 – Performances of our signatures and comparison with other schemes (128-bit security parameter sets).

Scheme	KeyGen	Sign	Verify
Dilithium (ROM)	0.04 ms	21 530 op/sec	30 709 op/sec
qTESLA (ROM)	0.33 ms	7 213 op/sec	46 570 op/sec
Falcon (ROM)	6.24 ms	7 789 op/sec	38 647 op/sec
This work (ROM)	7.48 ms	87 op/sec	1 370 op/sec
This work	9.46 ms	64 op/sec	840 op/sec

On the one hand, Fiat-Shamir schemes such as Dilithium and qTESLA do not come with the high cost of trapdoors associated to hash-and-sign lattice-based schemes. On the other hand, Falcon does use lattice trapdoors, but still remains efficient because it uses a Klein-like sampler (see Section 1.3.2.5) and relies on NTRU lattices. This leads to smaller parameters for the scheme, which in turn yields better performances. However, this gain in efficiency comes at the expense of an additional assumption due to the class of lattices used, which is not our case. The two schemes that use discrete Gaussian sampling, qTESLA and Falcon, then require many fewer calls to the discrete Gaussian sampler over  $\mathbb{Z}$ , which is the main practical cost of our scheme (see Table 2.1).

## 2.5 An Identity-Based Encryption Scheme on Modules

Finally, we built a more advanced construction based on our tools: a standard model identity-based encryption scheme. The definitions of such a scheme as well as the notions of correctness and security can be found in Section 1.4.2.1.

This IBE scheme is the identity-based encryption from [ABB10a; BFR<sup>+</sup>18] but

adapted to the module setting, with a different instantiation of the FRD encoding (see Section 1.4.2.2). Basically, it is the signature scheme presented in Section 2.4 combined with the Dual-Regev encryption scheme, which was presented in Section 1.4.1.1.

**The Scheme** The scheme is as follows, where  $H : \mathcal{M} \rightarrow \mathcal{R}_q^{d \times d}$  is an FRD encoding instantiated as explained in Section 1.4.2.2.

- $\text{Setup}(1^n) \rightarrow (mpk, msk)$ 
  - $(\mathbf{A}, \mathbf{T}) \leftarrow \text{TrapGen}(\mathbf{0}, \sigma)$
  - $\mathbf{u} \leftarrow \mathcal{U}(\mathcal{R}_q^d)$
  - $mpk \leftarrow (\mathbf{A}, \mathbf{u}) \in \mathcal{R}_q^{d \times (m+1)}$
  - $msk \leftarrow \mathbf{T} \in \mathcal{R}_q^{2d \times dk}$
- $\text{Extract}(1^n, mpk = (\mathbf{A}, \mathbf{u}), msk = \mathbf{T}, id \in \mathcal{ID}) \rightarrow \mathbf{sk}_{id}$ 
  - $\mathbf{H}_{id} \leftarrow H(id)$
  - $\mathbf{A}_{id} \leftarrow \mathbf{A} + \begin{bmatrix} \mathbf{0}_{d,2d} & | & \mathbf{H}_{id}\mathbf{G} \end{bmatrix}$
  - $\mathbf{sk}_{id} \leftarrow \text{SamplePre}(\mathbf{A}_{id}, \mathbf{T}, \mathbf{H}_{id}, \mathbf{u}, \zeta, \alpha)$
- $\text{Encrypt}(1^n, mpk = (\mathbf{A}, \mathbf{u}), id, M) \rightarrow C$ 
  - $\mathbf{H}_{id} \leftarrow H(id)$
  - $\mathbf{A}_{id} \leftarrow \mathbf{A} + \begin{bmatrix} \mathbf{0}_{d,2d} & | & \mathbf{H}_{id}\mathbf{G} \end{bmatrix}$
  - $\mathbf{s} \leftarrow \mathcal{U}(\mathcal{R}_q^d), \mathbf{e}_0 \leftarrow D_{\mathcal{R}^{m-k}, \tau}, \mathbf{e}_1 \leftarrow D_{\mathcal{R}^k, \gamma}, e' \leftarrow D_{\mathcal{R}, \tau}$ .
  - $\mathbf{b} \leftarrow (\mathbf{s}^t \mathbf{A}_{id})^t + (\mathbf{e}_0^t \mid \mathbf{e}_1^t)^t$  et  $c \leftarrow \mathbf{s}^t \mathbf{u} + e' + \lfloor q/2 \rfloor M$
  - $C \leftarrow (\mathbf{b}, c) \in \mathcal{R}_q^{m+1}$
- $\text{Decrypt}(1^n, \mathbf{sk}_{id} = \mathbf{x}, C = (\mathbf{b}, c)) \rightarrow M$ 
  - $res \leftarrow c - \mathbf{b}^t \mathbf{x}$ .

For each  $res_i$ , if it is closer to  $\lfloor q/2 \rfloor$  than to 0,  $M_i = 1$ , otherwise  $M_i = 0$ .

**Correctness.** Let us note  $\mathbf{x} = (\mathbf{x}_0^t \mid \mathbf{x}_1^t)^t$  with  $\mathbf{x}_0 \in \mathcal{R}_q^{m-k}$  and  $\mathbf{x}_1 \in \mathcal{R}_q^k$ . Decrypting a ciphertext then needs the error term  $e' - (\mathbf{e}_0^t \mid \mathbf{e}_1^t)(\mathbf{x}_0 \mid \mathbf{x}_1)^t = e' - \mathbf{e}_0^t \mathbf{x}_0 - \mathbf{e}_1^t \mathbf{x}_1$  to be bounded by  $\lfloor q/4 \rfloor$ .

**Parameters and Security.** The correctness imposes the following conditions on the parameters  $\gamma$  and  $\tau$  (see [BFR<sup>+</sup>18], Section 4.4):  $t\tau\sqrt{n} + 2t^2\tau\zeta n + t^2\gamma\zeta kn < \lfloor q/4 \rfloor$ . Moreover, we take  $\gamma = 2t\sigma\tau\sqrt{n}$  so that the security proof of the scheme holds.

**Theorem 11** Our IBE construction with parameters  $n, m, q, k, \sigma, \alpha, \gamma, \tau$  and  $\zeta$  is IND-sID-CPA secure in the standard model under the hardness of Module-LWE $_{n,q,D_{\mathcal{R}},\tau}$ .

### 2.5.1 Implementation and Performance

As in our standard model signature scheme, we make use of our ring to speed up the multiplication of polynomials by performing a partial NTT. We make use of the same encoding as in the previous section, which imposes the condition  $q \equiv 2r + 1 \pmod{4r}$  on the modulus, to map identities in  $\mathcal{M} = \mathbb{Z}_q^{n/r} \setminus \{\mathbf{0}\}$  to invertible elements in  $\mathcal{R}_q^{d \times d}$ . In Table 2.12, we present the performance of our implementation of this standard model IBE scheme.

Table 2.12 – Timings of the operations of our scheme: Setup, Extract, Encrpt, and Decrypt

Parameter Set	Setup	Extract	Encrypt	Decrypt
I	9.82 ms	16.54 ms	4.87 ms	0.99 ms
II	44.91 ms	18.09 ms	5.48 ms	1.04 ms

In Table 2.13, we give timings for the different operations of some IBE schemes. Our timings could seem worse than the ones in [BFR<sup>+</sup>18] but the two implementations cannot be compared as the latter’s limited security would make the comparison irrelevant. A part of the difference comes from the arithmetic we need to use in order to build a proper FRD encoding (see Section 1.4.2.2). Moreover, in contrast to [DLP14], we did not use NTRU lattices, which explains the differences in the timings.

Table 2.13 – Timings of the operations for some IBE schemes.

Scheme	$(\lambda, n)$	Setup	Extract	Encrypt	Decrypt
BF-128 [Fou13]	(128, –)	–	0.55 ms	7.51 ms	5.05 ms
DLP-14 [MSO17]	(80, 512)	4.034 ms	3.8 ms	0.91 ms	0.62 ms

# IDENTITY-BASED ENCRYPTION SCHEME USING APPROXIMATE TRAPDOORS

---

This chapter’s content is mainly based on a joint work [IPR23] with Malika Izabachène and Adeline Roux-Langlois. The corresponding implementation is available at [https://github.com/lucasprabel/approx\\_lattice](https://github.com/lucasprabel/approx_lattice).

## Contents

---

<b>3.1. Introduction</b> .....	100
3.1.1 Presentation .....	100
3.1.2 Related Work .....	100
3.1.3 Contributions: A Technical Overview .....	102
<b>3.2. Approximate ISIS Problem</b> .....	102
3.2.1 Definitions .....	102
3.2.2 Hardness of AISIS .....	103
<b>3.3. Approximate Gadget Trapdoors</b> .....	104
3.3.1 Approximate Gadget Lattices .....	104
3.3.2 Approximate F-Sampling .....	105
3.3.3 Approximate Perturbation Sampling .....	106
3.3.4 Sampling Approximate Gaussian Preimages .....	106
<b>3.4. Application to an Identity-Based Encryption Scheme</b> .....	107
3.4.1 The Scheme .....	108
3.4.2 Correctness .....	108
3.4.3 Parameters and Security .....	109
3.4.3.1 Concrete Security .....	110
3.4.3.2 Asymptotic Security .....	110
3.4.4 Implementation .....	112
<b>3.5. Performance and Comparison</b> .....	113

## 3.1 Introduction

### 3.1.1 Presentation

Practical implementations of advanced lattice-based constructions have received much attention since the first practical scheme instantiated over NTRU lattices, proposed in [DLP14]). They are using powerful lattice-based building blocks which allow them to build Gaussian preimage sampling and trapdoor generation efficiently. In this chapter, we propose a construction and implementation of an identity-based encryption scheme using approximate variants of the "gadget-based" trapdoors introduced in [CGM19]. This construction will use techniques presented in Chapter 2 adapted to the approximate setting, will rely on the Module-LWE hardness assumption and will make use of the Micciancio-Peikert paradigm with approximate trapdoors. We will prove that our scheme is secure, and we will provide details on its implementation, several timings, and a comparison analysis to explain our results.

More precisely, we study the possibility of using approximate trapdoors, with SIS / LWE assumptions, rather than the exact trapdoors of [MP12], in order to obtain a more efficient scheme, without drastically reducing the intrinsic security. The use of approximate trapdoors leads to the appearance of error terms which must be taken care of in the decryption phase of the scheme. The resulting scheme is IND-sID-CPA secure in the standard model and resulted in an implementation showing that it is more efficient than the counterpart IBE using exact trapdoors that we presented in Section 2.5.

Therefore, this chapter provides different solutions in order to obtain schemes that can be used in practice, by reducing the size of their parameters (keys and ciphers) and the execution time of their different algorithms. We hope these new constructions and implementations will pave the way for more practical advanced trapdoor-based constructions.

### 3.1.2 Related Work

In [GPV08], Gentry, Peikert and Vaikuntanathan described the first lattice-based IBE, relying on the Dual-Regev encryption scheme. This scheme is presented in Section 1.4.2.1. An important contribution of their work was a sampling algorithm (known as GPV sampling) which showed how to use a short basis as a trapdoor for generating short lattice vectors. This sampler was then used to construct a lattice-based IBE scheme, proven

adaptively secure against chosen-plaintext attack in the random oracle model as defined in [BF01; Coc01]. However, the master public key and user secret keys had large sizes in  $O(n^2)$  bits. Later on, a construction of a Hierarchical IBE (HIBE) scheme in the standard model was proposed in [CHK<sup>+</sup>10] based on a new mechanism for users' keys delegation. This IBE scheme was proven secure in the selective model where the adversary needs to target an identity beforehand. In 2010, Agrawal et al. [ABB10a] proposed a Learning With Errors (LWE)-based IBE scheme with a trapdoor structure and with performance comparable to the GPV scheme. Their construction viewed an identity as a sequence of bits and then assigned a matrix to each bit. It used a sampling algorithm to obtain a basis with low Gram-Schmidt norm for the master secret key and formed a lattice family with two associated trapdoors to generate short vectors; one for all lattices in the family and the other one for all but one. The principle of their scheme was the basis of our own construction presented in Section 2.5.

The first Ring-LWE based IBE scheme has been proposed by Ducas, Lyubashevsky and Prest [DLP14] (DLP-IBE), which is still considered the most efficient scheme to date due to smaller key sizes. The use of the ring variant increased efficiency by reducing the public key size and ciphertext size to  $O(n)$ . The security of their scheme holds in the random oracle model and is related to the NTRU hardness assumption. An efficient C implementation of the DLP-IBE scheme and a detailed performance analysis was provided in [MSO17]. In 2017, Campbell and Grover introduced a HIBE scheme, called *Latte*, which can be viewed as a combination of the DLP-IBE scheme with the delegation mechanism from [CHK<sup>+</sup>10]. An optimized implementation and refined analysis of *Latte*, has recently been proposed in [ZMS<sup>+</sup>21].

The work from [BFR<sup>+</sup>18] constructed an IBE using the notion of gadget-based trapdoors in the ring setting, introduced by [MP12]. Such trapdoors can be seen as linear transformations mapping hard instances of cryptographic problems on some lattices to easy instances on a lattice defined by a public "gadget matrix". The IBE from [BFR<sup>+</sup>18] also made use of the efficient Gaussian preimage sampling algorithms from [GM18] to propose an implementation of their scheme. In Chapter 2, this IBE and its associated sampling algorithms were adapted to the module setting and instantiated. The use of module lattices of dimension  $nd$ , where  $d$  is the rank module, led to a more flexible choice of parameters. In [ZY22], the authors proposed new efficient gadget sampling algorithms which didn't need floating-point arithmetic, and as fast as the original [GM18] sampler.

### 3.1.3 Contributions: A Technical Overview

All those constructions generally make use of dedicated trapdoors, needed by the authority to generate the secret key of a user. In that case, building the trapdoor and sampling particular short vectors are quite costly, and represent the main bottleneck in the efficiency of such schemes.

Therefore, the main contribution presented in this chapter is to provide and implement a lattice-based IBE scheme which makes use of families of gadget-based approximate trapdoors. This construction relies on the **Module-LWE** problem. We investigate how to instantiate and parametrize the approximate trapdoor preimage sampling over these family of trapdoors in a way to obtain a provable and efficient quantum-safe IBE scheme. As in previous IBE constructions, encryption is based on the Dual-Regev encryption scheme. We provide a complete public and open-source C implementation<sup>1</sup> with performance benchmarking. The implementation is modular and makes it easy to change the building blocks of our algorithms according to the desired properties that we want to get.

Our work explores the use of approximate trapdoors for the construction of IBE schemes and the potential practical insights we can gain from it. We first adapted the construction presented in Chapter 2 using approximate trapdoors. The error induced by the approximate setting requires changes at several levels, either for the choice of encoding or for the sampling algorithms. As expected, we obtain better timings for all four algorithms composing our IBE by using approximate trapdoors rather than exact ones.

## 3.2 Approximate ISIS Problem

### 3.2.1 Definitions

As shown in [CGM19], the gadget trapdoor proposed by Micciancio and Peikert in [MP12] can be modified to an approximate trapdoor. Doing so allows to additionally decrease the sizes of different parameters involved in cryptographic schemes making use of trapdoors such as the public matrix, the trapdoor or the preimage, without affecting the scheme's concrete security.

We begin by defining the **ISIS** problem in Definition 29, which is a key aspect of the concept of approximate trapdoors. The definition is close to the one of the **ISIS** problem, but the equality is more relaxed, by involving an error term.

---

1. [https://github.com/lucasprabel/approx\\_lattice](https://github.com/lucasprabel/approx_lattice)

**Definition 29 (AISIS $_{n,d,m,q,\alpha,\beta}$ )** For any  $n, d, m, q \in \mathbb{N}$  and  $\alpha, \beta \in \mathbb{R}$ , given  $\mathbf{A} \in \mathcal{R}_q^{d \times m}$ ,  $\mathbf{y} \in \mathcal{R}_q^d$ , the Approximate Inhomogeneous Short Integer Solution problem AISIS $_{n,d,m,q,\alpha,\beta}$  asks to find a vector  $\mathbf{x} \in \mathcal{R}^m$  such that  $\|\mathbf{x}\| \leq \beta$  and there is a vector  $\mathbf{z} \in \mathcal{R}^d$  satisfying:  $\|\mathbf{z}\| \leq \alpha$  and  $\mathbf{Ax} = \mathbf{y} + \mathbf{z} \bmod q$ .

Then, approximate trapdoors allow to solve the AISIS problem in the same way exact trapdoors allowed us to solve the ISIS problem.

**Definition 30 (Approximate Trapdoor)** A string  $\tau$  is called an  $(\alpha, \beta)$ -approximate trapdoor for a matrix  $\mathbf{A} \in \mathcal{R}_q^{d \times m}$  if there is a probabilistic polynomial time algorithm that given  $\tau, \mathbf{A}$  and any  $\mathbf{y} \in \mathcal{R}_q^d$ , outputs a non-zero vector  $\mathbf{x} \in \mathcal{R}^m$  such that  $\|\mathbf{x}\| \leq \beta$  and there is a vector  $\mathbf{z} \in \mathcal{R}^d$  satisfying:  $\|\mathbf{z}\| \leq \alpha$  and  $\mathbf{Ax} = \mathbf{y} + \mathbf{z} \bmod q$ .

### 3.2.2 Hardness of AISIS

The following theorem shows that the AISIS problem is hard to solve when not knowing an approximate trapdoor for specific parameters. More precisely, the reduction in Theorem 12 establishes the hardness of solving the AISIS problem based on the hardness of solving decisional LWE with low-norm secret.

**Theorem 12 ([CGM19, Theorem 3.3])** For  $n, m, q \in \mathbb{Z}$ ,  $\alpha, \beta \in \mathbb{R}^+$  and  $\theta, \chi$  distributions over  $\mathbb{Z}$  such that  $q > 4(\|\theta\| \cdot (\alpha + 1) + \|\theta\|^n \cdot \alpha \cdot \sqrt{n} + \|\chi\|^m \cdot \beta \cdot \sqrt{m})$ . Then  $\text{LWE}_{n,m,q,\theta,\mathcal{U}(\mathbb{Z}_q),\chi} \leq \text{AISIS}_{n,m,q,\alpha,\beta}$ .

The AISIS problem is also proven as hard as the exact ISIS problem. The reduction proofs in [CGM19] make use of a series of reductions between the HNFs versions of the ISIS and AISIS problems.

**Theorem 13 ([CGM19, Theorem 3.4])** The following reductions hold:

- $\text{ISIS}_{n,m+m,q,\beta} \geq \text{AISIS}_{n,m,q,\alpha+\beta,\beta}$ ;
- $\text{AISIS}_{n,m,q,\alpha,\beta} \geq \text{ISIS}_{n,n+m,q,\alpha+\beta}$ .



### 3.3 Approximate Gadget Trapdoors

In this section, we define the approximate variant of the  $\mathbf{G}$ -lattice presented in Section 2.2. We also see how to adapt the trapdoor generation and the different sampling algorithms we had when working in the exact setting.

#### 3.3.1 Approximate Gadget Lattices

The core idea behind approximate trapdoors is to work with an approximate gadget matrix  $\mathbf{F}$  instead of working with the usual gadget matrix  $\mathbf{G}$  introduced in [MP12]:

$$\mathbf{G} = \mathbf{I}_d \otimes \mathbf{g}^T = \mathbf{I}_d \otimes [1 \quad b \quad b^2 \quad \dots \quad b^{k-1}] \in \mathcal{R}^{d \times dk}.$$

The simple proposition from [CGM19] is to modify this gadget matrix  $\mathbf{G}$  by dropping an arbitrary number  $\ell$  of the smaller entries from the gadget vector  $[1 \quad b \quad b^2 \quad \dots \quad b^{k-1}]$  to get an approximate gadget matrix  $\mathbf{F}$ , defined as:

$$\mathbf{F} = \mathbf{I}_d \otimes \mathbf{f}^T = \mathbf{I}_d \otimes [b^\ell \quad b^{\ell+1} \quad b^{\ell+2} \quad \dots \quad b^{k-1}] \in \mathcal{R}^{d \times d(k-\ell)}.$$

This allows to introduce the notion of approximate gadget trapdoor, in the same fashion we did in Definition 28 with the exact  $\mathbf{G}$ -lattice.

**Definition 31** An approximate trapdoor for a matrix  $\mathbf{A} \in \mathcal{R}_q^{d \times m}$  is a matrix  $\mathbf{T} \in \mathcal{R}^{(m-d(k-\ell)) \times d(k-\ell)}$  such that

$$\mathbf{A} \begin{bmatrix} \mathbf{T} \\ \mathbf{I}_{d(k-\ell)} \end{bmatrix} = \mathbf{H}\mathbf{F}$$

for some invertible  $\mathbf{H} \in \mathcal{R}_q^{d \times d}$ , called the *tag* of  $\mathbf{A}$ .

The differences between the exact and approximate settings are recalled in Figure 3.1.

Having defined the notion of approximate  $\mathbf{F}$ -trapdoor, we now describe the ApproxTrapGen algorithm, which outputs a matrix  $\mathbf{A}$  along with its associated approximate trapdoor  $\mathbf{T}$ , given a tag  $\mathbf{H}$ . We denote  $\omega = d(k - \ell)$ ,  $\bar{m} = d \log q$  and  $m = \bar{m} + \omega$ .

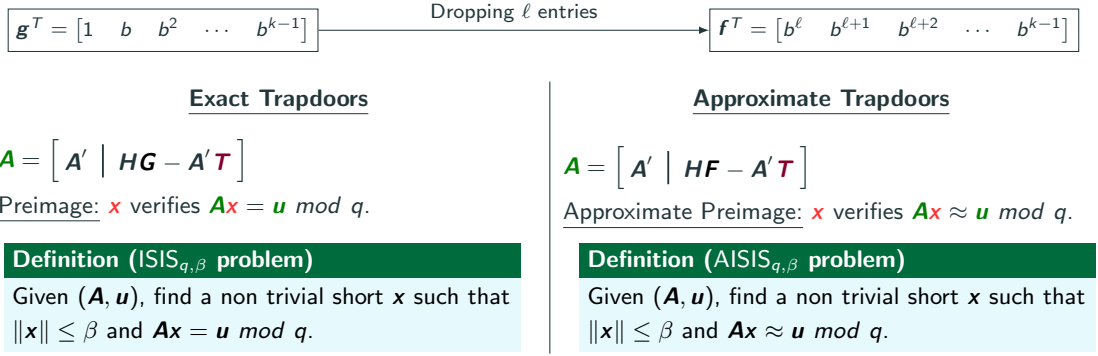


Figure 3.1 – The exact and approximate settings.

