

Kai-Min Chung
Yu Sasaki (Eds.)

LNCS 15487

Advances in Cryptology – ASIACRYPT 2024

30th International Conference on the Theory
and Application of Cryptology and Information Security
Kolkata, India, December 9–13, 2024
Proceedings, Part IV

4
Part IV



 Springer

Lecture Notes in Computer Science

15487

Founding Editors

Gerhard Goos
Juris Hartmanis

Editorial Board Members

Elisa Bertino, *Purdue University, West Lafayette, IN, USA*

Wen Gao, *Peking University, Beijing, China*

Bernhard Steffen , *TU Dortmund University, Dortmund, Germany*

Moti Yung , *Columbia University, New York, NY, USA*

The series Lecture Notes in Computer Science (LNCS), including its subseries Lecture Notes in Artificial Intelligence (LNAI) and Lecture Notes in Bioinformatics (LNBI), has established itself as a medium for the publication of new developments in computer science and information technology research, teaching, and education.


LNCS enjoys close cooperation with the computer science R & D community, the series counts many renowned academics among its volume editors and paper authors, and collaborates with prestigious societies. Its mission is to serve this international community by providing an invaluable service, mainly focused on the publication of conference and workshop proceedings and postproceedings. LNCS commenced publication in 1973.


Kai-Min Chung · Yu Sasaki
Editors

Advances in Cryptology – ASIACRYPT 2024

30th International Conference on the Theory
and Application of Cryptology and Information Security
Kolkata, India, December 9–13, 2024
Proceedings, Part IV

Editors

Kai-Min Chung 
Academia Sinica
Taipei, Taiwan

Yu Sasaki 
NTT Social Informatics Laboratories
Tokyo, Japan

ISSN 0302-9743

ISSN 1611-3349 (electronic)

Lecture Notes in Computer Science

ISBN 978-981-96-0893-5

ISBN 978-981-96-0894-2 (eBook)

<https://doi.org/10.1007/978-981-96-0894-2>

© International Association for Cryptologic Research 2025

This work is subject to copyright. All rights are solely and exclusively licensed by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Singapore Pte Ltd.
The registered company address is: 152 Beach Road, #21-01/04 Gateway East, Singapore 189721, Singapore

If disposing of this product, please recycle the paper.

Preface

The 30th Annual International Conference on the Theory and Application of Cryptology and Information Security (Asiacrypt 2024) was held in Kolkata, India, on December 9–13, 2024. The conference covered all technical aspects of cryptology and was sponsored by the International Association for Cryptologic Research (IACR).

We received a record 433 paper submissions for Asiacrypt from around the world. The Program Committee (PC) selected 127 papers for publication in the proceedings of the conference. As in the previous year, the Asiacrypt 2024 program had three tracks.

The two program chairs are greatly indebted to the six area chairs for their great contributions throughout the paper selection process. The area chairs were Siyao Guo for Information-Theoretic and Complexity-Theoretic Cryptography, Bo-Yin Yang for Efficient and Secure Implementations, Goichiro Hanaoka for Public-Key Cryptography Algorithms and Protocols, Arpita Patra for Multi-Party Computation and Zero-Knowledge, Prabhanjan Ananth for Public-Key Primitives with Advanced Functionalities, and Tetsu Iwata for Symmetric-Key Cryptography. The area chairs helped suggest candidates to form a strong program committee, foster and moderate discussions together with the PC members assigned as paper discussion leads to form consensus, suggest decisions on submissions in their areas, and nominate outstanding PC members. We are sincerely grateful for the invaluable contributions of the area chairs.

To review and evaluate the submissions, while keeping the load per PC member manageable, we selected the PC members consisting of 105 leading experts from all over the world, in all six topic areas of cryptology, and we also had approximately 468 external reviewers, whose input was critical to the selection of papers. The review process was conducted using double-blind peer review. The conference operated a two-round review system with a rebuttal phase. This year, we continued the interactive rebuttal from Asiacrypt 2023. After the reviews and first-round discussions, PC members and area chairs selected 264 submissions to proceed to the second round. The remaining 169 papers were rejected, including two desk-rejects. Then, the authors were invited to participate in a two-step interactive rebuttal phase, where the authors needed to submit a rebuttal in five days and then interact with the reviewers to address questions and concerns the following week. We believe the interactive form of the rebuttal encouraged discussions between the authors and the reviewers to clarify the concerns and contributions of the submissions and improved the review process. Then, after several weeks of second-round discussions, the committee selected the final 127 papers to appear in these proceedings. This year, we received seven resubmissions from the revise-and-resubmit experiment from Crypto 2024, of which five were accepted. The nine volumes of the conference proceedings contain the revised versions of the 127 papers that were selected. The final revised versions of papers were not reviewed again and the authors are responsible for their contents.