---

**Algorithm 7** ApproxTrapGen( $\mathbf{H}, \sigma$ )
 

---

- 1: Sample  $\bar{\mathbf{A}} \leftarrow \mathcal{U}(\mathcal{R}_q^{d \times \bar{m}})$ .
  - 2: Sample  $\mathbf{T} \leftarrow D_{\mathcal{R}^{\bar{m} \times \omega}, \sigma}$
  - 3: Set  $\mathbf{A} = [\bar{\mathbf{A}} \mid \mathbf{HF} - \bar{\mathbf{A}}\mathbf{T}] \in \mathcal{R}_q^{d \times m}$
  - 4: **return**  $(\mathbf{A}, \mathbf{T})$ .
- 

The correctness of the ApproxTrapGen algorithm is stated in Theorem 14.

### 3.3.2 Approximate F-Sampling

As when working with exact trapdoors, if we want to make use of approximate trapdoors in order to build cryptographic schemes, we must ensure beforehand to have efficient sampling algorithms available. As with the exact setting, sampling approximate Gaussian preimages will be done in 2 steps: the approximate  $\mathbf{F}$ -sampling and the perturbing sampling.

Here, we begin by describing the Approximate  $\mathbf{F}$ -sampling phase. Given a target vector  $\mathbf{v} \in \mathcal{R}^d$ , the goal of approximate  $\mathbf{F}$ -sampling is to compute a vector  $\mathbf{z} \in \mathcal{R}^{k-l}$  following a discrete Gaussian of parameter  $\alpha$  such that  $\mathbf{Fz} \approx \mathbf{v} \bmod q$ . We can perform approximate  $\mathbf{F}$ -sampling very easily using the previous exact  $\mathbf{G}$ -sampling algorithm described in Section 2.2.2:

**Algorithm 8** ApproxFSampling( $\mathbf{v}, \alpha$ )

- 
- 1: Sample  $\mathbf{x} \leftarrow \text{GSampling}(\mathbf{v}, \alpha)$
  - 2: Set  $\mathbf{z} = (x_{k-l}, \dots, x_{k-1})$
  - 3: **return**  $\mathbf{z}$
- 

**3.3.3 Approximate Perturbation Sampling**

The second phase of the approximate Gaussian preimage sampling is the perturbation sampling. We recall that perturbation sampling aims at sampling vectors following the Gaussian distribution over  $\mathcal{R}^m$  of covariance  $\Sigma_p = \zeta^2 \mathbf{I} - \alpha^2 \begin{bmatrix} \mathbf{T} \\ \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{T}^T & \mathbf{I} \end{bmatrix}$  so that when we sum the perturbation  $\mathbf{p}$  and  $\begin{bmatrix} \mathbf{T} \\ \mathbf{I} \end{bmatrix} \mathbf{z}$ , the final covariance matrix  $\Sigma_p + \alpha^2 \begin{bmatrix} \mathbf{T} \\ \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{T}^T & \mathbf{I} \end{bmatrix} = \zeta^2 \mathbf{I}$  does not leak any information about the trapdoor  $\mathbf{T}$ .

We can perform such a step using the exact version of the perturbation algorithm presented in Section 2.2.3.

**3.3.4 Sampling Approximate Gaussian Preimages**

We've previously described the approximate  $\mathbf{F}$ -sampling and the perturbation sampling algorithms. We can combine those two phases to get the ApproxSamplePre algorithm, described in Algorithm 9.

**Algorithm 9** ApproxSamplePre( $\mathbf{A}, \mathbf{R}, \mathbf{H}, \mathbf{u}, \zeta$ )

- 
- 1: Sample perturbation  $\mathbf{p} \leftarrow D_{\mathcal{R}^m, \sqrt{\Sigma_p}}$
  - 2: Set coset  $\mathbf{v} = \mathbf{H}^{-1}(\mathbf{u} - \mathbf{A}\mathbf{p})$
  - 3: Sample  $\mathbf{z} = (\mathbf{z}_1^T, \mathbf{z}_2^T)^T \leftarrow D_{\Lambda_q^v(\mathbf{G}), \sigma_g}$
  - 4: Set  $\mathbf{x} = \mathbf{p} + \begin{bmatrix} \mathbf{R} \\ \mathbf{I} \end{bmatrix} \mathbf{z}_2 \in \mathcal{R}^m$
  - 5: **return**  $\mathbf{x}$
- 

We state the correctness of the trapdoor generation algorithm ApproxTrapGen and of the approximate preimage sampling algorithm ApproxSamplePre in Theorem 14.

**Theorem 14 ([CGM19, Theorem 4.1])** There exists probabilistic, polynomial time algorithms  $\text{ApproxTrapGen}(\cdot)$  and  $\text{ApproxSamplePre}(\cdot)$  such that:

1.  $\text{ApproxTrapGen}(\mathbf{H}, \sigma)$  takes as input public parameters, a tag matrix  $\mathbf{H} \in \mathcal{R}^{d \times d}$  and parameter  $\sigma > 0$  and returns a matrix-approximate trapdoor pair  $(\mathbf{A}, \mathbf{R}) \in \mathcal{R}_q^{d \times m} \times \mathcal{R}^{\bar{m} \times \omega}$  where  $\mathbf{R}$  coefficients are drawn from a Gaussian distribution of parameter  $\sigma$  over  $\mathcal{R}$ .
2. Let  $\mathbf{A}$  be generated with an approximate trapdoor as above. The following two distributions are statistically indistinguishable:

$$\{(\mathbf{A}, \mathbf{x}, \mathbf{u}, \mathbf{y}) \mid \mathbf{u} \leftarrow \mathcal{U}(\mathcal{R}_q^d), \mathbf{x} \leftarrow \text{ApproxSamplePre}(\mathbf{A}, \mathbf{R}, \mathbf{0}, \mathbf{u}, \zeta), \mathbf{y} = \mathbf{u} - \mathbf{A}\mathbf{x} \bmod q\}$$

$$\text{and } \{(\mathbf{A}, \mathbf{x}, \mathbf{u}, \mathbf{y}) \mid \mathbf{x} \leftarrow D_{\mathcal{R}^m, \zeta}, \mathbf{y} \leftarrow D_{\mathcal{R}^d, \sigma \sqrt{(b^{2\ell}-1)/(b^2-1)}}, \mathbf{u} = \mathbf{A}\mathbf{x} + \mathbf{y} \bmod q\}$$

for any  $\sigma \geq \sqrt{b^2 + 1} \cdot \omega(\sqrt{\log d})$  and  $\zeta \gtrsim \sqrt{b^2 + 1} \frac{s_1^2(\mathbf{R})}{s_{2d}(\mathbf{R})} \eta_\epsilon(\mathbb{Z}^{dk})$ .

### 3.4 Application to an Identity-Based Encryption Scheme

In this section, we present our IBE scheme which makes use of approximate trapdoors. At a high level, the scheme will make use of the following blocks:

- The master secret key is a  $\mathbf{F}$ -approximate trapdoor  $\mathbf{R} \in \mathcal{R}^{\bar{m} \times \omega}$  associated to  $\mathbf{A}$  with tag  $\mathbf{0}$ , with subgaussian coefficients of parameter  $\sigma$ , with  $\omega = d(k - \ell)$ . The master public key is a tuple consisting of a uniformly random vector  $\mathbf{u} \in \mathcal{R}_q^d$  and the matrix  $\mathbf{A} \in \mathcal{R}_q^{d \times m}$ , with  $m = \bar{m} + \omega$  chosen as:  $\mathbf{A} = [\bar{\mathbf{A}} \mid -\bar{\mathbf{A}}\mathbf{R}]$ . Taking  $\bar{m} = d \log q$ , we get that  $\bar{\mathbf{A}}$  is full rank with high probability according to [BJR<sup>+</sup>22, Lemma 2.6]. Moreover, by taking  $\sigma > 4 \cdot 4^{\frac{1}{nd}} \sqrt{n}$ , we obtain that  $\mathbf{A}$  is statistically close to uniform by Corollary 7.5 from [LPR13].
- A "small-norm" FRD encoding  $H_M$  as described in Section 1.4.2.2. This allows anyone, with the knowledge of the master public key  $\mathbf{A}$ , to compute a public matrix  $\mathbf{A}_{\text{id}} = \mathbf{A} + [\mathbf{0}_{d, \bar{m}} \mid \mathbf{H}_{\text{id}}\mathbf{F}]$  associated to the identity  $\text{id}$  of a user. Then, the secret key for  $\text{id}$  is an approximate short vector  $\mathbf{x}_{\text{id}} \in \mathcal{R}^m$  obtained by using the  $\text{ApproxSamplePre}$  algorithm with the matrix  $\mathbf{A}_{\text{id}}$  as input. Such a vector satisfies the relation  $\mathbf{A}_{\text{id}}\mathbf{x}_{\text{id}} \approx \mathbf{u} \bmod q$ . We can bound the approximate error in this relation by using Theorem 14 and the fact that we use a small-norm FRD encoding.

- Finally, we use the Dual-Regev encryption scheme (described in Section 1.4.1.1) for the encryption and decryption algorithms, taking care of the additional error which appears when using approximate trapdoors.

### 3.4.1 The Scheme

We detail our IBE where users' keys are defined based on the approximate preimage sampling from Theorem 14 and encryption is based on the Dual-Regev scheme.

- $\text{Setup}(1^n) \rightarrow (\text{mpk}, \text{msk})$ :
  - $(\mathbf{A}, \mathbf{R}) \leftarrow \text{ApproxTrapGen1}(\mathbf{0}, \sigma) \in \mathcal{R}_q^{d \times m} \times \mathcal{R}^{\bar{m} \times w}$ ,  $\mathbf{u} \leftarrow \mathcal{U}(\mathcal{R}_q^d)$ ;
  - $\text{mpk} = (\mathbf{A}, \mathbf{u})$ ,  $\text{msk} = \mathbf{R}$ .
- $\text{Extract}(\text{mpk}, \text{msk}, \text{id}) \rightarrow \text{sk}_{\text{id}} \mathbf{x} \in \mathcal{R}^m$ :
  - $\mathbf{H}_{\text{id}} \leftarrow H_M(\text{id})$ ;  $\mathbf{A}_{\text{id}} \leftarrow \mathbf{A} + \begin{bmatrix} \mathbf{0}_{d, \bar{m}} & \mathbf{H}_{\text{id}} \mathbf{F} \end{bmatrix} \in \mathcal{R}_q^{d \times m}$ ;
  - $\mathbf{x} \leftarrow \text{ApproxSamplePre}(\mathbf{A}_{\text{id}}, \mathbf{R}, \mathbf{H}_{\text{id}}, \mathbf{u}, \zeta)$ ;
- $\text{Encrypt}(\text{mpk}, \text{id}, M) \rightarrow C = (\mathbf{b}, c) \in \mathcal{R}_q^{m+1}$ :
  - $\mathbf{H}_{\text{id}} \leftarrow H_M(\text{id})$ ;  $\mathbf{A}_{\text{id}} \leftarrow \mathbf{A} + \begin{bmatrix} \mathbf{0}_{d, \bar{m}} & \mathbf{H}_{\text{id}} \mathbf{F} \end{bmatrix} \in \mathcal{R}_q^{d \times m}$ ;
  - $\mathbf{s} \leftarrow D_{\mathcal{R}_q^d, \tau}$ ,  $\mathbf{e}_0 \leftarrow D_{\mathcal{R}^{m-w}, \tau}$ ,  $\mathbf{e}_1 \leftarrow D_{\mathcal{R}^w, \gamma}$ , and  $e' \leftarrow D_{\mathcal{R}, \tau}$ ;
  - $\mathbf{b} \leftarrow (\mathbf{s}^T \mathbf{A}_{\text{id}})^T + (\mathbf{e}_0^T \mid \mathbf{e}_1^T)^T$  and  $c \leftarrow \mathbf{s}^T \mathbf{u} + e' + \lfloor q/2 \rfloor M$ ;
- $\text{Decrypt}(\text{sk}_{\text{id}}, C) \rightarrow M$ :
  - set  $\mathbf{x} = \text{sk}_{\text{id}}$  and compute  $\text{res} \leftarrow c - \mathbf{b}^T \mathbf{x}$  which has integer coefficients;
  - for each  $i$ , if the coefficient  $\text{res}_i \in \mathbb{Z}$  is closer to  $\lfloor q/2 \rfloor$  than to 0, then  $M_i = 1$ , otherwise  $M_i = 0$ .

### 3.4.2 Correctness

To use approximate trapdoors with the Dual-Regev approach, we need to sample the LWE secret term with a small norm instead of sampling from the uniform distribution, in order to maintain the correctness of the schemes. Let's write  $\mathbf{y} \in \mathcal{R}_q^d$  the additional error we get by using approximate trapdoors instead of exact ones. The correctness of the decryption holds if the error term  $\|e' - (\mathbf{e}_0^T \mid \mathbf{e}_1^T) \mathbf{x} - \mathbf{s}^T \mathbf{y}\|$  is small enough, i.e. less than  $\lfloor q/4 \rfloor$ .

$$\begin{aligned}
 \text{res} &= c - \mathbf{b}^T \mathbf{x} \\
 &= \mathbf{u} \cdot \mathbf{s} + e' + \lfloor q/2 \rfloor M - \left[ (\mathbf{s}^T \mathbf{A}_{id})^T + (\mathbf{e}_0^T \mid \mathbf{e}_1^T)^T \right]^T \mathbf{x} \\
 &= \mathbf{u} \cdot \mathbf{s} + e' + \lfloor q/2 \rfloor M - \mathbf{s}^T \mathbf{A}_{id} \mathbf{x} - (\mathbf{e}_0^T \mid \mathbf{e}_1^T) \mathbf{x} \\
 &= \mathbf{u} \cdot \mathbf{s} + e' + \lfloor q/2 \rfloor M - \mathbf{s}^T (\mathbf{u} + \mathbf{y}) - (\mathbf{e}_0^T \mid \mathbf{e}_1^T) \mathbf{x} \\
 &= \underbrace{e' - (\mathbf{e}_0^T \mid \mathbf{e}_1^T) \mathbf{x} - \mathbf{s}^T \mathbf{y}}_{\text{error term}} + \lfloor q/2 \rfloor M \in \mathcal{R}.
 \end{aligned}$$

So we need to choose our parameters properly for the correctness of the Dual-Regev encryption to hold. We can bound as follows the Euclidean norms of the quantities that appear in the error term:

- $\|e'\| \leq t\tau\sqrt{n}$  from Lemma 1.
- $\|\mathbf{e}_0^T \mathbf{x}_0\| \leq 2t^2\tau\zeta nd$  from Lemma 1 and Theorem 14.
- $\|\mathbf{e}_1^T \mathbf{x}_1\| \leq t^2\gamma\zeta nd(k - \ell)$  from Lemma 1 and Theorem 14.
- $\|\mathbf{s}^T \mathbf{y}\| \leq t^2\tau n^{5/2} d\sigma_g \frac{q^{1/r}}{\sqrt{r}} \sqrt{(b^{2\ell} - 1)/(b^2 - 1)}$  from Lemma 1 and Theorem 14.

By substituting these bounds, we get the following constraints:

$$\begin{aligned}
 \|e' - (\mathbf{e}_0^T \mid \mathbf{e}_1^T) \mathbf{x} - \mathbf{s}^T \mathbf{y}\| &\leq \|e'\| + \|(\mathbf{e}_0^T \mid \mathbf{e}_1^T) \mathbf{x}\| + \|\mathbf{s}^T \mathbf{y}\| \\
 &\leq t\tau\sqrt{n} + t^2 nd \left[ \zeta (2\tau + \gamma(k - \ell)) + \tau n^{3/2} \sigma_g \frac{q^{1/r}}{\sqrt{r}} \sqrt{(b^{2\ell} - 1)/(b^2 - 1)} \right] \\
 &\leq \lfloor q/4 \rfloor.
 \end{aligned}$$

### 3.4.3 Parameters and Security

The following constraints, combined with the bounds of the norms above, must be met to ensure the correctness:

- The Gaussian parameter  $\sigma_g$  used for the  $\mathbf{G}$ -sampling in the ApproxSamplePre algorithm must verify  $\sigma_g \geq \sqrt{2b} \cdot (2b + 1) \cdot \sqrt{\log(2nw(1 + 1/\epsilon))/\pi}$  (see [GM18], Corollary 3.1).
- The Gaussian width for preimage sampling  $\zeta$  must follow the condition  $\zeta > \sqrt{(\sigma_g^2 + 1)s_1^2(\mathbf{R}) + \eta_\epsilon^2(\mathbb{Z}^{nm})}$ , knowing that  $s_1(\mathbf{R}) < 1.1\sigma(\sqrt{2nd} + \sqrt{nw} + 4.7)$  with high probability (see Chapter 2), where  $s_1(\mathbf{R})$  is the spectral norm of the trapdoor  $\mathbf{R}$ .

- The Gaussian width for approximate trapdoor generation  $\sigma$  must verify  $\sigma > 4 \cdot 4^{\frac{1}{nd}} \sqrt{n}$  to ensure the public matrix  $\mathbf{A}$  we use is statistically close to uniform (see [LPR13], Corollary 7.5).
- We choose to set the Gaussian parameter  $\gamma$  of the Gaussian error  $\mathbf{e}_1 \in \mathcal{R}^w$  as  $\gamma^2 = \sigma^2 \|\mathbf{e}_0\|^2 + 2nt^2 \sigma^2 \tau^2$ .

### 3.4.3.1 Concrete Security

To assess the concrete security of each parameter set of our scheme, we estimate the pseudorandomness of the public-key (corresponding to the **Module-LWE** security) and the hardness of breaking **ISIS**. The estimation of the **Module-LWE** security is done with the **Module-LWE** estimator from [APS15] with BKZ as the reduction model. We approximate our instances by an instance of an unstructured **LWE** problem in dimension  $nd$ . We follow a very pessimistic core-SVP hardness, where the cost of a BKZ algorithm with blocksize  $\kappa$  is taken to be the cost of only one call to an SVP oracle in dimension  $\kappa$ . For the **ISIS** problem, we follow the approach of [CGM19] which consists in computing the smallest block size achieving the target root hermite factor corresponding to forging a signature.

### 3.4.3.2 Asymptotic Security

We prove the asymptotic security of our scheme, which is stated in the following Theorem 15.

**Theorem 15** The IBE construction with parameters  $n, d, m, q, k, \ell, \sigma, \alpha, \zeta, \tau$  and  $\gamma$ , chosen as in the above description, is IND-sID-CPA secure in the standard model under the hardness of **Module-LWE** $_{n,d,q,\tau}$ .

*Proof.* The proof proceeds in a sequence of games  $G_i$  where the first game  $G_0$  is identical to the original IND-sID-CPA game. In the last game  $G_2$  in the sequence, the adversary has no information left about the initial message and hence has advantage zero. We then show that a PPT adversary cannot distinguish between the intermediate games, which will prove that the adversary has negligible advantage in winning the original one.

**Game  $G_0$ .** This is the initial IND-sID-CPA game between an adversary  $\mathcal{A}$  and an IND-sID-CPA challenger.

**Game  $G_1$ .** In this game, we change the way the challenger generates the public matrix  $\mathbf{A}$  by adding information about the identity  $\text{id}^*$  that  $\mathcal{A}$  targets for his attack. In Game 0,  $\mathbf{A}$  was generated thanks to  $\text{ApproxTrapGen1}(\mathbf{0}, \sigma)$ , together with the associated trapdoor  $\mathbf{R}$ . We had  $\mathbf{A} = [\bar{\mathbf{A}} \mid -\bar{\mathbf{A}}\mathbf{R}]$ . The public matrix  $\mathbf{A}$  is now generated in  $G_1$  thanks to the algorithm  $\text{ApproxTrapGen1}(-\mathbf{H}_{\text{id}^*}, \sigma)$ , so that we have  $\mathbf{A} = [\bar{\mathbf{A}} \mid -\mathbf{H}_{\text{id}^*}\mathbf{F} - \bar{\mathbf{A}}\mathbf{R}] \in \mathcal{R}_q^{d \times m}$ .

This matrix  $\mathbf{A}$  is statistically indistinguishable from a uniformly random matrix.

The challenger answers  $\mathcal{A}$ 's private key queries on identities  $\text{id} \neq \text{id}^*$  by calling  $\text{Extract}((\mathbf{A}, \mathbf{u}), \mathbf{R}, \text{id})$  and then using  $\text{ApproxSamplePre}(\mathbf{A}_{\text{id}}, \mathbf{R}, \mathbf{H}_{\text{id}}, \mathbf{u}, \zeta)$ . We have  $\mathbf{A}_{\text{id}} = [\bar{\mathbf{A}} \mid (\mathbf{H}_{\text{id}} - \mathbf{H}_{\text{id}^*})\mathbf{F} - \bar{\mathbf{A}}\mathbf{R}]$  and  $\mathbf{H}_{\text{id}} - \mathbf{H}_{\text{id}^*}$  is invertible because of the FRD construction, which allows the challenger to get a private key associated to the identity  $\text{id}$ .

**Game  $G_2$ .** This game is identical to  $G_1$  except that the challenge ciphertext  $\mathbf{C}^*$  is chosen as a uniformly random element in  $\mathcal{R}_q^{m+1}$  instead of being chosen by calling  $\text{Encrypt}((\mathbf{A}, \mathbf{u}), \text{id}^*, M_{b^*})$ .

*Reduction from Module-LWE.* We show that  $G_2$  and  $G_1$  are computationally indistinguishable for the adversary  $\mathcal{A}$  under the Module-LWE assumption.

Suppose  $\mathcal{A}$  has a non-negligible advantage in distinguishing  $G_2$  and  $G_1$ . We will use  $\mathcal{A}$  to construct a simulator  $\mathcal{B}$  who will be able to solve the Module-LWE problem with non-negligible advantage.

The simulator  $\mathcal{B}$  receives  $\bar{m} + 1$  samples  $(\mathbf{A}_i, b_i)_{0 \leq i \leq \bar{m}}$  with  $\mathbf{A}_i \in \mathcal{R}_q^d$  and  $b_i \in \mathcal{R}_q$  as an instance of the decisional Module-LWE problem. The simulator also receives the identity  $\text{id}^*$  targeted by the adversary  $\mathcal{A}$ . The simulator sets  $\mathbf{A}' = (\mathbf{A}_1, \dots, \mathbf{A}_{\bar{m}}) \in \mathcal{R}_q^{d \times \bar{m}}$  and  $\mathbf{b}' = (b_1, \dots, b_{\bar{m}}) \in \mathcal{R}_q^{\bar{m}}$  and then runs  $\text{ApproxTrapGen1}(-\mathbf{H}_{\text{id}^*}, \sigma)$  to get  $\mathbf{A} = [\mathbf{A}' \mid -\mathbf{H}_{\text{id}^*}\mathbf{F} - \mathbf{A}'\mathbf{R}]$  together with the trapdoor. Next,  $\mathcal{B}$  sets  $\mathbf{u} = \mathbf{A}_0 \in \mathcal{R}_q^d$  and he sends  $(\mathbf{A}, \mathbf{u})$  to  $\mathcal{A}$  as the master public key. Then,  $\mathcal{B}$  answers  $\mathcal{A}$ 's private key queries as in  $G_1$ .

Afterward, the adversary outputs two plaintexts  $M_0, M_1$  and sends them to  $\mathcal{B}$ . The simulator generates a random bit  $b^*$ , and the challenge ciphertext  $\mathbf{C}^* = (b^*, c^*)$  as follows:

$$\mathbf{b}^* = (\mathbf{b}'^T \mid -\mathbf{b}'^T \mathbf{R} + \hat{\mathbf{e}}^T)^T, \quad c^* = b_0 + \lfloor q/2 \rfloor M_{b^*},$$

where  $\hat{\mathbf{e}} \leftarrow D_{\mathcal{R}^w, \mu}$  for some real  $\mu$ , that is explicited below.

Samples from the LWE distribution. If the Module-LWE samples are drawn from the



Module-LWE distribution, we have:

$$\mathbf{b}'^T = \mathbf{s}^T \mathbf{A}' + \mathbf{e}'^T \text{ and } b_0 = \mathbf{s}^T \mathbf{A}_0 + e_0,$$

for some  $\mathbf{s} \in \mathcal{R}_q^d$ ,  $\mathbf{e}' \leftarrow D_{\mathcal{R}^m, \tau}$  and  $e_0 \leftarrow D_{\mathcal{R}, \tau}$ . Therefore, the first part of the ciphertext is  $\mathbf{b}^* = \mathbf{A}_{\text{id}^*}^T \mathbf{s} + (\mathbf{e}'^T | -\mathbf{e}'^T \mathbf{R} + \hat{\mathbf{e}}^T)^T$  and the second part is  $c^* = \mathbf{s}^T \mathbf{A}_0 + e_0 + \lfloor q/2 \rfloor M_{b^*}$ .

Then, for a fixed error  $\mathbf{e}'$ , the term  $-\mathbf{e}'^T \mathbf{R}$  is distributed as  $D_{\mathcal{R}^w, \sigma \|\mathbf{e}'\|}$  since  $\mathbb{E}[\mathbf{e}'^T \mathbf{R}] = \mathbf{e}'^T \mathbb{E}[\mathbf{R}] = \mathbf{0}$  and  $\text{cov}(\mathbf{e}'^T \mathbf{R}) = \sigma^2 \mathbf{e}'^T \mathbf{e}' \mathbf{I}_w = \sigma^2 \|\mathbf{e}'\|^2 \mathbf{I}_w$  by linearity of expectation and bilinearity of covariance. Moreover, the random variable  $\hat{\mathbf{e}}$  follows the Gaussian distribution  $D_{\mathcal{R}^w, \mu}$  and is independent from the random variable  $\mathbf{e}'^T \mathbf{R}$ . Therefore,  $-\mathbf{e}'^T \mathbf{R} + \hat{\mathbf{e}}$  is indistinguishable from a sample drawn from the distribution  $D_{\mathcal{R}^w, \gamma}$  for  $\mu$  satisfying  $\gamma^2 = (\sigma \|\mathbf{e}'\|)^2 + \mu^2$ .

Then, the challenge ciphertext  $(\mathbf{b}^*, c^*)$  follows the same distribution as in  $G_1$ .

Samples from the uniform distribution. If the Module-LWE samples are drawn from the uniform distribution, the ciphertext challenge also looks uniform as in  $G_2$ .

Finally, the adversary  $\mathcal{A}$  outputs a guess  $b$ . Because we supposed  $\mathcal{A}$  has a non-negligible advantage in distinguishing  $G_1$  and  $G_2$ , if  $b = b'$  with overwhelming probability, the simulator concludes that the challenged instance was drawn from the Module-LWE distribution. Otherwise,  $\mathcal{B}$  concludes that the Module-LWE instance was drawn from the uniform distribution.  $\square$

### 3.4.4 Implementation

Our IBE proof relies on a statistical trapdoor instantiation. Although the size of the parameters increases consequently, the use of approximate trapdoors allowed us to mitigate this loss in efficiency induced by the use of a statistical instantiation. In order to ensure decryption correctness, we also need to use a large modulus. This leads us to perform calculations carefully on 64-bit integers, so as not to affect our scheme efficiency. The IBE also makes use of a small-norm encoding, instead of the low-degree encoding used in Chapter 2, to ensure that the noise is still not too large in order to decrypt. The encoding we use sets constraints on the structure of the ring  $\mathcal{R}_q$  which is not compatible with the NTT for polynomial multiplications. Instead, we use a "partial NTT" based on [KLS18] results, which reduces multiplication in  $\mathcal{R}_q$  to multiplication in smaller rings. Finally, the sampling algorithms have been adapted to the approximate setting.

The scheme is implemented in C, inheriting the modularity of the implementation presented in Chapter 2. It relies on several basic blocks that can be swapped out: the arithmetic over  $\mathbb{Z}_q$  and  $\mathcal{R}_q$ , a pseudorandom number generator, and a (constant-time) sampler of discrete Gaussian distributions over  $\mathbb{Z}$ . To generate our specific discrete Gaussian distributions, we make use of the following building blocks: an AES-based pseudorandom number generator (implemented using AES-NI instructions for x86 architectures), and a sampler of discrete Gaussians over  $\mathbb{Z}$  similar to Karney’s sampler [Kar16]. We chose this sampler as it can generate samples in constant time, independently of the center, Gaussian parameter, and output value. All the computations that deal with non-integers are carried out with floating-point operations that do not involve subnormal numbers. We rely on results from [KLS18, Lemma 3] to reduce multiplications in  $\mathcal{R}_q$  to polynomials multiplications in rings of the form  $\mathbb{Z}_q[X]/\langle X^n + 1 \rangle$ . The CRT reduction we used then allowed us to speed up polynomial arithmetic in  $\mathcal{R}_q$ .

## 3.5 Performance and Comparison

In Table 3.1, we show some applicable parameter sets together with their concrete bit security using the Module-LWE estimator from [APS15] with BKZ as a reduction cost model. All the algorithms comprising the IBE over module lattices are more efficient than their exact counterpart at the same security level. This improvement concerns in particular Setup and Extract which are optimized by a factor  $\approx 1.5$ .

As expected, the 4 algorithms are more efficient for low values of  $d$ . Concerning the Decrypt algorithm, its execution time relies mostly on the value of  $n$  rather than  $d$ . Our timings have been obtained on an Intel i7-8650U CPU running at 4.2 GHz. We provide concrete parameter sets and the associated concrete results in Table 3.1.

$\lceil \log_b q \rceil$	$(n, d)$	$\ell$	$\zeta$	Setup	Extract	Encrypt	Decrypt	M – $\text{LWE}_{n,d,q,\tau}$
58	(256, 4)	0	1137729	203.52	56.88	17.69	2.72	81
58	(256, 4)	8	1068989	174.94	49.82	15.00	2.40	80
58	(256, 4)	15	1004226	152.70	41.57	12.79	2.19	78
60	(512, 3)	0	1398812	210.85	67.28	19.20	4.06	110
60	(512, 3)	8	1315427	189.35	59.33	17.13	3.80	109
60	(512, 3)	15	1236981	160.33	53.39	14.00	3.22	107

Table 3.1 – Proposed IBE parameters for our IBE with different pairs of polynomial ring dimension  $n$  and rank  $d$ , for different modulus sizes and taking  $\sigma_g = 54.9$ ,  $\sigma = 64.1$ .  $\sigma_g$  is the Gaussian parameter for the  $\mathbf{G}$ -sampling,  $\sigma$  is the Gaussian width of the trapdoor  $\mathbf{R}$  used in the Setup algorithm and  $\zeta$  is the standard deviation for the Gaussian preimage sampling in the Extract algorithm. Timings in columns Setup-to-Encrypt are given in ms.

We also compare our IBE performance with the IBE presented in Chapter 2, which corresponds to the case  $\ell = 0$ , that is to say the use of exact trapdoors instead of approximate ones. We observe that for a fixed pair  $(n, d)$ , the larger  $\ell$  is, the better timings are. Overall, the use of approximate trapdoors allows to obtain better timings for all the algorithms comprising the IBE scheme.

# IMPROVING EFFICIENCY USING NTRU LATTICES

---

This chapter’s content is mainly based on a joint work [IPR23] with Malika Izabachène and Adeline Roux-Langlois. The corresponding implementation is available at [https://github.com/lucasprabel/approx\\_lattice](https://github.com/lucasprabel/approx_lattice).

## Contents

---

<b>4.1. Introduction</b> .....	116
4.1.1 Presentation .....	116
<b>4.2. The DLP-IBE Scheme</b> .....	117
4.2.1 The Scheme .....	117
4.2.2 Parameters and Security .....	120
<b>4.3. Solving the NTRU Equation</b> .....	121
4.3.1 HNF Method .....	122
4.3.2 Resultant Method .....	123
4.3.3 Using the Field Norm for Power-of-Two Cyclotomic Fields .....	125
4.3.4 Tools for More General Fields .....	127
<b>4.4. A New Assumption: iNTRU</b> .....	128
4.4.1 Definition .....	129
4.4.2 Hardness of iNTRU .....	129
4.4.3 Approximate Trapdoors Based on iNTRU .....	130
<b>4.5. Application to a NTRU-Based IBE Scheme</b> .....	131
4.5.1 The Scheme .....	132
4.5.2 Correctness .....	132
4.5.3 Parameters and Security .....	133
4.5.4 Cryptanalysis in the Overstretched Regime .....	136
4.5.5 Implementation and Performance .....	144

## 4.1 Introduction

### 4.1.1 Presentation

As we have seen in Chapter 2 and Chapter 3, schemes based on structured lattices greatly benefit from the underlying polynomial structure, to the point that they can become faster than some widely deployed cryptosystems based on classical assumptions and can achieve reasonably small parameter sizes. But with the future deployment of post-quantum cryptography getting closer, it is important to explore new lattice-based cryptosystems which would help to further improve the efficiency of those schemes.

This chapter deals with the study of NTRU lattices and the efficiency (in terms of parameter sizes and running time), implementation perspectives and concrete security of cryptographic schemes based on the NTRU hardness assumption.

My work starts with a variant of NTRU, called iNTRU, which was introduced in [GGH<sup>+</sup>19]. This assumption can be seen as an inhomogeneous variant of NTRU: basically, the matrix version of iNTRU asks to distinguish between  $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$  defined as  $\mathbf{A} = \mathbf{S}^{-1}(\mathbf{G} - \mathbf{E}) \bmod q$  (where  $\mathbf{S} \in \mathbb{Z}_q^{n \times n}$  is a random invertible matrix,  $\mathbf{E}$  is a low-norm matrix and  $\mathbf{G}$  is the gadget matrix) from a matrix chosen uniformly at random in  $\mathbb{Z}_q^{n \times m}$ . Because the use of gadget trapdoors is not directly compatible with the NTRU hardness assumption, I used that particular variant of NTRU in order to build an identity-based encryption scheme based on the [MP12] paradigm. This allows to obtain a more efficient IBE than the one presented in Chapter 3, in terms of timings and parameter sizes. This new scheme also resulted in an implementation, making it possible to assess its timings performance and to compare it with the previous IBE presented in this manuscript.

Afterward, I worked on analysing the concrete security of cryptographic schemes based on this iNTRU hardness assumption. Because this new problem has been introduced recently, the literature is still scarce on this subject, hence the importance of carrying out detailed analyses of its practical security. I was more particularly interested in overstretched attacks. Those attacks were introduced in [KF17] in the case of NTRU lattices, and applied to cryptosystems that need a large modulus  $q$  compared to the dimension  $2n$  of the lattice underlying the scheme. Thus, in this chapter, I analyze the possibility of extending these overstretched attacks to iNTRU, based on recent results from [DvW21], the later article dealing with NTRU instances.

In addition, I present a work concerning the resolutions of NTRU equations, which are involved in particular in several cryptosystems such as Falcon [FHK<sup>+</sup>17] or BAT [FKP<sup>+</sup>22], during the key generation phase. The objective is to investigate techniques for effectively solving NTRU equations, when working with number fields that don't admit a subfields tower. In this case, solving these equations efficiently as presented in [PP19] using the field norm and the subfield structure is no longer possible, and other solutions must be found. Another goal is also to avoid floating point arithmetic during the reduce phase of the NTRU equation solving algorithms, in order to get an efficient implementation.

## 4.2 The DLP-IBE Scheme

While the first signature schemes based on NTRU lattices were proven to be insecure, Ducas, Lyubashevsky and Prest [DLP14] brought those specific lattices in the [GPV08] framework, thus constructing proven secure schemes: a signature and an identity-based encryption scheme. By using the structure of NTRU lattices, [DLP14] has shown that GPV-based schemes can be made efficient and practical for lattice-based cryptography.

Their signature would later constitute an important basis for Falcon [FHK<sup>+</sup>17], selected by the NIST in 2022 as a post-quantum cryptography standard for digital signatures.

### 4.2.1 The Scheme

In the [DLP14] scheme, operations take place over the polynomial rings  $\mathcal{R} = \mathbb{Z}[X]/(X^N + 1)$  and  $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^N + 1)$ . Here,  $N$  is a power of 2 and  $q$  is a prime congruent, verifying  $q = 1 \pmod{2N}$ . This latter equality implies that the polynomial  $x^N + 1$  splits into  $N$  factors, thus allowing efficient computations over the ring  $\mathcal{R}_q$ .

As explained above, the security underlying the DLP-IBE scheme comes from the NTRU as well as the Ring-LWE hardness assumptions (see Section 1.2.3.2). We give below a full description of the four algorithms comprising the DLP-IBE [DLP14] scheme.

The Setup algorithm outputs the master public key, which is a polynomial  $h \in \mathcal{R}_q$  and the master secret key  $\mathbf{B}$ , which can be seen as a "good" basis of the  $2N$ -dimensional public NTRU lattice.

---

**Algorithm 10** Setup( $N, q$ )
 

---

- 1: Let  $\sigma_f = 1.17\sqrt{\frac{q}{2N}}$
  - 2: Sample  $f, g \leftarrow D_{N, \sigma_f}$
  - 3: Let  $Norm = \max\left(\|(g, -f)\|, \left\|\frac{g\bar{f}}{f\bar{f}+g\bar{g}}, \frac{g\bar{g}}{f\bar{f}+g\bar{g}}\right\|\right)$ . If  $Norm > 1.17\sqrt{q}$ , go back to Step 2
  - 4: Compute  $\rho_f, \rho_g \in \mathcal{R}$  and  $R_f, R_g \in \mathbb{Z}$  such that  $\rho_f \cdot f = R_f \pmod{x^N + 1}$  and  $\rho_g \cdot g = R_g \pmod{x^N + 1}$ . If  $\gcd(R_f, R_g) \neq 1$  or  $\gcd(R_f, q) \neq 1$ , go to Step 2
  - 5: Compute  $u, v \in \mathbb{Z}$  such that  $u \cdot R_f + v \cdot R_g = 1$
  - 6: Set  $F = -qv\rho_g$  and  $G = -qu\rho_f$
  - 7: Let  $k = \lfloor \frac{F*\bar{f}+G*\bar{g}}{f*\bar{f}+g*\bar{g}} \rfloor \in \mathcal{R}$
  - 8: Let  $F \leftarrow F - k * f$  and  $G \leftarrow G - k * g$
  - 9: Set  $h = g * f^{-1} \pmod{q}$  and  $\mathbf{B} = \begin{bmatrix} g & -f \\ G & -F \end{bmatrix}$
  - 10: **return**  $mpk = h \in \mathcal{R}_q$  and  $msk = \mathbf{B}$
- 

The standard deviation  $\sigma_f$  of the Gaussian distribution from which the polynomials  $f$  and  $g$  are sampled is chosen to ensure that  $E[\|\mathbf{b}_1\|] = 1.17\sqrt{q}$ , which is the bound computed from heuristics in [DLP14] for how short the standard deviation can be set using the Klein sampler (described in Section 1.3.2.5). In Step 3 of the algorithm, a bound on the Gram-Schmidt norm of the matrix  $\mathbf{B}$  is computed, and the algorithm returns to Step 2 if it is not small enough. Indeed, the Gram-Schmidt norm of the master secret key  $\mathbf{B}$  needs to be small because it has an impact on the size of the user's secret keys computed by the Extract algorithm, thus impacting the security and the efficiency of the scheme. Then, the algorithm uses an extended Euclidean algorithm to compute  $F, G$  such that  $f * G - g * F = q$ . We give more detailed explanations about this step in Section 4.3.2. Then,  $F$  and  $G$  are reduced by computing  $F - k * f$  and  $G - k * g$  where  $k \in \mathcal{R}$  is defined as  $k = \lfloor \frac{F*\bar{f}+G*\bar{g}}{f*\bar{f}+g*\bar{g}} \rfloor$ , where  $\bar{f}$  (resp.  $\bar{g}$ ) is the adjoint of  $f$  (resp.  $g$ ) defined in Definition 4.2.1.

**Definition 32** Let  $f \in \mathbb{C}[X]/(\phi)$  where  $\phi \in \mathbb{Q}[X]$  is a monic polynomial with distinct complex roots  $(z_j)_j \in \mathbb{C}$ . The adjoint of  $f$ , denoted  $\bar{f}$ , is the unique polynomial  $\bar{f} \in \mathbb{C}[X]/(\phi)$  such that for each  $z_j$ , we have:

$$\bar{f}(z_j) = \overline{f(z_j)}.$$

This step simply consists in a straightforward generalization of the nearest plane algorithm from Babai, and allows to compute a solution  $(F, G)$  with small norm of the NTRU

equation:

$$f * G - g * F = q.$$

Finally, the algorithm outputs  $h = g * f^{-1} \pmod q$  as the master public key and the basis  $\mathbf{B}$  as the master secret key.

For the Extract algorithm, the public key associated to an identity  $\text{id}$  is  $H(\text{id})$ , where  $H$  is a cryptographic hash function mapping identities to  $\mathbb{Z}_q[x]/(x^N + 1)$ . Then, a secret key associated to the identity  $\text{id}$  is computed using the GPV preimage Gaussian sampling algorithm KleinSampler, and consists of a polynomial  $s_2$  verifying  $s_2 h + s_1 = H(\text{id})$ , where  $s_1$  is another polynomial with small coefficients. The KleinSampler algorithm has been defined in Algorithm 2.

---

**Algorithm 11** Extract( $\mathbf{B}, H, \text{id}$ )

---

- 1: Set  $t = H(\text{id})$
  - 2: Set  $(s_1, s_2) = (t, 0) - \text{KleinSampler}(\mathbf{B}, (t, 0), \sigma)$
  - 3: **return**  $sk_{\text{id}} = s_2$
- 

Finally, the Encrypt and Decrypt algorithms are based on the classical Dual-Regev encryption scheme [LPR13], already mentioned multiple times in this manuscript. Basically, a user public key consists of  $h$  and  $t = H(\text{id})$  and it is used to encrypt a message  $m \in \mathcal{R}$  with binary coefficients, by computing  $u = r * h + e_1$ ,  $v = r * t + e_2 + \lfloor q/2 \rfloor \cdot k$  and  $c = m \oplus H'(k)$  where the hash  $H'(k)$  of a uniformly sampled element  $k$  is used as a one-time pad with the message  $m$ .

---

**Algorithm 12** Encrypt( $h, \text{id}, m, H, H'$ )

---

- 1: Sample  $r, e_1, e_2 \leftarrow \{-1, 0, 1\}^N$  and  $k \leftarrow \{0, 1\}^N$
  - 2: Set  $t = H(\text{id})$
  - 3: Set  $u = r * h + e_1 \in \mathcal{R}_q$
  - 4: Set  $v = r * t + e_2 + \lfloor q/2 \rfloor \cdot k \in \mathcal{R}_q$
  - 5: Let  $v \leftarrow 2^l \lfloor v/2^l \rfloor$
  - 6: **return**  $(u, v, c) = (u, v, m \oplus H'(k))$
- 

Finally, with a proper choice of parameters, the receiver can decrypt the ciphertext  $(u, v, c)$  using his secret key  $sk_{\text{id}} = s_2$ , which allows him to recover the key  $k$  by rounding.



**Algorithm 13** Decrypt( $sk_{id}, (u, v, c)$ )

- 
- 1: Set  $w = v - u * s_2$
  - 2: Set  $k = \lfloor \frac{w}{q/2} \rfloor$
  - 3: **return**  $c \oplus H'(k)$
- 

## 4.2.2 Parameters and Security

For the sake of comparison with the IBE based on a family of NTRU lattices that will be presented in Section 4.5, the original suggestion of parameter sets by the authors of [DLP14] are given in Table 4.1.

Table 4.1 – Sets of parameters proposed in [DLP14; MSO17] for the DLP-IBE scheme.

Parameter Set	$(N, \lceil \log_2(q) \rceil)$	Security Level	Root Hermite Factor
0	(512, 23)	1.0075	80
1	(512, 24)	1.0079	80
2	(512, 26)	1.0085	< 80
3	(1024, 18)	1.0038	192
4	(1024, 20)	1.0039	192
5	(1024, 22)	1.0043	< 192

The concrete security level of the parameter sets proposed in [DLP14; MSO17] is estimated using the root Hermit factor [GN08; DDL<sup>+</sup>13]. This quantity, denoted  $\delta$ , allows to measure the hardness of a given lattice problem. In [DLP14], the authors use that  $\delta \approx 1.007$  gives an estimation of 80 bits of security and for  $\delta < 1.004$ , 192 bits of security is guaranteed.

For the problem of finding a vector  $\mathbf{v}$  in a lattice  $\Lambda$  of dimension  $n$  and whose norm is larger than the  $n^{\text{th}}$  root of  $\det(\Lambda)$ , the root Hermite factor is defined as  $\delta^n = \frac{\|\mathbf{v}\|}{\det(\Lambda)^{1/n}}$ .