The PC nominated and voted for three papers to receive the Best Paper Awards. The Best Paper Awards went to Mariya Georgieva Belorgey, Sergiu Carпов, Nicolas Gama,

Sandra Guasch and Dimitar Jetchev for their paper “Revisiting Key Decomposition Techniques for FHE: Simpler, Faster and More Generic”, Xiaoyang Dong, Yingxin Li, Fukang Liu, Siwei Sun and Gaoli Wang for their paper “The First Practical Collision for 31-Step SHA-256”, and Valerio Cini and Hoeteck Wee for their paper “Unbounded ABE for Circuits from LWE, Revisited”. The authors of those three papers were invited to submit extended versions of their papers to the Journal of Cryptology.

The program of Asiacrypt 2024 also featured the 2024 IACR Distinguished Lecture delivered by Paul Kocher and one invited talk, nominated and voted by the PC. The invited speaker had not yet been determined when this preface was written. Following Eurocrypt 2024, we selected seven PC members for the Distinguished PC Members Awards, nominated by the area chairs and program chairs. The Outstanding PC Members Awards went to Sherman S. M. Chow, Elizabeth Crites, Matthias J. Kannwischer, Mustafa Khairallah, Ruben Niederhagen, Maciej Obremski and Keita Xagawa.

Following Crypto 2024, Asiacrypt 2024 included an artifact evaluation process for the first time. Authors of accepted papers were invited to submit associated artifacts, such as software or datasets, for archiving alongside their papers; 14 artifacts were submitted. Rei Ueno was the Artifact Chair and led an artifact evaluation committee of 10 members listed below. In the interactive review process between authors and reviewers, the goal was not just to evaluate artifacts but also to improve them. Artifacts that passed successfully through the artifact review process were publicly archived by the IACR at <https://artifacts.iacr.org/>.

Numerous people contributed to the success of Asiacrypt 2024. We would like to thank all the authors, including those whose submissions were not accepted, for submitting their research results to the conference. We are very grateful to the area chairs, PC members, and external reviewers for contributing their knowledge and expertise, and for the tremendous amount of work that was done with reading papers and contributing to the discussions. We are greatly indebted to Bimal Kumar Roy, the General Chairs, for their efforts in organizing the event, to Kevin McCurley and Kay McKelly for their help with the website and review system, and to Jih-Wei Shih for the assistance with the use of the review system. We thank the Asiacrypt 2024 advisory committee members Bart Preneel, Huaxiong Wang, Bo-Yin Yang, Goichiro Hanaoka, Jian Guo, Ron Steinfeld, and Michel Abdalla for their valuable suggestions. We are also grateful for the helpful advice and organizational material provided to us by Crypto 2024 PC co-chairs Leonid Reyzin and Douglas Stebila, Eurocrypt 2024 PC co-chairs Marc Joye and Gregor Leander, and TCC 2023 chair Hoeteck Wee. We also thank the team at Springer for handling the publication of these conference proceedings.

December 2024

Kai-Min Chung
Yu Sasaki

Organization

General Chair

Bimal Kumar Roy

TCG CREST Kolkata, India

Program Committee Chairs

Kai-Min Chung

Academia Sinica, Taiwan

Yu Sasaki

NTT Social Informatics Laboratories Tokyo,
Japan and National Institute of Standards and
Technology, USA