In [DDL<sup>+</sup>13], the authors made some experiments that lead them to estimate the value of the root Hermite factor for the problem of finding an unusually-short vector in a NTRU lattice as  $\delta^n = \frac{\sqrt{n/(2\pi e)} \det(\Lambda)^{1/n}}{0.4\|\mathbf{v}\|}$ .

The security analysis carried out in [DLP14] then used those equalities to estimate the concrete security of their DLP-IBE scheme. They distinguish 3 different vulnerable parts in their IBE, for which they compute the value of the associated root hermite factor.

The first one involves the master public key  $h$  of the scheme. Given the description of the Setup algorithm,  $h$  is generated as a NTRU polynomial  $h = g * f^{-1}$ . The best attacks to distinguish such a polynomial from a random one then consists in finding 2 short polynomials  $f, g$  such that  $h * f - g = 0 \pmod q$ . This is equivalent to finding an unusually short vector  $(f, g)$  in a NTRU lattice of dimension  $2N$  and determinant  $q^N$ . Given the above expression of the root Hermite factor for such a problem, we thus have  $\delta = (\sqrt{N}/1.368)^{\frac{1}{2N}}$ .

The second one involves the user secret keys  $sk_{id}$ . A secret key vector  $(s_1, s_2)$  has a norm of  $\sigma\sqrt{2N}$  where  $\sigma = 1.17\eta_\epsilon(\mathbb{Z}) \cdot \|\tilde{\mathbf{B}}\|$ , and thus, given the above expression of the root Hermite factor, we find  $\delta^{2N} = \frac{\sigma\sqrt{2N}}{\sqrt{q}}$ .

Finally, the last one concerns the encryption part of the scheme. In order to break the CPA-security of the scheme, the best attacks consists in recovering the errors  $e_1$  and  $e_2$  sampled in the Encrypt algorithm. Knowing a ciphertext  $(u, v, c)$ , we then have  $(t * h^{-1}) * e_1 - e_2 = (t * h^{-1}) * u - v \pmod q$ , which is equivalent to the problem of finding a vector  $(e_1, e_2, 1)$  of dimension  $2N + 1$  in a NTRU lattice of dimension  $2N + 1$  with determinant  $q^N$ . Again, using the above expression for the root Hermite factor gives  $\delta = (0.74\sqrt{q})^{\frac{1}{2N}}$ .

After having considered these 3 different cases and pointing out the underlying lattice problems, the computations of the root Hermite factor for each case highlights the fact that the attack against the CPA-security of the scheme is the most vulnerable part. Therefore, the security level displayed in Table 4.1 comes from the computations of the root Hermite factor  $\delta = (0.74\sqrt{q})^{\frac{1}{2N}}$ .

### 4.3 Solving the NTRU Equation

In the DLP-IBE scheme presented in Section 4.2, the key generation is the most costly component, mainly due to the need to solve the NTRU equation, which consists in finding a small  $(F, G)$  satisfying

$$fG - gF = q \pmod{x^N + 1}.$$

Therefore, in this section, we focus on the resolution of NTRU equations, which are involved in particular in several cryptosystems such as the DLP-IBE scheme as we've seen, but also Falcon [FHK<sup>+</sup>17] or BAT [FKP<sup>+</sup>22]. Resolving NTRU equation is generally done during the key generation phase of those schemes. This phase aims to generate short

polynomials  $f$  and  $g$  and then solve the NTRU equation to obtain  $F$  and  $G$ , which define the trapdoor basis  $\mathbf{B}_{f,g} = \begin{bmatrix} g & -f \\ G & -F \end{bmatrix}$ .

Solving this kind of equation is the main bottleneck involved in the NTRU key generation algorithm. This essentially comes from the large coefficients of the intermediate  $F$  and  $G$ . However, many efforts have been made those last few years toward accelerating this step of the key generation phase ([FHK<sup>+</sup>17; PP19]).

Another interesting question regarding this problem is to investigate techniques for effectively solving NTRU equations when working with number fields that don't admit a subfields tower. In this case, using a solver such as the one presented in [PP19], which uses the field norm and the subfield structure is no longer possible, and other solutions must be found.

Let's start by presenting classical and more recent methods which aim at efficiently solving NTRU equations.

### 4.3.1 HNF Method

First, we recall a method, exposed in particular in the Extended Euclid algorithm in Dedekind Domains from [Coh93] by Henri Cohen and used for example in the revised NTRUSign key generation algorithm of [SS13].

We denote  $\mathcal{K}$  a number field of degree  $d = [\mathcal{K} : \mathbb{Q}]$  and  $\mathcal{O}_{\mathcal{K}}$  its ring of integers.

**Algorithm 14** CompleteBasis( $\mathcal{M}, (\mathbf{c}_1, d_1)$ )**Input:** A rank 2 module  $\mathcal{M} = \mathbf{a}_1 b_1 \oplus \mathbf{a}_2 b_2$  and a submodule  $(\mathbf{c}_1, d_1)$ .**Output:** A module  $(\mathbf{c}_2, d_2) \subset \mathcal{M}$  such that  $\mathbf{c}_1 d_1 \oplus \mathbf{c}_2 d_2 = \mathcal{M}$ 

- 1: Let  $B = [b_1, b_2]$  and  $(u, v) \leftarrow B^{-1} d_1$
- 2: **if**  $u = 0$  **then**  $x = 0$  and  $y = 1/v$
- 3: **if**  $v = 0$  **then**  $x = 1/u$  and  $y = 0$
- 4:  $\mathbf{I} \leftarrow$  a  $\mathbb{Z}$ -basis of  $u \mathbf{a}_1^{-1} \mathbf{c}_1$
- 5:  $\mathbf{J} \leftarrow$  a  $\mathbb{Z}$ -basis of  $v \mathbf{a}_2^{-1} \mathbf{c}_1$
- 6:  $\mathbf{C} \leftarrow (\mathbf{I} \mid \mathbf{J})$  the concatenation of the matrices  $\mathbf{I}$  and  $\mathbf{J}$
- 7: Compute the HNF decomposition of  $\mathbf{C}$ :  $\mathbf{CZ} = (\mathbf{I}_d \mid \mathbf{0})$ . Let's denote  $(\mathbf{z}_I^T, \mathbf{z}_J^T)$  the first column of  $\mathbf{Z}$
- 8:  $x' \leftarrow$  the  $\mathcal{O}_K$  element corresponding to  $\mathbf{Jz}_J$
- 9:  $x \leftarrow x'/v$  and  $y \leftarrow -(1 - x')/u$
- 10: **return**  $(\mathbf{a}_1 \mathbf{a}_2 \mathbf{c}_1^{-1}, x b_1 + y b_2)$

This method doesn't exploit the presence of a tower of subfields, and uses the computation of Hermite Normal Form (HNF) in step 7, which makes it not so efficient, having to deal with big integers. Indeed, this algorithm has quadratic space and quasi-cubic time complexities which represents a significant drawback for the efficiency of the key generation algorithm.

### 4.3.2 Resultant Method

Another approach makes use of the computations of resultants. We give the definition of the resultant of two complex polynomials in Definition 33.

**Definition 33 (Resultant)** Let  $f, g \in \mathbb{C}[X]$  be two polynomials of degree  $n$  and  $m$ . We assume that  $f$  (resp.  $g$ ) can be written as  $f(x) = \sum_{i=0}^n f_i x^i = f_n \prod_{i=0}^{n-1} (x - \alpha_i)$  (resp. as  $g(x) = \sum_{i=0}^m g_i x^i = g_m \prod_{i=0}^{m-1} (x - \beta_i)$ ). Then, the *resultant* of  $f$  and  $g$  is defined as:

$$\text{Res}(f, g) = f_n^m g_m^n \prod_{i,j} (\alpha_i - \beta_j) = f_n^m \prod_i g(\alpha_i) = (-1)^{nm} g_m^n \prod_i f(\beta_i).$$

We give a brief description of the method below, presented in [PP19]. It is used in different cryptographic schemes such as NTRUSign [HHP<sup>+</sup>03], the original Latte scheme

[CG17] or DLP-IBE [DLP14] that we've presented in Section 4.2.

In the following algorithm, we work in the ring  $\mathcal{R} = \mathbb{Z}[X]/(\phi)$  where  $\phi$  can be any monic irreducible polynomial over  $\mathbb{Q}[X]$ .

---

**Algorithm 15** NTRUSolveResultant( $f, g$ )

---

**Input:**  $f, g \in \mathbb{Z}[X]/(\phi)$

**Output:**  $F, G \in \mathbb{Z}[X]/(\phi)$  such that  $fG - gF = q$

- 1: Compute  $R_f \in \mathbb{Z}$  and  $s \in \mathbb{Z}[X]$  such that  $sf = R_f \pmod{\phi}$
  - 2: Compute  $R_g \in \mathbb{Z}$  and  $t \in \mathbb{Z}[X]$  such that  $tg = R_g \pmod{\phi}$
  - 3: Compute the Bézout coefficients  $u, v \in \mathbb{Z}$  such that  $uR_f + vR_g = \delta$  where  $\delta = \gcd(R_f, R_g)$
  - 4: **if**  $\delta$  doesn't divide  $q$  **then** abort
  - 5:  $(F, G) \leftarrow (-(vq/\delta)t, (uq/\delta)s)$
  - 6: Reduce( $f, g, F, G$ )
  - 7: **return**  $(F, G)$
- 

Steps 1 and 2 make use of the extended Euclidean algorithm to compute the Bézout coefficients  $s, s', t, t' \in \mathbb{Z}[X]$  and  $R_f, R_g \in \mathbb{Z}$  such that  $sf + s'\phi = R_f$  and  $tg + t'\phi = R_g$ , but  $s'$  and  $t'$  don't need to be stored. Then, in step 3, the gcd of  $R_f$  and  $R_g$  is computed, with its Bézout coefficients  $u, v \in \mathbb{Z}$ . In step 5, if  $\delta$  is a divisor of  $q$ , it then provides a solution to the NTRU equation.

However, while  $F$  and  $G$  verify  $fG - gF = q \pmod{\phi}$  after step 5, the sizes of their coefficients are too big and need to be reduced with respect to  $(f, g)$  for efficiency purposes, hence the call to the Reduce subroutine in step 6. The Reduce algorithm was also used during the Setup algorithm of the DLP-IBE scheme presented in Section 4.2. We recall this subroutine in Algorithm 16. This simply consists in a generalization of the nearest plane algorithm by Babai [Bab86].

---

**Algorithm 16** Reduce( $f, g, F, G$ )

---

**Input:**  $f, g, F, G \in \mathbb{Z}[X]/(\phi)$ **Output:**  $\tilde{F}, \tilde{G} \in \mathbb{Z}[X]/(\phi)$  such that  $f\tilde{G} - g\tilde{F} = fG - gF \pmod{\phi}$ 

- 1: Let  $(\tilde{F}, \tilde{G}) = (F, G)$ .
  - 2: **while**  $k \neq 0$  **do**
  - 3:     Compute  $k = \lfloor \frac{F*\tilde{f}+G*\tilde{g}}{f*\tilde{f}+g*\tilde{g}} \rfloor$
  - 4:     Let  $(\tilde{F}, \tilde{G}) = (\tilde{F} - kf, \tilde{G} - kg)$
  - 5: **return**  $(\tilde{F}, \tilde{G})$
- 

Nonetheless, Algorithm 15 also presents a quadratic space and quasi-cubic time complexities, which makes it the main bottleneck of the efficiency of the NTRU key generation algorithm. This reduces the performance of schemes like [HHP<sup>+</sup>03] or [DLP14] which rely on that method. Furthermore, for some schemes like Latte [CG17], that require to solve an NTRU equation during every extraction of a user secret key, this represents an all the more significant drawback.

### 4.3.3 Using the Field Norm for Power-of-Two Cyclotomic Fields

Therefore, to gain in efficiency, schemes like Falcon [FHK<sup>+</sup>17] make use of a new solver in its key generation process to compute the NTRU polynomials  $F, G$  from the sampled polynomials  $f$  and  $g$ . Their technique exploits the tower of rings structure of the fields they are working on, in order to deal with larger coefficients more efficiently.

This technique was introduced in [PP19], in which the authors present new methods and algorithms for solving the NTRU equation. These algorithms result both from the recursive application of the field norm to the classic NTRU solver NTRUSolveResultant described in Section 4.3.2, and from improvements made to the calculations of resultants. We give a definition of the field norm below.

**Definition 34 (Field norm)** If  $\mathbb{K}$  is a number field,  $\mathbb{L}$  a Galois extension of  $\mathbb{K}$  and  $\text{Gal}(\mathbb{L}/\mathbb{K})$  the Galois group associated to  $\mathbb{L}/\mathbb{K}$ , then the field norm, denoted  $N_{\mathbb{L}/\mathbb{K}} : \mathbb{L} \longrightarrow \mathbb{K}$  (or simply  $N$  when clear from context) is defined on  $\mathbb{L}$  by:

$$N_{\mathbb{L}/\mathbb{K}}(f) = \prod_{g \in \text{Gal}(\mathbb{L}/\mathbb{K})} g(f).$$

The improvement concerning the computations of the resultant comes from the observation that for integers  $p$  and  $m$ , we have  $\text{Res}(\Phi_{pm}, f) = \text{Res}(\Phi_m, N(f))$ .

When working in a power-of-two cyclotomic fields, for some  $n = 2^k$ , we have  $\Phi_{2n} = x^n + 1$ . Taking  $p = 2$  and  $m = n$ , we can then apply the above equality repeatedly, which gives the following recursive algorithm to compute  $\text{Res}(x^n + 1, f)$ .

---

**Algorithm 17** Resultant( $n, f$ )

---

**Input:**  $f \in \mathbb{Z}[X]$  and  $n = 2^k$

**Output:** The resultant  $\text{Res}(x^n + 1, f)$  of  $f$  with  $x^n + 1$

- 1: **if**  $n = 1$  **then return**  $f_0$
  - 2: **return** Resultant( $n/2, N(f)$ )
- 

But the main improvement presented in [PP19] concerning the NTRU solver comes from recursively applying the field norm in the NTRUSolveResultant algorithm presented in Section 4.3.2. We describe the recursive algorithm making use of the field norm below.

---

**Algorithm 18** NTRUSolveTower( $n, f, g$ )

---

**Input:**  $f, g \in \mathbb{Z}[X]/(x^n + 1)$  and  $n = 2^k$

**Output:**  $F, G \in \mathbb{Z}[X]/(x^n + 1)$  such that  $fG - gF = q$

- 1: **if**  $n = 1$  **then**
  - 2:     Compute the Bézout coefficients  $u, v \in \mathbb{Z}$  such that  $uf - vg = \delta$  where  $\delta = \text{gcd}(f, g)$
  - 3:     **if**  $\delta$  doesn't divide  $q$  **then abort**
  - 4:      $(F, G) \leftarrow (vq/\delta, uq/\delta)$
  - 5:     **return**  $(F, G)$
  - 6: **else**
  - 7:      $(F', G') \leftarrow \text{NTRUSolveTower}(n/2, N(f), N(g))$
  - 8:      $F \leftarrow \left( \prod_{\phi \in \text{Gal}(\mathbb{L}/\mathbb{K})} \phi(g) \right) F'(x^2)$  and  $G \leftarrow \left( \prod_{\phi \in \text{Gal}(\mathbb{L}/\mathbb{K})} \phi(f) \right) G'(x^2)$
  - 9:     Reduce( $f, g, F, G$ )
  - 10: **return**  $(F, G)$
- 

With this new solver, exploiting the tower of subfields of the cyclotomic field we are working on helps to reduce the space and time complexities. They are now quasi-linear, and this new method thus represents an important improvement over the two solvers presented previously. This comes from the fact that the degree of the polynomials is divided by 2 after each recursion step.

We sum up the complexities of the 3 solvers we've described in Table 4.2.

Table 4.2 – Comparison of the complexities of 3 algorithms for solving the NTRU equation, where  $n$  is the dimension of the underlying ring.

NTRU Solver	Time Complexity	Space Complexity
HNF ([Coh93])	$\tilde{O}(n^3)$	$O(n^2)$
Resultant ([HHP+03])	$\tilde{O}(n^3)$	$O(n^2)$
Tower ([PP19])	$\tilde{O}(n)$	$O(n \log(n))$

#### 4.3.4 Tools for More General Fields

In order to have an efficient NTRU key generation algorithm for more general fields than power-of-two cyclotomic fields (for which the NTRUSolveTower algorithm is not usable), I studied in particular some implementation techniques which could help to speed up the resolution of NTRU equations when used with a classical NTRU solver. Some of those methods are presented below.

**Residue Number System.** The Residue Number System (RNS) allows to represent an integer  $z$  by smaller integers  $(z_1, \dots, z_n)$  by using a base of coprime integers  $(m_1, \dots, m_n)$ , where  $z_i = z \pmod{m_i}$  is called the residue of  $z$  modulo  $m_i$ . By doing so, it helps to get rid of the large integers involved in the computations done by the NTRU solver.

Representing an integer  $z$  in RNS is straightforward, while going back from the RNS representation to the standard representation is done using the Chine Remainder Theorem. In the RNS representation, operations such as addition, subtraction or multiplication are done very efficiently. Indeed, those operations can simply be performed for each moduli  $m_i$ . Therefore, it could help to speed up the computations of our NTRU solver. For other operations in the RNS representation such as modular reduction, some results [BT15] present particular methods which help to perform them efficiently.

**NTT.** Alongside the use of RNS, it is possible to use the NTT to represent polynomials. As explained in [PP19], using RNS together with NTT representations means that a polynomial  $f$  is replaced thanks to the RNS representation with  $n$  polynomials  $f_i$  with coefficients being integers modulo  $m_j$  and we can then applied, depending on the number field, the NTT representation to each of the  $f_i$ .



This further improves the efficiency of the classical NTRU solver when working on more general number fields.

**Rejecting solutions.** The key generation algorithm will occasionally fail, for example when the polynomial  $f$  is not invertible. As suggested in [PP19], we could also arbitrarily reject a couple  $(f, g)$  during the key generation process if it implies slower future computations in our implementation of the NTRU solver. In this case, as long as the rejection rate remains quite low, this could help improve further the efficiency of the key generation algorithm. Doing so, a trade-off between the time complexity of the NTRU solver and the rejection rate appears.

In [Por23], the authors thus explain that for some parameter sets for 3 different schemes, BAT [FKP<sup>+</sup>22], Falcon [FHK<sup>+</sup>17] and Hawk [DPP<sup>+</sup>22], the NTRU solver fails less than 36% of the time for  $n = 1024$ , in favor of greater efficiency.

## 4.4 A New Assumption: iNTRU

Given that NTRU cryptosystems are generally efficient in practice and allow to work with compact parameters, we explored the possibility of using gadget trapdoors, as described in Chapter 2, together with the NTRU hardness assumption. Our goal was to then obtain more efficient schemes in terms of timings and parameter sizes. In particular, we wanted to improve the efficiency of the IBE scheme described in Chapter 3 in order to close the gap between our IBE using gadget trapdoors and the DLP-IBE scheme, using GPV trapdoors together with the NTRU hardness assumption.

However, this novel approach is not straightforwardly compatible with NTRU lattices and in order to achieve the possibility of working with the NTRU hardness assumption together with gadget trapdoors, we used a variant of the NTRU hardness assumption, called iNTRU (for inhomogeneous NTRU), which was first defined in [GGH<sup>+</sup>19]. This new scheme also resulted in an implementation<sup>1</sup>, making it possible to assess its timings performance and comparing it with the previous IBE described in Chapter 3 and with the different existing IBE schemes of the literature ([DLP14] and [ZMS<sup>+</sup>21] in particular).

---

1. [https://github.com/lucasprabel/approx\\_lattice](https://github.com/lucasprabel/approx_lattice)

### 4.4.1 Definition

We first define the iNTRU hardness assumption, introduced in [GGH<sup>+</sup>19].

**Definition 35 (iNTRU<sub>q,χ</sub>)** Let  $k, q$  be integers and  $\chi$  a distribution over  $\mathcal{R}$ . The input of the iNTRU<sub>q,χ</sub> problem is a vector  $\mathbf{a} \in \mathcal{R}_q^k$  which is either taken uniform in  $\mathcal{R}_q^k$  or either set as  $\mathbf{a} = r^{-1}(\mathbf{g} + \mathbf{e})$  where  $(r, \mathbf{e})$  is drawn from  $\chi^{k+1}$ . The goal is to decide which is the case.

### 4.4.2 Hardness of iNTRU

In their paper, the authors from [GGH<sup>+</sup>19] showed a reduction of a matrix-variant of iNTRU called MiNTRU from a non-standard Ring-LWE with a trapdoor oracle access problem that they introduce. We adapted the reduction to the iNTRU problem as well.

First, let us define the  $n$ -secret LWE distribution as

$$\{(\mathbf{A}, \mathbf{B} = \mathbf{S}\mathbf{A} + \mathbf{E}) \mid \mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{S} \leftarrow \mathbb{Z}_q^{n \times n}, \mathbf{E} \leftarrow \chi^{n \times m}\}$$

for some distribution  $\chi$ .

In  $n$ -secret LWE, introduced in [GGH<sup>+</sup>19], we are given two matrices  $\mathbf{A}, \mathbf{B} \in \mathbb{Z}_q^{n \times m}$  ( $m > n$ ) with  $\mathbf{A}$  a uniformly random matrix, and need to decide if  $\mathbf{B}$  is also a uniformly random matrix or is chosen as  $\mathbf{B} = \mathbf{S}\mathbf{A} + \mathbf{E}$  with a uniform  $\mathbf{S} \in \mathbb{Z}_q^{n \times n}$  and a low norm  $\mathbf{E} \in \mathbb{Z}_q^{n \times m}$ . This problem becomes easy if we are also given a trapdoor for the matrix  $\mathbf{A}$ . But we don't know any effective distinguisher if we are given a trapdoor for the matrix  $\mathbf{B}$  instead.

We define a trapdoor oracle for an arbitrary matrix  $\mathbf{B} \in \mathbb{Z}_q^{n \times m}$  as an oracle which takes as input  $\mathbf{B}$ , a vector  $\mathbf{v} \in \mathbb{Z}_q^n$  and outputs a discrete Gaussian integer vector  $\mathbf{x} \in \mathbb{Z}^m$  conditioned on  $\mathbf{B}\mathbf{x} \bmod q = \mathbf{v}$ .

We then assume that the decision problem of this new version of LWE is still hard. In [GGH<sup>+</sup>19], the authors show a hardness reduction from this version of LWE to the iNTRU hardness assumption.

**Proposition 4 (Adapted from [GGH<sup>+</sup>19])** Let  $n \in \mathbb{N}$ ,  $q < 2^{\text{poly}(n)}$ ,  $\chi$  be a distribution over  $\mathbb{Z}_q$  and  $m \geq n \log q$ . Further, let  $q = \omega(\sqrt{m})$ . Then, the pseudorandomness of iNTRU with error distribution  $\chi^m \cdot \mathbf{b}^{-1}(\mathbf{g})$  follows from the pseudorandomness of Ring-LWE with a trapdoor oracle for  $\mathbf{b}$ .

*Proof.* We show a reduction from Ring-LWE with a trapdoor oracle for  $\mathbf{b}$  to iNTRU with error distribution  $\chi^m \cdot \mathbf{b}^{-1}(\mathbf{g})$ . We are given the input  $(\mathbf{a}, \mathbf{b})$  where  $\mathbf{a}, \mathbf{b} \in \mathcal{R}_q^m$  as a Ring-LWE instance. By calling  $k$  times the trapdoor oracle for  $\mathbf{b}$ , we get  $\mathbf{X} \leftarrow \mathbf{b}^{-1}(\mathbf{g}) \in \mathcal{R}^{m \times k}$  such that  $\mathbf{b}^T \mathbf{X} = \mathbf{g}^T \pmod{q}$ .

If  $(\mathbf{a}, \mathbf{b})$  is generated uniformly and independently, then  $\mathbf{a}\mathbf{X} \pmod{q}$  is negligibly close to uniformly random by the leftover hash lemma. Otherwise, we have  $\mathbf{b} = s\mathbf{a} + \mathbf{e}$  when  $(\mathbf{a}, \mathbf{b})$  is sampled from the Ring-LWE distribution where  $s \in \mathcal{R}_q$  is the secret. Therefore,  $\mathbf{a}^T \mathbf{X} = s^{-1}(\mathbf{b}^T - \mathbf{e}^T)\mathbf{X} = s^{-1}(\mathbf{g}^T - \mathbf{e}^T \mathbf{X}) = s^{-1}(\mathbf{g}^T - \mathbf{e}')$   $\pmod{q}$ . Hence,  $\mathbf{a}^T \mathbf{X} \pmod{q}$  is an instance of iNTRU with error distribution  $\chi^m \cdot \mathbf{b}^{-1}(\mathbf{g})$ .  $\square$

### 4.4.3 Approximate Trapdoors Based on iNTRU

In [GL20], Genise and Li introduced a family of Ring-SIS approximate trapdoors whose pseudorandomness is based on the iNTRU problem. They showed that the efficient gadget-based trapdoor framework of [MP12] exists on a family of NTRU lattices. Their trapdoor scheme enjoys small secret keys and we have shown that it is compatible with applications requiring tag matrices.

In their second trapdoor scheme, the matrix  $\begin{bmatrix} -\mathbf{e}^T \\ r\mathbf{I} \end{bmatrix}$  is used as a  $\mathbf{f}$ -trapdoor for  $\begin{bmatrix} 1 & \mathbf{a} \end{bmatrix} = \begin{bmatrix} 1 & r^{-1}(\mathbf{f}^T + \mathbf{e}^T) \end{bmatrix}$  where  $(r, \mathbf{e}) \leftarrow \chi^{k-\ell+1}$  is drawn from a distribution with small entries and  $\mathbf{f}$  is the approximate gadget vector.

We begin by describing the ApproxTrapGen algorithm which generates a public matrix  $\mathbf{a}$  together with an approximate trapdoor  $\mathbf{R}$ , whose pseudorandomness is based on the iNTRU problem.

---

#### Algorithm 19 ApproxTrapGen( $\sigma$ )

---

- 1: Sample  $r \leftarrow \mathcal{D}_{\mathcal{R}, \sigma}$ ,  $\mathbf{e} \leftarrow \mathcal{D}_{\mathcal{R}^m, \sigma}$
  - 2: Set  $\mathbf{a}' = r^{-1}(\mathbf{f} + \mathbf{e}) \in \mathcal{R}^m$
  - 3: Set  $\mathbf{a} = (1, \mathbf{a}') \in \mathcal{R}_q^{m+1}$
  - 4: Set  $\mathbf{R} = \begin{bmatrix} -\mathbf{e}^T \\ r\mathbf{I}_m \end{bmatrix} \in \mathcal{R}^{(m+1) \times m}$
  - 5: **return**  $(\mathbf{a}, \mathbf{R})$
- 

As when working with exact trapdoors, if we want to make use of approximate trapdoors in order to build cryptographic schemes, we must ensure beforehand to have efficient

sampling algorithms available. As with the exact setting, sampling approximate Gaussian preimages will be done in 2 steps: the approximate  $\mathbf{F}$ -sampling and the perturbing sampling.

The following algorithm `ApproxSamplePre` shows how to perform efficient preimage sampling algorithm for a public  $\mathbf{a}$  when given an iNTRU approximate trapdoor  $\mathbf{R}$ .

---

**Algorithm 20** `ApproxSamplePre`( $\mathbf{a}, \mathbf{R}, u, \zeta$ )

---

- 1: Sample perturbation  $\mathbf{p} \leftarrow D_{\mathcal{R}^{m+1}, \sqrt{\Sigma_p}}$
  - 2: Set coset  $v = (u - \mathbf{a}^T \mathbf{p}) \in \mathcal{R}_q$
  - 3: Sample  $\mathbf{z} = (\mathbf{z}_1^T, \mathbf{z}_2^T)^T \leftarrow D_{\Lambda_q^\perp(\mathbf{g}^T), \sigma_g}$
  - 4: Set  $\mathbf{x} = \mathbf{p} + \mathbf{R}\mathbf{z}_2 \in \mathcal{R}^{m+1}$
  - 5: **return**  $\mathbf{x}$
- 

The authors from [GL20] then got the following results, which can be compared with those described in Theorem 14 from Chapter 3.

**Theorem 16** Let  $r \leftarrow \chi$  and  $\mathbf{e}^T \leftarrow \chi^m$  and set the trapdoor function description as  $\mathbf{a} = \begin{bmatrix} 1 & \mathbf{a}' \end{bmatrix} = \begin{bmatrix} 1 & r^{-1}(\mathbf{f} + \mathbf{e}) \end{bmatrix} \in \mathcal{R}_q^{m+1}$ . Let  $\eta = \eta_\epsilon(\mathbb{Z}^{n \times m})$  and  $\sigma_g = \eta_\epsilon(\Lambda_q^\perp(\mathbf{g}^T)) \geq \sqrt{b^2 + 1} \cdot \eta_\epsilon(\mathbb{Z}^{n \times m})$  for some  $\epsilon \in (0, 1)$  and  $\zeta \succeq \sqrt{\sigma_g^2 \mathbf{R}\mathbf{R}^T + \eta^2 \mathbf{I}_{m+1}}$ . Then, the following distributions are within a max-log distance  $3 \log \frac{1+\epsilon}{1-\epsilon} \leq \frac{6\epsilon}{1-\epsilon}$ :

$$\{(\mathbf{a}, \mathbf{x}, u, y) \mid u \leftarrow \mathcal{U}(\mathcal{R}_q), \mathbf{x} \leftarrow \text{ApproxSamplePre}(\mathbf{a}, \mathbf{R}, u, \zeta), y = u - \mathbf{a}^T \mathbf{x} \in \mathcal{R}_q\}$$

and  $\{(\mathbf{a}, \mathbf{x}, u, y) \mid \mathbf{x} \leftarrow D_{\mathcal{R}^{m+1}, \zeta}, y \leftarrow D_{\mathcal{R}, \sigma_e} \pmod{q}, u = \mathbf{a}^T \mathbf{x} + y \in \mathcal{R}_q\}$

for  $\sigma_e = \sigma_g \sqrt{(b^{2\ell} - 1)/(b^2 - 1)}$ .

## 4.5 Application to a NTRU-Based IBE Scheme

In this section, we introduce an IBE scheme based on the iNTRU hardness assumption, instantiated using gadget-based approximate trapdoors over iNTRU trapdoors combined with the Dual-Regev encryption scheme over rings.

### 4.5.1 The Scheme

We recall that  $m = k - \ell$  where  $k = \lceil \log_b q \rceil$  and that the approximate gadget vector  $\mathbf{f}$  is defined as  $\mathbf{f}^T = [b^\ell \ b^{\ell+1} \ b^{\ell+2} \ \dots \ b^{k-1}] \in \mathcal{R}^{k-\ell}$ . Here, the master public key is a vector  $\mathbf{a} \in \mathcal{R}_q^{m+1}$  generated with the ApproxTrapGen algorithm and whose pseudorandomness is based on the iNTRU problem. The master secret key  $r, \mathbf{e} \in \mathcal{R}^{m+1}$  defines an  $\mathbf{f}$ -approximate trapdoor associated with  $\mathbf{a}$ . An identity is mapped to an element in  $\mathcal{R}_q$  by the use of a hash function modeled as a random oracle in the security proof; the secret key associated with an identity  $\text{id}$  is an approximate short vector  $\mathbf{x} \in \mathcal{R}^{m+1}$ .

We give below a full description of the four algorithms comprising our IBE:

- Setup( $1^n$ )  $\rightarrow$  (mpk, msk):
  - let  $\mathbf{a} = [1 \ \mathbf{a}_0^T]^T \in \mathcal{R}_q^{m+1}$  and  $\mathbf{R} = \begin{bmatrix} -\mathbf{e}^T \\ r\mathbf{I}_m \end{bmatrix} \in \mathcal{R}^{(m+1) \times m}$  output by ApproxTrapGen( $\sigma$ ); and let  $\mathcal{H} : \{0, 1\}^* \rightarrow \mathcal{R}_q$  a hash function;
  - output mpk =  $(\mathbf{a}, \mathcal{H})$  and msk =  $\mathbf{R}$ .
- Extract(mpkm, msk, id)  $\rightarrow \mathbf{x}_{\text{id}} = \mathbf{x}_2 \in \mathcal{R}^m$ :
  - define the tag  $h_{\text{id}} = \mathcal{H}(\text{id}) \in \mathcal{R}_q$ ;
  - sample a short preimage  $\mathbf{x} = (x_1, \mathbf{x}_2^T)^T \leftarrow \text{ApproxSamplePre}(\mathbf{a}, \mathbf{R}, h_{\text{id}}, \zeta)$ ;
- Encrypt(mpkm, id,  $M$ )  $\rightarrow \mathbf{C} = (\mathbf{b}, c) \in \mathcal{R}_q^{m+1}$ :
  - compute  $h_{\text{id}} = \mathcal{H}(\text{id})$ ; sample  $s \leftarrow D_{\mathcal{R}, \tau}$ ,  $\mathbf{e}_1 \leftarrow D_{\mathcal{R}^m, \tau}$ ,  $e_2 \leftarrow D_{\mathcal{R}, \tau}$ ;
  - compute  $\mathbf{b} = s\mathbf{a}_0 + \mathbf{e}_1 \in \mathcal{R}_q^m$  and  $c = h_{\text{id}} \cdot s + e_2 + \lfloor q/2 \rfloor M \in \mathcal{R}_q$ , where a message is encoded as  $M \in \mathcal{R}_2$ ;
- Decrypt( $\mathbf{x}_{\text{id}}, \mathbf{C}$ )  $\rightarrow M$ :
  - parse  $\mathbf{x}_{\text{id}}$  as  $(x_1, \mathbf{x}_2)$ ; and compute  $\text{res} \leftarrow c - \mathbf{b}^T \mathbf{x}_2$ ;
  - for each  $i$ , if the coefficient  $\text{res}_i \in \mathbb{Z}$  is closer to  $\lfloor q/2 \rfloor$  than to 0,  $M_i = 1$ , otherwise  $M_i = 0$ .

### 4.5.2 Correctness

Following the description of the Encrypt algorithm, we have the following equality:

$$\begin{aligned}
 c - \mathbf{b}^T \mathbf{x}_2 &= h_{\text{id}} \cdot s + e_2 + \lfloor q/2 \rfloor M - (s\mathbf{a}_0 + \mathbf{e}_1)^T \mathbf{x}_2 \\
 &= s \cdot x_1 + s\mathbf{a}_0^T \mathbf{x}_2 - y \cdot s + e_2 + \lfloor q/2 \rfloor M - s\mathbf{a}_0^T \mathbf{x}_2 - \mathbf{e}_1^T \mathbf{x}_2 \\
 &= \lfloor q/2 \rfloor M + s \cdot (x_1 - y) + e_2 - \mathbf{e}_1^T \mathbf{x}_2.
 \end{aligned}$$

Furthermore, the following bounds apply:

- $\|s \cdot x_1\| \leq t^2 \tau \zeta n$  from Lemma 1 and Theorem 16.
- $\|y \cdot s\| \leq t^2 \tau \sigma_g \sqrt{(b^{2\ell} - 1)/(b^2 - 1)} n$  from Lemma 1 and Theorem 16.
- $\|e_2\| \leq t\tau\sqrt{n}$  from Lemma 1.
- $\|\mathbf{e}_1^T \mathbf{x}_2\| \leq t^2 \tau \zeta nm$  from Lemma 1 and Theorem 16.

By substituting these bounds, we obtain:

$$\begin{aligned} \|s \cdot (x_1 - y) + e_2 - \mathbf{e}_1^T \mathbf{x}_2\| &\leq \|s \cdot x_1\| + \|y \cdot s\| + \|e_2\| + \|\mathbf{e}_1^T \mathbf{x}_2\| \\ &\leq t^2 \tau \left[ \zeta (m + 1) + \sigma_g \sqrt{(b^{2\ell} - 1)/(b^2 - 1)} \right] + t\tau\sqrt{n} \\ &\leq \lfloor q/4 \rfloor. \end{aligned}$$

Then, the following constraints combined with the errors norm constraints above should be satisfied to ensure the correctness of the scheme:

- The Gaussian parameter  $\sigma_g$  used for the **G**-sampling in the ApproxSamplePre algorithm must verify  $\sigma_g \geq \sqrt{2b} \cdot (2b + 1) \cdot \sqrt{\log(2nw(1 + 1/\epsilon))/\pi}$  (see [GM18], Corollary 3.1).
- The Gaussian width for preimage sampling  $\zeta$  must follow the condition  $\zeta > \sqrt{(\sigma_g^2 + 1)s_1^2(\mathbf{R}) + \eta_\epsilon^2(\mathbb{Z}^{nm})}$ , where  $s_1(\mathbf{R})$  is the spectral norm of the trapdoor  $\mathbf{R}$  (see Chapter 2, Lemma 9).

### 4.5.3 Parameters and Security

**Asymptotic Security.** The following theorem states the IND-sID-CPA of our IBE scheme.

**Theorem 17** Our IBE construction with parameters  $n, m, q, k, \ell, \sigma, \sigma_g, \zeta, \tau$  and  $\gamma$  is IND-sID-CPA secure in the random oracle model under the hardness of  $\text{iNTRU}_{q, D_{\mathbf{R}}, \sigma}$  and  $\text{Ring-LWE}_{n, q, \tau}$ .

We now give a proof of Theorem 17.

*Proof.* The idea of the proof is to simulate the view of the adversary given a Ring-LWE instance  $(a'_i, b'_i)$  for  $0 \leq i \leq m$  for which we build an attacker  $\mathcal{S}$  who makes  $Q_{\mathcal{H}}$  queries to  $\mathcal{H}$ .

We construct a new simulation game given  $\mathbf{a}_0 = [a'_1 \dots a'_m]$ .  $\mathcal{S}$  chooses an index  $i^*$  at random in  $[1, Q_{\mathcal{H}}]$  and sets  $h_{\text{id}^*} = a'_0$ .

A hash query on input  $\text{id}$  is answered as follows: it first checks whether an entry of the form  $(\text{id}, *, *, *)$  already exists in the hash table. If it is not the case, it responds with a value  $u_{\text{id}}$  such that an approximated preimage for  $\mathcal{H}(\text{id}) = h_{\text{id}}$  is known i.e: it samples  $\mathbf{x} \leftarrow D_{\mathcal{R}^{m+1}, \zeta}$  and  $y_{\text{id}} \leftarrow D_{\mathcal{R}, \sigma_e}$  such that  $y_{\text{id}} = (x_1^T, \mathbf{x}_2^T)^T [1 \ \mathbf{a}_0]$ , sets  $h_{\text{id}} = (x_1^T, \mathbf{x}_2^T)^T [1 \ \mathbf{a}_0] - y_{\text{id}}$  and adds the tuple  $(\text{id}, h_{\text{id}}, \mathbf{x}_2, y_{\text{id}})$  in the hash table.

An extraction query for an identity  $\text{id}$  is responded as follows: assuming that an entry for  $\text{id}$  already exists in the table, the corresponding  $\mathbf{x}_2$  is output by  $\mathcal{S}$ .

Note that the response to the extract queries is close to the response provided in the real game by Theorem 16 if  $\mathcal{H}$  is modeled as a random oracle and the way the public key is generated is indistinguishable from that in the real game under the iNTRU assumption.

When the attacker outputs two messages  $m_0, m_1$  and  $\text{id}^*$ , if  $\text{id}^*$  has not been queried to  $\mathcal{H}$ , then  $\mathcal{S}$  aborts; otherwise the challenger sets the challenger ciphertext as:

$$(\mathbf{b}, c) = \left( [b'_1 \ \dots \ b'_m], b'_0 + \lfloor q/2 \rfloor m_b \right) \in \mathcal{R}_q^{m+1}.$$

The attacker  $\mathcal{S}$  outputs the same bit as  $\mathcal{A}$ . Assuming no abort has occurred, they both have the same advantage, which concludes the proof.  $\square$

**Concrete Security.** In [GGH<sup>+</sup>19], the authors only propose a reduction from a non-standard version of LWE to iNTRU. Therefore, in the absence of a thorough study on the asymptotic and practical security of the iNTRU problem, which we leave for future work, we have chosen to estimate the security of our iNTRU instances by relying on the existing cryptanalysis on NTRU. As explained in [GGH<sup>+</sup>19], there is a syntactic link between NTRU and iNTRU. To the best of our knowledge, there is no known reduction between the two problems and the analysis of the iNTRU assumption might deserve additional study. Still, we additionally consider the practical security of NTRU for our targetted sets of parameters and we take into account the known cryptanalysis efforts on iNTRU.

For security estimation, we follow the same approach as in [GL20] to assess the concrete security of the IBE scheme. We determine the hardness of our underlying lattice problem by computing the root Hermite factor, introduced in [GN08]. Then, we use the following heuristic relation between the blocksize  $\kappa$  and the root Hermite factor  $\delta$  to find the smallest

blocksize which would break our underlying lattice problem:

$$\delta \approx \left(\frac{\kappa}{2\pi e}(\pi\kappa)^{1/\kappa}\right)^{1/2(\kappa-1)}.$$

Finally, our experiments estimate the running time of the BKZ algorithm to analyze the concrete security of the scheme. This algorithm makes use of an oracle to solve the Shortest Vector Problem (SVP) in smaller lattices. We chose the "Core-SVP" model introduced in [ADP<sup>+</sup>16] in the sieving regime as the SVP oracle for the BKZ algorithm with time complexity  $2^{0.292\kappa+16.4}$  in the blocksize  $\kappa$ .

**Cryptanalysis in the Overstretched Regime.** In [LW20], the authors adapt the attack from [KF17] to iNTRU, which they apply to the parameters originally proposed for the homomorphic encryption scheme from [GGH<sup>+</sup>19]. The attack, as in [KF17], can be performed when the modulus  $q$  is much larger than the dimension of the associated lattices (this situation is thus called the *overstretched regime*).

The targetted iNTRU instance in [LW20] has an error and secret that follow a uniform ternary distribution. Following [KF17] attack, [LW20] uses the fact that a very dense sublattice can be found in this NTRU-like lattice, because of the overstretched regime. Relying on a lemma from Pataki and Tural [PT08, Lemma 1], they can bound the volume of this sublattice and run a BKZ-reduction that leads to a full recovery of the iNTRU secret.

More recently, [DvW21] has improved the asymptotic bound given by Kirchner and Fouque [KF17] of  $q \leq n^{2.783+o(1)}$  by conducting a refined analysis which lowers the overstretched regime for NTRU with ternary distribution to the value  $q = n^{2.484+o(1)}$ . They also provide a concrete analysis, computing a bound on the modulus  $q$  for which the attacks exploiting the overstretched regime are more efficient than standard secret key recovery attacks. The authors of [DvW21] ran experiments that allow the detection of the "fatigue point", which separates these two regimes.

Therefore, a natural question would be to adapt the cryptanalysis carried out by [DvW21] to iNTRU instances, just as [LW20] leveraged the analysis from [KF17] to attack the cryptosystem [GGH<sup>+</sup>19] whose security is based on iNTRU. We found that the attack from [LW20] could thus be improved in different ways, relying on the refined analysis from [DvW21], which can indeed be adapted for the iNTRU instances we consider, where the secret and the error both follow a Gaussian distribution. Indeed, in both cases, the cryptanalysis in the overstretched regime requires performing lattice reductions on sublattices



of a NTRU-like lattice, in order to retrieve the very dense sublattice.

We present in Section 4.5.4 a detailed adaptation of this attack to iNTRU. Concerning the concrete parameters analysis of our IBE scheme, we took care to fall outside the range of parameters affected by the attack in such overstretched regimes.

First, it could be adapted using, as done in [DvW21], strong lattice reduction with progressive-BKZ instead of LLL/BKZ with blocksize 20 used in [LW20]. Preliminary experiments also give [DvW21] concrete estimates for blocksize values and BKZ insertion's positions in order to make the attack successful. They also rely on a more precise Heuristic than the GSA used in [LW20], that they call ZGSA. This heuristic, combined with the Pataki-Tural lemma, brings a more precise estimation of the profile of a random lattice basis after performing BKZ reduction. Finally, they propose a finer analysis of the estimation of the sublattices' volumes involved in the reduction phase of the attack. In particular, they do not rely on binary distributions as [LW20] does to estimate volumes.

#### 4.5.4 Cryptanalysis in the Overstretched Regime.

Therefore, we show how to adapt the cryptanalysis of the iNTRU instance carried out by [LW20] against the scheme [GGH<sup>+</sup>19] defined over random binary secrets using the recent analysis from [DvW21].

The considered attack handles iNTRU instances where the secret and the error follow Gaussian distributions and is built on the refinements of the cryptanalysis of NTRU in the overstretched regime provided by [DvW21]. The cryptanalysis carried out by [LW20] and [DvW21] both use the fact that we are able to bound the volume of an appropriate very dense sublattice of the iNTRU-lattice.

Let  $\mathbf{S} \leftarrow \chi^{n \times n}$  be an invertible matrix and  $\mathbf{E} \leftarrow \chi^{n \times m}$  where  $\chi$  is some Gaussian distribution with small standard deviation. The MiNTRU problem asks to recover  $\mathbf{S}$  from  $\mathbf{C} \in \mathbb{Z}_q^{n \times m}$ , where  $\mathbf{C}$  is defined as

$$\mathbf{C} = (\mathbf{G} - \mathbf{E})\mathbf{S}^{-1} \pmod{q}.$$

We will call this version the "matrix" version of iNTRU. In this version, the coefficients of  $\mathbf{F}$  and  $\mathbf{G}$  are independently sampled from a discrete Gaussian and the matrices have no additional structure. An additional "circulant" version is considered in [DvW21] where the NTRU secret key  $(\mathbf{f}, \mathbf{g})$  defines the matrices  $\mathbf{F}$  and  $\mathbf{G}$  by setting  $\mathbf{F}_{i,j} := \mathbf{f}_{(i+j \bmod n)}$  and  $\mathbf{G}_{i,j} := \mathbf{g}_{(i+j \bmod n)}$ . In this analysis, we only consider the matrix variant of iNTRU,

as it was also the case in [LW20].

We'll come down to square matrices by denoting  $\mathbf{C}_0$  the  $n \times n$  upper left block of the matrix  $\mathbf{C}$ . This allows us to define the following NTRU-like lattice:

$$\Lambda_q(\mathbf{C}_0) = \{(\mathbf{u}, \mathbf{v}) \mid \mathbf{C}_0 \mathbf{v} - \mathbf{u} = \mathbf{0} \text{ mod } q\}.$$

One can notice that this lattice has the following public basis matrix  $\mathbf{B}$ :

$$\mathbf{B} := \begin{bmatrix} q\mathbf{I}_n & \mathbf{C}_0 \\ \mathbf{0}_n & \mathbf{I}_n \end{bmatrix} \in \mathbb{Z}^{2n \times 2n}$$

which admits a dense sublattice of rank  $n$  defined by the following basis matrix

$$\mathbf{B}^{S, E_0} := \begin{bmatrix} \mathbf{I}_n - \mathbf{E}_0 \\ \mathbf{S} \end{bmatrix} \in \mathbb{Z}^{2n \times n}$$

where  $\mathbf{E}_0$  denotes the  $n \times n$  upper left block of the matrix  $\mathbf{E}$ .

Solving the iNTRU problem amounts to recovering the dense sublattice spanned by  $\mathbf{B}^{S, E_0}$ .

Our asymptotic analysis will use several heuristics, which we define below.

**Heuristic 1** (Gaussian heuristic). *Let  $\Lambda$  be a lattice of rank  $n$ . Then,*

$$\lambda_1(\Lambda) = \sqrt{\frac{n}{2\pi e}} \cdot \text{vol}(\Lambda)^{1/n}$$

Furthermore, we denote  $\text{gh}(n) := \sqrt{\frac{n}{2\pi e}}$  the expected value of the first minimum of a  $n$ -dimensional lattice of volume 1.

**Heuristic 2** (Geometric Series Assumption (GSA)). *Let  $\mathbf{B} = (\mathbf{b}_1, \dots, \mathbf{b}_d)$  be a BKZ- $\beta$  reduced basis. Then, the Gram-Schmidt vectors have norms satisfying:*

$$\log(\|\mathbf{b}_i^*\|) = \frac{d-1-2i}{2} \cdot \log(\alpha_\beta) + \frac{\log(\det(\mathbf{B}))}{d},$$

where  $\alpha_\beta := \text{gh}(\beta)^{2/(\beta-1)}$ .

**Heuristic 3** (Z Geometric Series Assumption (ZGSA)). *Let  $\mathbf{B} = (\mathbf{b}_1, \dots, \mathbf{b}_{2n})$  be a basis of a  $q$ -ary lattice of dimension  $2n$  with  $n$  vectors  $(q, 0, \dots, 0), (0, q, 0, \dots), \dots$ . After a*

BKZ- $\beta$  reduction, the Gram-Schmidt vectors have norms satisfying for all  $i \in [1, n]$ :

$$\|\mathbf{b}_i^*\| = \begin{cases} q & \text{if } i \leq n - m \\ \sqrt{q} \cdot \alpha_\beta^{\frac{2n-1-2i}{2}} & \text{if } n - m < i \leq n + m - 1 \\ 1 & \text{if } i \geq n + m - 1 \end{cases}$$

where  $\alpha_\beta = \mathbf{gh}(\beta)^{2/(\beta-1)}$  and  $m = \frac{1}{2} + \frac{\log(q)}{2\log(\alpha_\beta)}$ .

If we want to apply Heuristic 1 to our iNTRU instance, then the expected minimal length of the lattice spanned by  $\mathbf{B}$  should be  $\lambda_1(\Lambda(\mathbf{B})) \approx \sqrt{\frac{nq}{\pi e}}$ . However, column vectors of  $\mathbf{B}^{\mathbf{S}, \mathbf{E}_0}$  have a length of about  $\sqrt{2n\sigma^2 + (1 - 2\sigma)}$  since coefficients of  $\mathbf{S}$  and  $\mathbf{E}$  are independently sampled from the discrete Gaussian  $\chi$  of standard deviation  $\sigma$ . Indeed, if we denote  $(\mathbf{a}, \mathbf{s})$  the  $j$ -th column vector of  $\mathbf{B}^{\mathbf{S}, \mathbf{E}_0}$  for  $1 \leq j \leq n$ , with  $\mathbf{a}, \mathbf{s} \in \mathbb{Z}^n$ , we have:

$$\begin{aligned} \|(\mathbf{a}, \mathbf{s})\|^2 &= \|\mathbf{a}\|^2 + \|\mathbf{s}\|^2 \\ &= \|\tilde{\mathbf{a}}\|^2 + a_j^2 + \|\mathbf{s}\|^2 && \text{where } \tilde{\mathbf{a}} = (a_1, \dots, a_{j-1}, a_{j+1}, \dots, a_n) \\ &\approx \sigma^2(n-1) + (1-2\sigma)^2 + \sigma^2 n \\ &= 2n\sigma^2 + (1-2\sigma). \end{aligned}$$

In [DvW21], the column vectors of the secret basis have a length of about  $\sqrt{2n\sigma^2}$ . In their article, the authors state that recovering elements from the dense sublattice is enough to be able to decrypt the messages. In the same way, our objective is to recover those short vectors, so that we can retrieve the secret key  $\mathbf{S}$ .

More precisely, we wish to identify the value of the blocksize  $\beta$  for which the BKZ algorithm allows to either retrieve a vector as short as the secret key, or a vector from the dense sublattice. As in [DvW21], we define two different events in order to distinguish this standard regime from the overstretched regime. The Secret Key Recovery event  $\text{SKR}_\kappa$  happens when a vector as short as a secret key vector is inserted by BKZ in the basis at position  $\kappa$ . The Dense Sublattice Discovery  $\text{DSD}_\kappa$  happens when a dense lattice vector longer than a secret key vector is inserted by BKZ in the basis at position  $\kappa$ .

During the lattice reduction of the cryptanalysis, the first event which occurs defines into which regime we fall. For a fixed  $n$ , the "fatigue point", as it is called in [DvW21], is the value of the modulus  $q$  where the overstretched regime starts.

**Kirchner-Fouque Estimate.** First, let's begin by computing estimates for the block-size  $\beta$  following the analysis of [KF17] for our iNTRU instance.

The attack's principle in the overstretched regime relies on the Lemma 10 by Pataki and Tural [PT08] which allows to bound the volume of a sublattice. Indeed, the existence of the  $n$ -dimensional dense sublattice  $\Lambda(\mathbf{B}^{\mathbf{S}, E_0})$  gives a constraint on the Gram-Schmidt norms of the basis vectors of  $\Lambda(\mathbf{B})$ .

**Lemma 10 (Pataki and Tural)** Let  $\mathcal{L}$  be a  $d$ -dimensional lattice and  $\mathbf{b}_1, \dots, \mathbf{b}_d$  be a basis of  $\mathcal{L}$ . Then, if  $\mathcal{L}' \subset \mathcal{L}$  is a  $n$ -dimensional sublattice of  $\mathcal{L}$ , we have:

$$\text{vol}(\mathcal{L}') \geq \min_{\substack{J \subset [n] \\ |J|=n}} \prod_{j \in J} \|\mathbf{b}_j^*\|.$$

Let's apply Lemma 10 with the  $2n$ -dimensional lattice  $\mathcal{L} = \Lambda(\mathbf{B})$  and the  $n$ -dimensional sublattice  $\mathcal{L}' = \Lambda(\mathbf{B}^{\mathbf{S}, E_0}) \subset \mathcal{L}$ . Then, the right hand side of the inequality gives the volume of the projected sublattice  $\Lambda(\mathbf{B}_{[n, 2n[})$ . Thus, we obtain:

$$\text{vol}(\Lambda(\mathbf{B}^{\mathbf{S}, E_0})) \geq \text{vol}(\Lambda(\mathbf{B}_{[n, 2n[})).$$

Following Heuristic 3, we assume that the norms of the Gram-Schmidt vectors form a decreasing sequence. Thus, the right-hand side volume increases when we run the BKZ- $\beta$  algorithm with increasing block sizes  $\beta$ . Indeed, we have:

$$\begin{aligned} \prod_{i=n+1}^{2n} \|\mathbf{b}_i^*\| &= \prod_{i=n+m}^{2n} \|\mathbf{b}_i^*\| \cdot \prod_{i=n+1}^{n+m-1} \|\mathbf{b}_i^*\| \\ &= \prod_{i=n+1}^{n+m-1} \|\mathbf{b}_i^*\| \\ &= \prod_{i=n+1}^{n+m-1} \sqrt{q} \cdot \alpha_\beta^{\frac{2n-1-2i}{2}} \\ &= \prod_{i=1}^{m-1} \sqrt{q} \cdot \alpha_\beta^{\frac{-1-2i}{2}} \\ &= q^{\frac{m-1}{2}} \cdot \alpha_\beta^{-\frac{1}{2}(m-1)^2} \end{aligned}$$

with  $\alpha_\beta = \text{gh}(\beta)^{2/(\beta-1)}$  and  $m = \frac{1}{2} + \frac{\log(q)}{2 \log(\alpha_\beta)}$ .

Therefore, we can assume that for a large enough value of  $\beta$ , the right-hand side of

the previous inequality will not be upper bounded by  $\text{vol}(\Lambda(\mathbf{B}^{\mathcal{S}, \mathcal{E}_0}))$  anymore, indicating the presence of vectors from the dense sublattice.

This allows to get an estimate, similar to the one from [KF17], which states that the DSD event happens when

$$\text{vol}(\Lambda(\mathbf{B}^{\mathcal{S}, \mathcal{E}_0})) < q^{\frac{m-1}{2}} \cdot \alpha_\beta^{-\frac{1}{2}(m-1)^2}. \quad (4.1)$$

By Hadamard inequality, we have:

$$\text{vol}(\Lambda(\mathbf{B}^{\mathcal{S}, \mathcal{E}_0})) = |\det(\mathbf{B}^{\mathcal{S}, \mathcal{E}_0})| \leq \prod_{i=1}^n \|\mathbf{x}_i\| \approx (\sqrt{2n\sigma^2 + (1 - 2\sigma)})^n.$$

where  $\mathbf{x}_i$  denotes the  $i$ -th column vector from  $\mathbf{B}^{\mathcal{S}, \mathcal{E}_0}$ .

We are looking for an asymptotic estimate of the value of the blocksize  $\beta$  for which the BKZ algorithm recovers a vector from the dense sublattice. Therefore, we want an asymptotic bound on the blocksize  $\beta$  which implies the DSD event.

As in [DvW21], let's write  $q = \Theta(n^{\mathcal{Q}})$ ,  $\|(\mathbf{a}, \mathbf{s})\| = \Theta(n^{\mathcal{S}})$  (where  $(\mathbf{a}, \mathbf{s})$  denotes a column vector of  $\mathbf{B}^{\mathcal{S}, \mathcal{E}_0}$ ) and  $\beta = (\mathcal{B} + o(1))n$ , for some unknown constants  $\mathcal{Q}$ ,  $\mathcal{S}$  and  $\mathcal{B}$ . We are therefore looking for an asymptotic bound on  $\mathcal{B}$ , involving  $\mathcal{Q}$  and  $\mathcal{S}$ , that leads to the DSD event.

By definition,  $\alpha_\beta = \text{gh}(\beta)^{2/(\beta-1)}$  with  $\text{gh}(\beta) = \sqrt{\frac{n}{2\pi e}}$ . Therefore, we have  $\alpha_\beta = \left[\frac{\mathcal{B}n}{2\pi e} + o(n)\right]^{1/(\mathcal{B}n+o(n))}$ . Then, asymptotically,  $\alpha_\beta \approx (\mathcal{B}n)^{1/(\mathcal{B}n)}$ .

We also have by definition  $m = \frac{1}{2} + \frac{\log(q)}{2\log(\alpha_\beta)}$ . Therefore, asymptotically,  $m \approx \frac{\mathcal{B}\mathcal{Q}}{2}n$ .

Finally, the Hadamard inequality above gives that  $\text{vol}(\Lambda(\mathbf{B}^{\mathcal{S}, \mathcal{E}_0}))$  is upper bounded by  $n^{n\mathcal{S}+o(n)}$ . Moreover, we have  $q^{\frac{m-1}{2}} \cdot \alpha_\beta^{-\frac{1}{2}(m-1)^2} = n^{\frac{\mathcal{B}\mathcal{Q}^2}{8}+o(n)}$ . Therefore, solving Condition (4.1) gives  $\mathcal{B} \geq \frac{8\mathcal{S}}{\mathcal{Q}^2}$  as an asymptotic estimate. The obtained bound for  $\mathcal{B}$  is similar to the Kirchner-Fouque estimate computed in [DvW21], associated to the condition the blocksize  $\beta$  has to meet so that the DSD event happens.

Therefore, we get a similar asymptotic estimate when working with iNTRU instances instead of NTRU instances, in the case where we follow the analysis from [KF17].

**Ducas-van Woerden Asymptotic Estimate.** In [DvW21], Ducas and van Woerden run some experiments by making use of progressive BKZ up to blocksize 60 on their NTRU lattice, for fixed values  $n$  and  $\sigma$  and for several moduli  $q$ . Then, they looked at when the

BKZ algorithm triggered the  $\text{SKR}_\kappa$  or  $\text{DSD}_\kappa$  events.

They noticed that the  $\text{DSD}_\kappa$  event appeared most of the time at positions  $\kappa = n + k - \beta$  for  $0 < k \ll n$ . That is to say the BKZ algorithm selects  $\mathbf{w}$  as a shortest vector in  $\mathcal{L}(\mathbf{B}_{[\kappa:\min(\kappa+\beta, 2n)]})$  and lifts it to a full vector  $\mathbf{v} \in \mathcal{L}(\mathbf{B}_{[0:\min(\kappa+\beta, 2n)]})$  which is then inserted in  $\mathbf{B}$  at position  $\kappa = n + k - \beta$ . Therefore, they we have  $\pi_{n+k-\beta}(\mathbf{v}) = \mathbf{w}$ .

The resulting norm  $\|\pi_{n+k-\beta}(\mathbf{v})\|$  is then close to the expected value we got when assuming the norm of  $\mathbf{v}$  is well balanced over all the vectors  $\mathbf{b}_1^*, \dots, \mathbf{b}_{n+k}^*$ , that is to say  $\|\pi_{n+k-\beta}(\mathbf{v})\|$  is close to  $\sqrt{\frac{\beta}{n+k}}\|\mathbf{v}\|$ . Therefore, the inserted vector  $\mathbf{v}$  must necessarily be close to a shortest vector of the sublattice  $\mathcal{L}(\mathbf{B})_{[1, n+k]} \cap \mathcal{L}(\mathbf{B}^{\mathcal{S}, E_0})$ . Indeed, if it was not the case, we could assume that the projection of such a shortest vector would have a norm smaller than  $\pi_{n+k-\beta}(\mathbf{v})$  and would have been inserted instead of  $\mathbf{v}$ . This argument gives a similar claim to the one from [DvW21], which provides an equivalent of their new estimate:

**Claim 1** (Ducas-van Woerden Estimate). *Let  $\mathbf{v}$  be a shortest vector of the sublattice  $\mathcal{L}(\mathbf{B})_{[1, n+k]} \cap \mathcal{L}(\mathbf{B}^{\mathcal{S}, E_0})$  for some  $0 < k \leq n$ . Then, the BKZ algorithm with blocksize  $\beta$  triggers the DSD event when  $\|\pi_{n+k-\beta}(\mathbf{v})\| < \|\mathbf{b}_{n+k-\beta}^*\|$ .*

In our case, working with iNTRU instances, let's write  $\mathcal{L}_{\cap[1, n+k]}^{\mathcal{S}, E_0} := \mathcal{L}(\mathbf{B})_{[1, n+k]} \cap \mathcal{L}(\mathbf{B}^{\mathcal{S}, E_0})$  the intersected sublattice. In order to apply Claim 1, we want to bound  $\lambda_1(\mathcal{L}_{\cap[1, n+k]}^{\mathcal{S}, E_0})$ . To do so, we start by looking at the volume of  $\mathcal{L}_{\cap[1, n+k]}^{\mathcal{S}, E_0}$ .

We recall a result from [DvW21], which can be seen as a generalization of Lemma 10.

**Lemma 11** ([DvW21], Lemma 3.3) Let  $\mathcal{L}$  denote a  $n$ -dimensional lattice and  $\mathbf{b}_1, \dots, \mathbf{b}_n$  a basis of  $\mathcal{L}$ . For  $s > 1$  and for  $\mathcal{L}' \subset \mathcal{L}$  a  $k$ -dimensional sublattice of  $\mathcal{L}$ , we denote  $d := \dim(\mathcal{L}_{[1, s]} \cap \mathcal{L}')$ . Then, we have:

$$\text{vol}(\mathcal{L}_{[1, s]} \cap \mathcal{L}') \leq \text{vol}(\mathcal{L}') \cdot \left( \min_I \prod_{i \in I} \|\mathbf{b}_i^*\| \right)^{-1}.$$

where  $I$  ranges over all subsets of  $\{s+1, \dots, n\}$  of size  $k-d$ .

Let's assume, as in [DvW21], that the span of  $\mathbf{b}_1, \dots, \mathbf{b}_n$  and the span of  $\mathcal{L}^{\mathcal{S}, E_0}$  behave like random  $n$ -dimensional subspace. Then, we can assume with high probability that they have a trivial intersection in the  $2n$ -dimensional ambient space. Therefore, because the dense sublattice  $\mathcal{L}^{\mathcal{S}, E_0}$  is of rank  $n$ , we have  $\dim(\mathcal{L}_{\cap[1, n+k]}^{\mathcal{S}, E_0}) = k$  for all  $k \in \{0, \dots, n\}$ .

Then, let's use this result to apply Lemma 11 for  $s = n + k$  and  $\mathcal{L} := \mathcal{L}(\mathbf{B})$  a  $2n$ -

dimensional lattice with basis  $\mathbf{b}_1, \dots, \mathbf{b}_{2n}$  and its sublattice  $\mathcal{L}' := \mathcal{L}^{\mathcal{S}, \mathbf{E}_0}$  of rank  $n$ . We get the following Corollary 1.

**Corollary 1** Let's denote  $\mathcal{L}^{C_0, q}$  an iNTRU lattice with a  $n$ -dimensional dense sublattice  $\mathcal{L}^{\mathcal{S}, \mathbf{E}_0}$  such that  $\dim(\mathcal{L}_{\cap[1, n+k]}^{\mathcal{S}, \mathbf{E}_0}) = k$  for some  $k \leq 0$ . Then, we have:

$$\text{vol}(\mathcal{L}_{\cap[1, n+k]}^{\mathcal{S}, \mathbf{E}_0}) \leq \text{vol}(\mathcal{L}^{\mathcal{S}, \mathbf{E}_0}) \cdot \left( \prod_{i=n+k+1}^{2n} \|\mathbf{b}_i^*\| \right)^{-1}.$$

To apply Corollary 1, we will use Hadamard inequality again to bound  $\text{vol}(\mathcal{L}^{\mathcal{S}, \mathbf{E}_0})$ :  $\text{vol}(\Lambda(\mathbf{B}^{\mathcal{S}, \mathbf{E}_0})) \leq (\sqrt{2n\sigma^2 + (1 - 2\sigma)})^n$ .

Applying Corollary 1 together with Heuristic 3 finally gives the following asymptotic estimate.

**Claim 2.** Let's write  $q = \Theta(n^{\mathcal{Q}})$ ,  $\|(\mathbf{a}, \mathbf{s})\| = \Theta(n^{\mathcal{S}})$  (where  $(\mathbf{a}, \mathbf{s})$  denotes a column vector of  $\mathbf{B}^{\mathcal{S}, \mathbf{E}_0}$ ) and  $\beta = (\mathcal{B} + o(1))n$ , for some unknown constants  $\mathcal{Q}$ ,  $\mathcal{S}$  and  $\mathcal{B}$ . Then, the BKZ algorithm with blocksize  $\beta$  triggers the DSD event when

$$\mathcal{B} = \frac{8\mathcal{S}}{\mathcal{Q}^2 + 1} + o(1).$$

*Proof.* As before, let's apply the Hadamard inequality:  $\text{vol}(\mathcal{L}^{\mathcal{S}, \mathbf{E}_0}) \leq \|(\mathbf{a}, \mathbf{s})\|^n$ . Therefore, given  $\|(\mathbf{a}, \mathbf{s})\| = \Theta(n^{\mathcal{S}})$ , we have:

$$\begin{aligned} \log(\text{vol}(\mathcal{L}^{\mathcal{S}, \mathbf{E}_0})) &\leq n \log(\|(\mathbf{a}, \mathbf{s})\|) \\ &= n \log(O(n^{\mathcal{S}})) \\ &= Sn \log(n) + O(n). \end{aligned}$$

As above, we assume that  $\dim(\mathcal{L}_{\cap[1, n+k]}^{\mathcal{S}, \mathbf{E}_0}) = k$ . Therefore, we can apply Corollary 1, which gives:

$$\begin{aligned}
 \log(\text{vol}(\mathcal{L}_{\cap[1,n+k]}^{\mathcal{S},E_0})) &\leq \ln \text{vol}(\mathcal{L}^{\mathcal{S},E_0}) - \ln \left( \prod_{j=n+1+k}^{2n} \|\mathbf{b}_j^*\|^* \right) \\
 &\leq Sn \log(n) - \ln \left( \prod_{j=n+1+k}^{n+m-1} \|\mathbf{b}_j^*\|^* \cdot \prod_{j=n+m}^{2n} \|\mathbf{b}_j^*\|^* \right) + O(n) \\
 &\leq Sn \log(n) - \sum_{j=n+1+k}^{n+m-1} \log(\|\mathbf{b}_j^*\|) + O(n) \text{ according to the ZGSA Heuristic.}
 \end{aligned}$$

The ZGSA Heuristic also gives  $\log(\|\mathbf{b}_j^*\|) = \log(\sqrt{q} \cdot \alpha_\beta^{\frac{2n-1-2j}{2}})$ . Moreover,  $\log(\alpha_\beta) = \frac{1}{N\mathcal{B}} \log(N) + O(\frac{1}{N})$ , which finally gives

$$\log(\|\mathbf{b}_j^*\|) = \frac{1}{2} \left[ Q + \frac{2n-1-2j}{\mathcal{B}n} \right] \log(n) + O(1).$$

Now, we can compute the sum:

$$\begin{aligned}
 \sum_{j=n+1+k}^{n+m-1} \log(\|\mathbf{b}_j^*\|) &= \sum_{j=n+1+k}^{n+m-1} \frac{1}{2} \left[ Q + \frac{2n-1-2j}{\mathcal{B}n} \right] \log(n) \\
 &= \sum_{j=0}^{m-k-1} \frac{1}{2} \left[ Q - \frac{2k+2i+1}{\mathcal{B}n} \right] \log(n) \\
 &= \frac{1}{2} \log(n) \left[ (m-k) \left( Q - \frac{k+m}{\mathcal{B}n} \right) \right] \\
 &= \frac{1}{2} n \log(n) \left[ \frac{\mathcal{B}Q - 2\mathcal{K}}{2} Q - \frac{\mathcal{B}^2 Q^2 - \mathcal{K}^2}{4\mathcal{B}} \right] + O(n) \text{ because } m = \frac{\mathcal{B}Q}{2} n + O(n) \\
 &= \frac{1}{2} n \log(n) \left[ \frac{\mathcal{B}Q - 2\mathcal{K}}{2} \left( Q - \frac{\mathcal{B}Q + 2\mathcal{K}}{2\mathcal{B}} \right) \right] + O(n) \\
 &= \frac{n \log(n)}{8\mathcal{B}} (\mathcal{B}Q - 2\mathcal{K})^2 + O(n).
 \end{aligned}$$

Therefore,

$$\log(\text{vol}(\mathcal{L}_{\cap[1,n+k]}^{\mathcal{S},E_0})) \leq Sn \log(n) - \frac{(\mathcal{B}Q - 2\mathcal{K})^2}{8\mathcal{B}} n \log(n) + O(n).$$

We use this bound in conjunction with Minkowski's bound to get the following inequality on the first minimum of the intersected sublattice:



$$\log(\lambda_1(\mathcal{L}_{\cap[1,n+k]}^{\mathcal{S},E_0})) \leq \left( -\frac{(\mathcal{B}Q - 2\mathcal{K})^2}{8\mathcal{B}\mathcal{K}} + \frac{\mathcal{S}}{\mathcal{K}} + \frac{1}{2} \right) \log(n) + O(1).$$

The projection of such a vector is detected by the BKZ algorithm at position  $\kappa = n+k-\beta$  if its length is shorter than  $\|\mathbf{b}_{n+k-\beta}\|$  where  $\log(\|\mathbf{b}_{n+k-\beta}\|) = \left(\frac{1}{2}Q + \frac{\mathcal{B}-\mathcal{K}}{\mathcal{B}} \log(n)\right) + O(1)$  according to the ZGSA Heuristic.

Combining this condition with the above bound on the first minimum of the intersected sublattice gives us the following condition on  $\mathcal{B}$ :

$$\mathcal{B} \geq \frac{2\sqrt{(2\mathcal{S} - \mathcal{K})^2 + \mathcal{K}^2 Q^2} + 2(2\mathcal{S} - \mathcal{K})}{Q^2}.$$

Choosing  $\mathcal{K} = \frac{4\mathcal{S}}{Q^2+1}$  minimizes the right hand side of the inequality and reduces the condition to

$$\mathcal{B} = \frac{8\mathcal{S}}{Q^2 + 1}.$$

□

Therefore, Claim 2 allows to estimate the value of the fatigue point to  $q = n^{2.484+o(1)}$ , and we got a similar result for iNTRU instances than the one presented in [DvW21], which involved NTRU instances.

However, our asymptotic analysis needs to be backed up by additional experimental results. I have adapted the implementation proposed by [DvW21] for iNTRU instances, but our experiments concerning estimates on the size of the blocksize for different sets of parameters are still in progress. Nonetheless, the asymptotic analysis above seems to suggest that the attack for iNTRU instances in the overstretched regime is similar to that on iNTRU instances, and that in particular, we have no particular loss of security working with this variant of NTRU, in the case of overstretched attacks.

### 4.5.5 Implementation and Performance

**Timings.** Our timings have been obtained on an Intel i7-8650U CPU running at 1.9 GHz, and then scaled at 4.2GHz to compare ourselves with other schemes. Results are provided in Table 4.3. The use of the efficient gadget-based approximate trapdoor framework together with the iNTRU hardness assumption allows us to obtain efficient algorithms. The slowest of the 4 algorithms of the IBE scheme are the Setup and Extract algorithms,

$(n, q)$	$\lceil \log_2(q) \rceil$	Setup	Extract	Encrypt	Decrypt	Bit security
(256, 1073741441)	30	1.01	2.13	0.39	0.03	64
(512, 1073741441)	30	2.12	3.79	0.74	0.06	115
(1024, 1073741441)	30	4.08	7.65	1.49	0.11	223
(256, 16777601)	25	0.78	1.66	0.23	0.02	35
(512, 16777601)	25	1.48	3.00	0.49	0.04	82
(1024, 16777601)	25	3.32	5.92	1.10	0.07	159

Table 4.3 – Timings of the operations for different values of  $n$ , given in ms from Setup to Decrypt.

which correspond respectively to the approximate trapdoor generation and preimage sampling phases of the scheme. However, the Setup algorithm is usually not performed often, and the subroutine algorithms used by Extract for sampling are really modular, leaving the way for possible future improvements. Moreover, our Setup and Extract algorithms are competitive with other NTRU-based IBE (see Table 4.4 and Table 4.5).

**Comparisons.** We compare our IBE timings with the ones of [DLP14] (re-implemented in [MSO17]) and with [ZMS<sup>+</sup>21], two IBE schemes whose security is based on the NTRU hardness assumption. Our comparison experiments were carried out using equivalent parameters sets between the different schemes. In particular, we have been careful to use equivalent module sizes and equivalent noise rates when performing Gaussian sampling.

Scheme	$(n, \lceil \log_2(q) \rceil)$	Security level	Setup	Extract	Encrypt	Decrypt
[DLP14]	(512, 26)	< 80	3.84s	1.77	0.10	0.05
[DLP14]	(1024, 26)	< 192	23.93s	6.95	0.27	0.09
Our scheme	(512, 25)	82	1.48	3.00	0.49	0.04
Our scheme	(1024, 25)	159	3.32	5.92	1.10	0.07

Table 4.4 – Timings comparison of the different operations of our IBE scheme and the one from [DLP14] (the timings are extracted from the [MSO17] article, scaled up to account for CPU differences) for different parameter sets. The timings are given in ms, except for the Setup algorithm from [DLP14], which is given in seconds.

Scheme	$(n, \lceil \log_2(q) \rceil)$	Security level	Setup	Extract	Encrypt	Decrypt
[ZMS <sup>+</sup> 21]	(1024, 24)	128	102	0.82	0.05	0.06
[ZMS <sup>+</sup> 21]	(2048, 25)	256	292	2.62	0.10	0.13
[ZMS <sup>+</sup> 21]	(1024, 36)	80	165	26.4	0.08	0.09
[ZMS <sup>+</sup> 21]	(2048, 38)	160	643	57.8	0.16	0.18
Our scheme	(1024, 25)	159	3.32	5.92	1.10	0.07
Our scheme	(2048, 25)	293	10.21	12.79	2.96	0.16
Our scheme	(1024, 30)	191	4.08	7.65	1.49	0.11
Our scheme	(2048, 30)	412	12.51	16.03	3.89	0.23

Table 4.5 – Timings comparison in ms of the different operations of our IBE scheme and the one from [ZMS<sup>+</sup>21] for different sets of parameters.

We observe that we obtain better timings for the Setup algorithm than [DLP14] and [ZMS<sup>+</sup>21] and for the Extract for some sets of parameters. Furthermore, our Decrypt algorithm is slightly faster than [DLP14]. However, the Encrypt algorithm is less efficient than theirs. The use of binomial distribution improves the timings for encryption. An improvement can be obtained in our case by using the binomial distribution, but we need more samples in our case which still affects performance for encryption; we make  $n(2+m)$  calls to the integer Gaussian sampler while encryption in [DLP14] and [ZMS<sup>+</sup>21] can make use of only  $3n$  binomial sampling calls. As in [ZMS<sup>+</sup>21], our Extract algorithm is slower than the Sign algorithm from the Falcon signature scheme [FHK<sup>+</sup>17]. Note also that for a similar security level, Falcon can use smaller parameters than us.

We obtain an overhead in terms of parameter sizes compared to [DLP14] and [ZMS<sup>+</sup>21], but the trapdoor generations rely on a different paradigm. However, for completeness, we provide comparisons with [DLP14] in Table 4.6 and with [ZMS<sup>+</sup>21] in Table 4.7 in terms of parameters' sizes.

Table 4.6 – Parameters sizes comparison between our IBE and [DLP14] for different sets of parameters at a similar security level, in Bytes.

Scheme	$(n, \lceil \log_2(q) \rceil)$	Security	$mpk$	$msk$	$sk_{id}$	Ciphertext
[DLP14]	(512, 23)	< 80	1536	6144	1375	1625
[DLP14]	(1024, 27)	< 192	3584	14336	3375	3750
Our scheme	(512, 25)	$\approx 80$	14720	16192	16192	16192
Our scheme	(1024, 25)	$\approx 192$	32000	35200	35200	35200

Table 4.7 – Parameters sizes comparison between our IBE and [ZMS<sup>+</sup>21] for different sets of parameters at a similar security level, in Bytes.


Scheme	$(n, \lceil \log_2(q) \rceil)$	Security	$mpk$	$msk$	$sk_{id}$	Ciphertext
[ZMS <sup>+</sup> 21]	(1024, 36)	80	4608	18432	4608	9248
[ZMS <sup>+</sup> 21]	(1024, 24)	128	3072	12288	3072	6176
[ZMS <sup>+</sup> 21]	(2048, 25)	256	6400	25600	6400	12832
Our scheme	(512, 25)	> 80	14720	16192	16192	16192
Our scheme	(1024, 25)	> 128	32000	35200	35200	35200
Our scheme	(2048, 25)	> 256	64000	70400	70400	70400

Nonetheless, as stated in [CGM19], the use of approximate trapdoors instead of exact ones helps us to reduce the sizes of the public key and signatures by up to two times. Therefore, as expected, our obtained sizes for the master and the users' private keys and the ciphertexts are close to the ones of [GL20] whose signature scheme relies on the same paradigm as our IBE scheme.



# CONCLUSION

---

 THE last few years have been particularly exciting for the development of lattice-based cryptography. The announcement by the NIST of the candidates of its post-quantum cryptography competition to be standardized is a particularly good illustration of the interest and promise offered by lattice-based cryptography. It has established itself as the most promising candidate among the major families of post-quantum cryptography, offering a wide range of attractive features: strong theoretical results for security such as worst-case to average-case reductions ; a broad spectrum of advanced primitives such as identity-based encryption, signature with efficient protocols or fully-homomorphic encryption ; sophisticated implementation techniques and detailed concrete security analyzes of cryptosystems, . . .

The work conducted during my thesis allowed me to be part of the research works that led to this variety of characteristics, and to contribute to the growth of lattice-based cryptography.

**Lattice Trapdoors on Modules.** I presented in Chapter 2 the work I did on the development and implementation of Gaussian preimage sampling techniques on module lattices. Module lattices possess additional algebraic structure compared to unstructured ones, which leads to a more compact representation of parameters and an improved running time of the algorithms. They also have a weaker algebraic structure compared to ideal lattices, which translates in practice to a more flexible choice of parameters for cryptographic schemes and more control over the trade-off between efficiency and security. Thus, I worked on adapting different algorithms based on gadget trapdoors, such as the trapdoor generation and Gaussian sampling algorithms, in the module setting. In order to assess the performance of these algorithms, I also worked on a C implementation from scratch. Finally, relying on these tools, as applications, I worked on two instantiations and implementations of proven trapdoor-based signature schemes in the module setting, and on an identity-based encryption scheme.

---

**Approximate Trapdoors and Applications.** Chapter 3 follows on from the previous chapter, by presenting my researches for new methods to improve the efficiency of cryptographic schemes making use of gadget trapdoors. I worked on improving the efficiency of the previous identity-based encryption scheme by making use of approximate trapdoors instead of exact ones. The use of approximate trapdoors leads to the appearance of error terms which must be taken care of in the decryption phase of encryption schemes. I worked on making use of those approximate trapdoors to design an identity-based encryption scheme that can handle this error term while using tag matrices, and proposed a concrete implementation of approximate trapdoor generation and approximate Gaussian sampling algorithms, together with an implementation of this new identity-based encryption scheme, showing that the new scheme was more efficient than the counterpart IBE using exact trapdoors.

**NTRU Lattices.** Finally, I presented in Chapter 4 my work concerning NTRU lattices. I studied a variant of NTRU, called iNTRU, which allows to build cryptographic schemes making use of gadget trapdoors, while the use of gadget trapdoors is not straightforwardly compatible with the standard NTRU hardness assumption. I used that particular variant of NTRU in order to build another identity-based encryption scheme based on the [MP12] paradigm, in order to obtain a more efficient IBE in terms of timings and parameter sizes. This new scheme also resulted in an implementation, making it possible to assess its timings performance and comparing it with my previous IBE. I was also interested in the concrete security of cryptographic schemes based on the iNTRU hardness assumption, the literature being still scarce on this subject, this new problem having been introduced recently. I was more particularly interested in overstretched attacks, introduced in [KF17], which apply when the size of the modulus  $q$  is way larger compared to the dimension  $2n$  of the underlying NTRU lattice. In addition, I have also worked on the resolutions of NTRU equations, for which solvers are used during the key generation phase of several cryptosystems such as Falcon [FHK<sup>+</sup>17] or BAT [FKP<sup>+</sup>22]. My goal was to investigate techniques for effectively solving NTRU equations, when working with number fields that don't admit a tower of subfields. In this case, solving these equations efficiently as presented in [PP19] using the field norm and the subfield structure is no longer possible, and other solutions must be found. I was also interested in the possibility of avoiding floating point arithmetic during the reduce phase of the NTRU equation solving algorithms, in order to get an efficient implementation.

---

**Perspectives.** Therefore, during this thesis, I was able to explore ideas whose common objective was to improve the efficiency of cryptographic schemes based on lattices, keeping in mind in particular the potential future advent of quantum computers. Between the various schemes selected by the NIST for standardization and new ideas, some of which have been outlined in this manuscript, more recently published, lattice-based cryptography thus appears to be a sound proposal that can be used in practice.

The NIST wrote:

*The question of when a large-scale quantum computer will be built is a complicated one. While in the past it was less clear that large quantum computers are a physical possibility, many scientists now believe it to be merely a significant engineering challenge. Some engineers even predict that within the next twenty or so years sufficiently large quantum computers will be built to break essentially all public key schemes currently in use.*

This thesis is therefore part of these efforts to strengthen information security systems to withstand the capabilities of quantum computing.

However, many questions remain unanswered. Even after the NIST announcement of the schemes selected to be standardized, those schemes are not mature enough to completely ensure their security. For instance, progress needs to be made on the understanding of the hardness of the underlying lattice problems, the choice of concrete parameters to be used still requires further research, and the design of secure implementations, which must take into account aspects such as side-channel attacks, awaits extended analysis.

Some of these issues were highlighted in the new call for proposals for quantum-resistant public-key digital signature schemes issued by the NIST in 2022. In particular, they looked to broaden their range of signature schemes and desired alternatives that do not rely on structured lattices. This choice of not using structured lattices is seen as a more conservative choice for long-term security since structured lattices may suffer from specific vulnerabilities exploiting their algebraic structure.

Therefore, while lattice-based cryptography has established itself as a promising candidate among the major families of post-quantum cryptography, offering a wide range of attractive features and increasingly effective cryptographic schemes in practice, there are still many interesting challenges to explore.





# REFERENCES

---

- [AAB<sup>+</sup>17] S. Akleylek, E. Alkim, P. S. L. M. Barreto, N. Bindel, J. Buchmann, E. Eaton, G. Gutoski, J. Krämer, P. Longa, H. Polat, J. E. Ricardini, and G. Zanon. Qtesla, 2017, URL: [https://qtesla.org/wp-content/uploads/2019/01/qTESLA\\_v2.4\\_01.25.2019.pdf](https://qtesla.org/wp-content/uploads/2019/01/qTESLA_v2.4_01.25.2019.pdf).
- [ABB<sup>+</sup>17] E. Alkim, N. Bindel, J. A. Buchmann, Ö. Dagdelen, E. Eaton, G. Gutoski, J. Krämer, and F. Pawlega. Revisiting TESLA in the quantum random oracle model, in *PQCrypto*, volume 10346 of *LNCS*, pages 143–162, Springer, 2017.
- [ABB<sup>+</sup>19] E. Alkim, P. S. L. M. Barreto, N. Bindel, P. Longa, and J. E. Ricardini. The lattice-based digital signature scheme qtesla, *IACR Cryptology ePrint Archive*, 2019:85, 2019.
- [ABB10a] S. Agrawal, D. Boneh, and X. Boyen. Efficient lattice (H)IBE in the standard model, in *EUROCRYPT*, volume 6110 of *LNCS*, pages 553–572, Springer, 2010.
- [ABB10b] S. Agrawal, D. Boneh, and X. Boyen. Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical IBE, in *CRYPTO*, volume 6223 of *LNCS*, pages 98–115, Springer, 2010.
- [ABV<sup>+</sup>12] S. Agrawal, X. Boyen, V. Vaikuntanathan, P. Voulgaris, and H. Wee. Functional encryption for threshold functions (or fuzzy IBE) from lattices, in *Public Key Cryptography*, volume 7293 of *Lecture Notes in Computer Science*, pages 280–297, Springer, 2012.
- [ACD<sup>+</sup>18] M. R. Albrecht, B. R. Curtis, A. Deo, A. Davidson, R. Player, E. W. Postlethwaite, F. Virdia, and T. Wunderer. Estimate all the {lwe, ntru} schemes!, in *SCN*, volume 11035 of *LNCS*, pages 351–367, Springer, 2018.
- [AD97] M. Ajtai and C. Dwork. A public-key cryptosystem with worst-case/average-case equivalence, in *STOC*, pages 284–293, ACM, 1997.

- 
- [ADP<sup>+</sup>16] E. Alkim, L. Ducas, T. Pöppelmann, and P. Schwabe. Post-quantum key exchange - A new hope, *in USENIX Security Symposium*, pages 327–343, USENIX Association, 2016.
- [AFL<sup>+</sup>12] M. Abdalla, P. Fouque, V. Lyubashevsky, and M. Tibouchi. Tightly-secure signatures from lossy identification schemes, *in EUROCRYPT*, volume 7237 of *LNCS*, pages 572–590, Springer, 2012.
- [AHH<sup>+</sup>19] M. R. Albrecht, C. Hanser, A. Höller, T. Pöppelmann, F. Virdia, and A. Wallner. Implementing rlwe-based schemes using an RSA co-processor, *IACR TCHES*, 2019:169–208, 2019.
- [Ajt96] M. Ajtai. Generating hard instances of lattice problems (extended abstract), *in STOC*, pages 99–108, ACM, 1996.
- [Ajt99] M. Ajtai. Generating hard instances of the short basis problem, *in ICALP*, volume 1644 of *LNCS*, pages 1–9, Springer, 1999.
- [Alp15] J. Alperin-Sheriff. Short signatures with short public keys from homomorphic trapdoor functions, *in Public Key Cryptography*, volume 9020 of *LNCS*, pages 236–255, Springer, 2015.
- [APS15] M. R. Albrecht, R. Player, and S. Scott. On the concrete hardness of learning with errors, *J. Mathematical Cryptology*, 9:169–203, 2015.
- [Bab86] L. Babai. On lovász’ lattice reduction and the nearest lattice point problem, *Comb.*, 61:1–13, 1986.
- [BB13] R. E. Bansarkhani and J. A. Buchmann. Improvement and efficient implementation of a lattice-based signature scheme, *in SAC*, volume 8282 of *LNCS*, pages 48–67, Springer, 2013.
- [BDF<sup>+</sup>11] D. Boneh, Ö. Dagdelen, M. Fischlin, A. Lehmann, C. Schaffner, and M. Zhandry. Random oracles in a quantum world, *in ASIACRYPT*, volume 7073 of *LNCS*, pages 41–69, Springer, 2011.
- [BEP<sup>+</sup>21] P. Bert, G. Eberhart, L. Prabel, A. Roux-Langlois, and M. Sabt. Implementation of lattice trapdoors on modules and applications, *in PQCrypto*, volume 12841 of *Lecture Notes in Computer Science*, pages 195–214, Springer, 2021.
- [Ber22] D. J. Bernstein. Fast norm computation in smooth-degree abelian number fields, *IACR Cryptol. ePrint Arch.:*980, 2022.

- 
- [BF01] D. Boneh and M. K. Franklin. Identity-based encryption from the weil pairing, *in CRYPTO*, volume 2139 of *LNCS*, pages 213–229, Springer, 2001.
- [BFR<sup>+</sup>18] P. Bert, P. Fouque, A. Roux-Langlois, and M. Sabt. Practical implementation of ring-sis/lwe based signature and IBE, *in PQCrypto*, volume 10786 of *LNCS*, pages 271–291, Springer, 2018.
- [BG14] S. Bai and S. D. Galbraith. An improved compression technique for signatures based on learning with errors, *in CT-RSA*, volume 8366 of *LNCS*, pages 28–47, Springer, 2014.
- [BHJ<sup>+</sup>15] F. Böhl, D. Hofheinz, T. Jager, J. Koch, and C. Striecks. Confined guessing: new signatures from standard assumptions, *J. Cryptology*, 281:176–208, 2015.
- [BJR<sup>+</sup>22] K. Boudgoust, C. Jeudy, A. Roux-Langlois, and W. Wen. On the hardness of module learning with errors with short distributions, *IACR Cryptol. ePrint Arch.*:472, 2022.
- [BLP<sup>+</sup>13] Z. Brakerski, A. Langlois, C. Peikert, O. Regev, and D. Stehlé. Classical hardness of learning with errors, *in Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 575–584, 2013.
- [Boy10] X. Boyen. Lattice mixing and vanishing trapdoors: A framework for fully secure short signatures and more, *in Public Key Cryptography*, volume 6056 of *LNCS*, pages 499–517, Springer, 2010.
- [BT13] K. Bigou and A. Tisserand. Improving modular inversion in RNS using the plus-minus method, *in CHES*, volume 8086 of *Lecture Notes in Computer Science*, pages 233–249, Springer, 2013.
- [BT15] K. Bigou and A. Tisserand. Improving modular inversion in RNS using the plus-minus method, *IACR Cryptol. ePrint Arch.*:193, 2015.
- [CG17] P. Campbell and M. Groves. Practical post-quantum hierarchical identity-based encryption, *in 16th IMA International Conference on Cryptography and Coding*, 2017.
- [CGM19] Y. Chen, N. Genise, and P. Mukherjee. Approximate trapdoors for lattices and smaller hash-and-sign signatures, *in ASIACRYPT (3)*, volume 11923 of *LNCS*, pages 3–32, Springer, 2019.

- 
- [CHK<sup>+</sup>10] D. Cash, D. Hofheinz, E. Kiltz, and C. Peikert. Bonsai trees, or how to delegate a lattice basis, *in EUROCRYPT*, volume 6110 of *LNCS*, pages 523–552, Springer, 2010.
- [CN11] Y. Chen and P. Q. Nguyen. BKZ 2.0: better lattice security estimates, *in ASIACRYPT*, volume 7073 of *LNCS*, pages 1–20, Springer, 2011.
- [Coc01] C. C. Cocks. An identity based encryption scheme based on quadratic residues, *in IMACC*, volume 2260 of *LNCS*, pages 360–363, Springer, 2001.
- [Coh93] H. Cohen. *A course in computational algebraic number theory*, volume 138 of *Graduate texts in mathematics*, Springer, 1993.
- [CPS<sup>+</sup>19] C. Chuengsatiansup, T. Prest, D. Stehlé, A. Wallet, and K. Xagawa. Modfalcon: compact signatures based on module NTRU lattices, *IACR Cryptol. ePrint Arch.*, 2019:1456, 2019.
- [DDL<sup>+</sup>13] L. Ducas, A. Durmus, T. Lepoint, and V. Lyubashevsky. Lattice signatures and bimodal gaussians, *in CRYPTO (1)*, volume 8042 of *LNCS*, pages 40–56, Springer, 2013.
- [DDP<sup>+</sup>18] W. Dai, Y. Doröz, Y. Polyakov, K. Rohloff, H. Sajjadpour, E. Savas, and B. Sunar. Implementation and evaluation of a lattice-based key-policy ABE scheme, *IEEE Trans. Information Forensics and Security*, 135:1169–1184, 2018.
- [DH76] W. Diffie and M. E. Hellman. New directions in cryptography, *IEEE Trans. Inf. Theory*, 226:644–654, 1976.
- [DKL<sup>+</sup>17] L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler, and D. Stehlé. Crystals-dilithium, 2017, URL: <https://pq-crystals.org/dilithium/data/dilithium-specification.pdf>.
- [DKL<sup>+</sup>18] L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler, and D. Stehlé. Crystals-dilithium: A lattice-based digital signature scheme, *TCHES*, 20181:238–268, 2018.
- [DKR<sup>+</sup>18] J. D’Anvers, A. Karmakar, S. S. Roy, and F. Vercauteren. Saber: module-lwr based key exchange, cpa-secure encryption and cca-secure KEM, *in AFRICACRYPT*, volume 10831 of *LNCS*, pages 282–305, Springer, 2018.

- 
- [DLP14] L. Ducas, V. Lyubashevsky, and T. Prest. Efficient identity-based encryption over NTRU lattices, *in ASIACRYPT (2)*, volume 8874 of *LNCS*, pages 22–41, Springer, 2014.
- [DN12a] L. Ducas and P. Q. Nguyen. Faster gaussian lattice sampling using lazy floating-point arithmetic, *in ASIACRYPT*, volume 7658 of *Lecture Notes in Computer Science*, pages 415–432, Springer, 2012.
- [DN12b] L. Ducas and P. Q. Nguyen. Learning a zonotope and more: cryptanalysis of ntrusign countermeasures, *in ASIACRYPT*, volume 7658 of *LNCS*, pages 433–450, Springer, 2012.
- [DP16] L. Ducas and T. Prest. Fast fourier orthogonalization, *in ISSAC*, pages 191–198, ACM, 2016.
- [DPP<sup>+</sup>22] L. Ducas, E. W. Postlethwaite, L. N. Pulles, and W. P. J. van Woerden. Hawk: module LIP makes lattice signatures fast, compact and simple, *in ASIACRYPT (4)*, volume 13794 of *Lecture Notes in Computer Science*, pages 65–94, Springer, 2022.
- [DvW21] L. Ducas and W. P. J. van Woerden. NTRU fatigue: how stretched is over-stretched?, *in ASIACRYPT (4)*, 2021.
- [FHK<sup>+</sup>17] P.-A. Fouque, J. Hoffstein, P. Kirchner, V. Lyubashevsky, T. Pornin, T. Prest, T. Ricosset, G. Seiler, W. Whyte, and Z. Zhang. FALCON: fast-fourier lattice-based compact signatures over NTRU, 2017, URL: <https://falcon-sign.info/falcon.pdf>.
- [FKP<sup>+</sup>22] P. Fouque, P. Kirchner, T. Pornin, and Y. Yu. BAT: small and fast KEM over NTRU lattices, *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2022:240–265, 2022.
- [Fou13] E. Fouotsa. *Calcul des couplages et arithmetique des courbes elliptiques pour la cryptographie*, PhD thesis, Rennes 1, 2013.
- [Gam84] T. E. Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms, *in CRYPTO*, volume 196 of *Lecture Notes in Computer Science*, pages 10–18, Springer, 1984.
- [Gen09] C. Gentry. Fully homomorphic encryption using ideal lattices, *in STOC*, pages 169–178, ACM, 2009.

- 
- [GGH<sup>+</sup>19] N. Genise, C. Gentry, S. Halevi, B. Li, and D. Micciancio. Homomorphic encryption for finite automata, *in ASIACRYPT*, 2019.
- [GGH97] O. Goldreich, S. Goldwasser, and S. Halevi. Public-key cryptosystems from lattice reduction problems, *in CRYPTO*, volume 1294 of *LNCS*, pages 112–131, Springer, 1997.
- [GKV10] S. D. Gordon, J. Katz, and V. Vaikuntanathan. A group signature scheme from lattice assumptions, *in ASIACRYPT*, volume 6477 of *Lecture Notes in Computer Science*, pages 395–412, Springer, 2010.
- [GL20] N. Genise and B. Li. Gadget-based intru lattice trapdoors, *in INDOCRYPT*, 2020.
- [GLP12] T. Güneysu, V. Lyubashevsky, and T. Pöppelmann. Practical lattice-based cryptography: A signature scheme for embedded systems, *in CHES*, volume 7428 of *LNCS*, pages 530–547, Springer, 2012.
- [GM18] N. Genise and D. Micciancio. Faster gaussian sampling for trapdoor lattices with arbitrary modulus, *in EUROCRYPT (1)*, volume 10820 of *LNCS*, pages 174–203, Springer, 2018.
- [GN08] N. Gama and P. Q. Nguyen. Predicting lattice reduction, *in Advances in Cryptology–EUROCRYPT 2008: 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey, April 13-17, 2008. Proceedings 27*, pages 31–51, Springer, 2008.
- [GPR<sup>+</sup>18] K. D. Gür, Y. Polyakov, K. Rohloff, G. W. Ryan, and E. Savas. Implementation and evaluation of improved gaussian sampling for lattice trapdoors, *in WAHC@CCS*, pages 61–71, ACM, 2018.
- [GPR<sup>+</sup>19] K. D. Gür, Y. Polyakov, K. Rohloff, G. W. Ryan, H. Sajjadpour, and E. Savas. Practical applications of improved gaussian sampling for trapdoor lattices, *IEEE Trans. Computers*, 68(4):570–584, 2019.
- [GPV08] C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions, *in STOC*, pages 197–206, ACM, 2008.
- [GVW13] S. Gorbunov, V. Vaikuntanathan, and H. Wee. Attribute-based encryption for circuits, *in STOC*, pages 545–554, ACM, 2013.

- 
- [HHP<sup>+</sup>03] J. Hoffstein, N. Howgrave-Graham, J. Pipher, J. H. Silverman, and W. Whyte. NTRUSIGN: digital signatures using the NTRU lattice, *in CT-RSA*, volume 2612 of *LNCS*, pages 122–140, Springer, 2003.
- [HIL<sup>+</sup>99] J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby. A pseudorandom generator from any one-way function, *SIAM Journal on Computing*, 28<sup>4</sup>:1364–1396, 1999.
- [HPS01] J. Hoffstein, J. Pipher, and J. H. Silverman. NSS: an NTRU lattice-based signature scheme, *in EUROCRYPT*, volume 2045 of *LNCS*, pages 211–228, Springer, 2001.
- [HPS98] J. Hoffstein, J. Pipher, and J. H. Silverman. NTRU: A ring-based public key cryptosystem, *in ANTS*, volume 1423 of *LNCS*, pages 267–288, Springer, 1998.
- [IPR23] M. Izabachène, L. Prabel, and A. Roux-Langlois. Identity-based encryption from lattices using approximate trapdoors, *in Australasian Conference on Information Security and Privacy*, pages 270–290, Springer, 2023.
- [Kar16] C. F. F. Karney. Sampling exactly from the normal distribution, *ACM Trans. Math. Softw.*, 42<sup>1</sup>:3:1–3:14, 2016.
- [Ker83] A. Kerckhoffs. La cryptographie militaire, *J. Sci. Militaires*, 9<sup>4</sup>:5–38, 1883.
- [KF17] P. Kirchner and P. Fouque. Revisiting lattice attacks on overstretched NTRU parameters, *in EUROCRYPT (1)*, volume 10210 of *Lecture Notes in Computer Science*, pages 3–26, 2017.
- [Kle00] P. N. Klein. Finding the closest lattice vector when it’s unusually close, *in SODA*, pages 937–941, ACM/SIAM, 2000.
- [KLS18] E. Kiltz, V. Lyubashevsky, and C. Schaffner. A concrete treatment of fiat-shamir signatures in the quantum random-oracle model, *in EUROCRYPT (3)*, volume 10822 of *LNCS*, pages 552–586, Springer, 2018.
- [LCC14] R. W. F. Lai, H. K. F. Cheung, and S. S. M. Chow. Trapdoors for ideal lattices with applications, *in Inscrypt*, volume 8957 of *LNCS*, pages 239–256, Springer, 2014.
- [LLL<sup>+</sup>13] F. Laguillaumie, A. Langlois, B. Libert, and D. Stehlé. Lattice-based group signatures with logarithmic signature size, *in ASIACRYPT (2)*, volume 8270 of *LNCS*, pages 41–61, Springer, 2013.



- 
- [LLL82] A. K. Lenstra, H. W. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients, *Mathematische annalen*, 261ARTICLE:515–534, 1982.
- [LM08] V. Lyubashevsky and D. Micciancio. Asymptotically efficient lattice-based digital signatures, in *TCC*, volume 4948 of *LNCS*, pages 37–54, Springer, 2008.
- [LPR10] V. Lyubashevsky, C. Peikert, and O. Regev. On ideal lattices and learning with errors over rings, in *EUROCRYPT*, volume 6110 of *LNCS*, pages 1–23, Springer, 2010.
- [LPR13] V. Lyubashevsky, C. Peikert, and O. Regev. A toolkit for ring-lwe cryptography, in *EUROCRYPT*, volume 7881 of *LNCS*, pages 35–54, Springer, 2013.
- [LS15] A. Langlois and D. Stehlé. Worst-case to average-case reductions for module lattices, *DCC*, 753:565–599, 2015.
- [LS18] V. Lyubashevsky and G. Seiler. Short, invertible elements in partially splitting cyclotomic rings and applications to lattice-based zero-knowledge proofs, in *EUROCRYPT (1)*, volume 10820 of *LNCS*, pages 204–224, Springer, 2018.
- [LS19] V. Lyubashevsky and G. Seiler. NTTRU: truly fast NTRU using NTT, *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 20193:180–201, 2019.
- [LW20] C. Lee and A. Wallet. Lattice analysis on mintru problem, Cryptology ePrint Archive, Paper 2020/230, 2020.
- [Lyu09] V. Lyubashevsky. Fiat-shamir with aborts: applications to lattice and factoring-based signatures, in *ASIACRYPT*, volume 5912 of *LNCS*, pages 598–616, Springer, 2009.
- [Lyu12] V. Lyubashevsky. Lattice signatures without trapdoors, in *EUROCRYPT*, volume 7237 of *LNCS*, pages 738–755, Springer, 2012.
- [MBG<sup>+</sup>16] C. A. Melchor, J. Barrier, S. Guelton, A. Guinet, M. Killijian, and T. Lepoint. Nflib: ntt-based fast lattice library, in *CT-RSA*, volume 9610 of *LNCS*, pages 341–356, Springer, 2016.
- [Mic02] D. Micciancio. Generalized compact knapsacks, cyclic lattices, and efficient one-way functions from worst-case complexity assumptions, in *FOCS*, pages 356–365, IEEE Computer Society, 2002.

- 
- [MN17] B. Mennink and S. Neves. Optimal prfs from blockcipher designs, *IACR ToSC*, 2017 $\mathfrak{3}$ :228–252, 2017.
- [MP12] D. Micciancio and C. Peikert. Trapdoors for lattices: simpler, tighter, faster, smaller, in *EUROCRYPT*, volume 7237 of *LNCS*, pages 700–718, Springer, 2012.
- [MR07] D. Micciancio and O. Regev. Worst-case to average-case reductions based on gaussian measures, *SIAM J. Comput.*, 371:267–302, 2007.
- [MSO17] S. McCarthy, N. Smyth, and E. O’Sullivan. A practical implementation of identity-based encryption over NTRU lattices, in *IMACC*, volume 10655 of *LNCS*, pages 227–246, Springer, 2017.
- [MW17] D. Micciancio and M. Walter. Gaussian sampling over the integers: efficient, generic, constant-time, in *CRYPTO (2)*, volume 10402 of *LNCS*, pages 455–485, Springer, 2017.
- [Nat15] National Institute of Standards and Technology. SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions, technical report, 2015.
- [NR06] P. Q. Nguyen and O. Regev. Learning a parallelepiped: cryptanalysis of GGH and NTRU signatures, in *EUROCRYPT*, volume 4004 of *Lecture Notes in Computer Science*, pages 271–288, Springer, 2006.
- [Pei09] C. Peikert. Public-key cryptosystems from the worst-case shortest vector problem, in *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 333–342, 2009.
- [Pei10] C. Peikert. An efficient and parallel gaussian sampler for lattices, in *CRYPTO*, volume 6223 of *LNCS*, pages 80–97, Springer, 2010.
- [PHS19] A. Pellet-Mary, G. Hanrot, and D. Stehlé. Approx-svp in ideal lattices with pre-processing, *IACR Cryptology ePrint Archive*, 2019:215, 2019.
- [Por23] T. Pornin. Improved key pair generation for falcon, BAT and hawk, *IACR Cryptol. ePrint Arch.*:290, 2023.
- [PP19] T. Pornin and T. Prest. More efficient algorithms for the NTRU key generation using the field norm, in *Public Key Cryptography (2)*, volume 11443 of *Lecture Notes in Computer Science*, pages 504–533, Springer, 2019.

- 
- [PR06] C. Peikert and A. Rosen. Efficient collision-resistant hashing from worst-case assumptions on cyclic lattices, *in TCC*, volume 3876 of *LNCS*, pages 145–166, Springer, 2006.
- [PT08] G. Pataki and M. Tural. On sublattice determinants in reduced bases, *arXiv preprint arXiv:0804.4014*, 2008.
- [Reg05] O. Regev. On lattices, learning with errors, random linear codes, and cryptography, *in STOC*, pages 84–93, ACM, 2005.
- [Rom90] J. Rompel. One-way functions are necessary and sufficient for secure signatures, *in STOC*, pages 387–394, ACM, 1990.
- [RSA78] R. L. Rivest, A. Shamir, and L. M. Adleman. A method for obtaining digital signatures and public-key cryptosystems, *Commun. ACM*, 212:120–126, 1978.
- [SE94] C.-P. Schnorr and M. Euchner. Lattice basis reduction: improved practical algorithms and solving subset sum problems, *Mathematical programming*, 66:181–199, 1994.
- [Sei18] G. Seiler. Faster AVX2 optimized NTT multiplication for ring-lwe lattice cryptography, *IACR Cryptology ePrint Archive*, 2018:39, 2018.
- [Sha84] A. Shamir. Identity-based cryptosystems and signature schemes, *in CRYPTO*, volume 196 of *LNCS*, pages 47–53, Springer, 1984.
- [Sho94] P. W. Shor. Polynomial time algorithms for discrete logarithms and factoring on a quantum computer, *in ANTS*, volume 877 of *LNCS*, page 289, Springer, 1994.
- [SS11] D. Stehlé and R. Steinfeld. Making NTRU as secure as worst-case problems over ideal lattices, *in EUROCRYPT*, volume 6632 of *LNCS*, pages 27–47, Springer, 2011.
- [SS13] D. Stehlé and R. Steinfeld. Making ntruencrypt and ntrusign as secure as standard worst-case problems over ideal lattices, *IACR Cryptology ePrint Archive*, 2013:4, 2013.
- [SST<sup>+</sup>09] D. Stehlé, R. Steinfeld, K. Tanaka, and K. Xagawa. Efficient public key encryption based on ideal lattices, *in ASIACRYPT*, volume 5912 of *LNCS*, pages 617–635, Springer, 2009.

- 
- [The19] The European Telecommunications Standards Institute. Quantum-Safe Identity-based Encryption, Technical Report, The European Telecommunications Standards Institute, Sophia-Antipolis, France, 2019, URL: [https://www.etsi.org/deliver/etsi\\_tr/103600\\_103699/103618/01.01.01\\_60/tr\\_103618v010101p.pdf](https://www.etsi.org/deliver/etsi_tr/103600_103699/103618/01.01.01_60/tr_103618v010101p.pdf).
- [Yam16] S. Yamada. Adaptively secure identity-based encryption from lattices with asymptotically shorter public parameters, *in EUROCRYPT (2)*, volume 9666 of *LNCS*, pages 32–62, Springer, 2016.
- [YJW23] Y. Yu, H. Jia, and X. Wang. Compact lattice gadget and its applications to hash-and-sign signatures, *CoRR*, abs/2305.12481, 2023.
- [ZMS<sup>+</sup>21] R. K. Zhao, S. McCarthy, R. Steinfeld, A. Sakzad, and M. O’Neill. Quantum-safe hibe: does it cost a latte?, ePrint Archive, 2021.
- [ZY22] S. Zhang and Y. Yu. Towards a simpler lattice gadget toolkit, *in Public Key Cryptography (1)*, volume 13177 of *Lecture Notes in Computer Science*, pages 498–520, Springer, 2022.





---

**Titre :** Trappes en Cryptographie Basée sur les Réseaux Euclidiens : Applications et Implémentation

**Mot clés :** Cryptographie Basée sur les Réseaux Euclidiens, Trappes, Chiffrement Basé sur l'Identité, Variantes Structurées, Implémentation

**Résumé :** La cryptographie basée sur les réseaux euclidiens s'est imposée comme un candidat prometteur parmi les principales familles de cryptographie post-quantique, offrant un large éventail de caractéristiques attrayantes. Mes travaux s'inscrivent dans cet effort de recherche, en contribuant au développement de la cryptographie basée sur les réseaux euclidiens. Je commence par présenter le travail sur le développement et l'implémentation de techniques d'échantillonnage de préimages gaussiennes sur les réseaux modules, dont la structure algébrique conduit à une représentation plus compacte des paramètres et à une amélioration du temps d'exécution des algorithmes comparé aux réseaux euclidiens non structurés. Deux signatures et un système de chiffrement basé sur l'identité, ainsi que leur implémentation, sont donnés comme applications. Ensuite, je présente des améliorations par rapport au schéma de chiffrement basé sur l'identité précédent en utilisant des trappes approchées au lieu de trappes exactes, et je propose des implémentations concrètes de ce schéma et des algorithmes de génération de trappes approchées et d'échantillonnage gaussien. Enfin, je présente un travail sur les réseaux NTRU. J'ai notamment étudié une variante de NTRU, appelée iNTRU, qui permet de construire des schémas cryptographiques utilisant des trappes gadget.

---

**Title:** Trapdoors in Lattice-Based Cryptography: Applications and Implementation

**Keywords:** Lattice-Based Cryptography, Trapdoors, Identity-Based Encryption, Structured Variants, Implementation

**Abstract:** Lattice-based cryptography has established itself as a promising candidate among the major families of post-quantum cryptography, offering a wide range of attractive features. The work conducted during my thesis allowed me to be part of the research works that led to this variety of characteristics, and to contribute to the growth of lattice-based cryptography. I first present the work on the development and implementation of Gaussian preimage sampling techniques on module lattices, which possess additional algebraic structure compared to unstructured ones, which leads to a more compact representation of parameters and an improved running time. Two signatures and an identity-based encryption scheme, together with their implementation, are given as applications. Next, I present improvements over the previous identity-based encryption scheme by making use of approximate trapdoors instead of exact ones, and proposed concrete implementations of this scheme and of approximate trapdoor generation and Gaussian sampling algorithms. Finally, I present a work on NTRU lattices. I studied a variant of NTRU, called iNTRU, which allows to build cryptographic schemes making use of gadget trapdoors.