Area Chairs

Prabhanjan Ananth

University of California, Santa Barbara, USA

Siyao Guo

NYU Shanghai, China

Goichiro Hanaoka

National Institute of Advanced Industrial Science
and Technology, Japan

Tetsu Iwata

Nagoya University, Japan

Arpita Patra

Indian Institute of Science Bangalore, India

Bo-Yin Yang

Academia Sinica, Taiwan

Program Committee

Akshima

NYU Shanghai, China

Bar Alon

Ben-Gurion University, Israel

Elena Andreeva

TU Wien, Austria

Nuttapong Attrapadung

AIST, Japan

Subhadeep Banik

University of Lugano, Switzerland

Zhenzhen Bao

Tsinghua University, China

James Bartusek

University of California, Berkeley, USA

Hanno Becker

Amazon Web Services, UK

Sonia Belaïd

CryptoExperts, France

Ward Beullens

IBM Research, Switzerland

Andrej Bogdanov

University of Ottawa, Canada

Pedro Branco	Max Planck Institute for Security and Privacy, Germany
Gaëtan Cassiers	UCLouvain, Belgium
Céline Chevalier	CRED, Université Paris-Panthéon-Assas, and DIENS, France
Avik Chakraborti	Institute for Advancing Intelligence TCG CREST, India
Nishanth Chandran	Microsoft Research India, India
Jie Chen	East China Normal University, China
Yu Long Chen	KU Leuven and National Institute of Standards and Technology, Belgium
Mahdi Cheraghchi	University of Michigan, USA
Nai-Hui Chia	Rice University, USA
Wonseok Choi	Purdue University, USA
Tung Chou	Academia Sinica, Taiwan
Arka Rai Choudhuri	NTT Research, USA
Sherman S. M. Chow	Chinese University of Hong Kong, China
Chitchanok Chuengsatiansup	University of Melbourne, Australia
Michele Ciampi	University of Edinburgh, UK
Valerio Cini	NTT Research, USA
Elizabeth Crites	Web3 Foundation, Switzerland
Nico Döttling	CISPA Helmholtz Center, Germany
Avijit Dutta	Institute for Advancing Intelligence TCG CREST, India
Daniel Escudero	JP Morgan AlgoCRYPT CoE and JP Morgan AI Research, USA
Thomas Espitau	PQShield, France
Jun Furukawa	NEC Corporation, Japan
Rosario Gennaro	CUNY, USA
Junqing Gong	East China Normal University, China
Rishab Goyal	University of Wisconsin-Madison, USA
Julia Hesse	IBM Research Europe, Switzerland
Akinori Hosoyamada	NTT Social Informatics Laboratories, Japan
Michael Hutter	PQShield, Austria
Takanori Isobe	University of Hyogo, Japan
Joseph Jaeger	Georgia Institute of Technology, USA
Matthias J. Kannwischer	Chelpis Quantum Corp, Taiwan
Bhavana Kanukurthi	Indian Institute of Science, India
Shuichi Katsumata	PQShield and AIST, Japan
Jonathan Katz	Google and University of Maryland, USA
Mustafa Khairallah	Lund University, Sweden
Fuyuki Kitagawa	NTT Social Informatics Laboratories, Japan

Karen Klein	ETH Zurich, Switzerland
Mukul Kulkarni	Technology Innovation Institute, United Arab Emirates
Po-Chun Kuo	WisdomRoot Tech, Taiwan
Joouyoung Lee	KAIST, South Korea
Wei-Kai Lin	University of Virginia, USA
Feng-Hao Liu	Washington State University, USA
Jiahui Liu	Massachusetts Institute of Technology, USA
Qipeng Liu	UC San Diego, USA
Shengli Liu	Shanghai Jiao Tong University, China
Chen-Da Liu-Zhang	Lucerne University of Applied Sciences and Arts and Web3 Foundation, Switzerland
Yun Lu	University of Victoria, Canada
Ji Luo	University of Washington, USA
Silvia Mella	Radboud University, Netherlands
Peihan Miao	Brown University, USA
Daniele Micciancio	UCSD, USA
Yusuke Naito	Mitsubishi Electric Corporation, Japan
Khoa Nguyen	University of Wollongong, Australia
Ruben Niederhagen	Academia Sinica, Taiwan and University of Southern Denmark, Denmark
Maciej Obremski	National University of Singapore, Singapore
Miyako Ohkubo	NICT, Japan
Eran Omri	Ariel University, Israel
Jiaxin Pan	University of Kassel, Germany
Anat Paskin-Cherniavsky	Ariel University, Israel
Goutam Paul	Indian Statistical Institute, India
Chris Peikert	University of Michigan, USA
Christophe Petit	University of Birmingham and Université libre de Bruxelles, Belgium
Rachel Player	Royal Holloway University of London, UK
Thomas Prest	PQShield, France
Shahram Rasoolzadeh	Ruhr University Bochum, Germany
Alexander Russell	University of Connecticut, USA
Santanu Sarkar	IIT Madras, India
Sven Schäge	Eindhoven University of Technology, Netherlands
Gregor Seiler	IBM Research Europe, Switzerland
Sruthi Sekar	Indian Institute of Technology, India
Yaobin Shen	Xiamen University, China
Danping Shi	Institute of Information Engineering, Chinese Academy of Sciences, China
Yifan Song	Tsinghua University, China

Katerina Sotiraki	Yale University, USA
Akshayaram Srinivasan	University of Toronto, Canada
Marc Stöttinger	Hochschule RheinMain, Germany
Akira Takahashi	J.P. Morgan AI Research and AlgoCRYPT CoE, USA
Qiang Tang	University of Sydney, Australia
Aishwarya Thiruvengadam	IIT Madras, India
Emmanuel Thomé	Inria Nancy, France
Junichi Tomida	NTT Social Informatics Laboratories, Japan
Monika Trimoska	Eindhoven University of Technology, Netherlands
Huaxiong Wang	Nanyang Technological University, Singapore
Meiqin Wang	Shandong University, China
Qingju Wang	Telecom Paris, Institut Polytechnique de Paris, France
David Wu	UT Austin, USA
Keita Xagawa	Technology Innovation Institute, United Arab Emirates
Chaoping Xing	Shanghai Jiaotong University, China
Shiyuan Xu	University of Hong Kong, China
Anshu Yadav	IST, Austria
Shota Yamada	AIST, Japan
Yu Yu	Shanghai Jiao Tong University, China
Mark Zhandry	NTT Research, USA
Hong-Sheng Zhou	Virginia Commonwealth University, USA

Additional Reviewers

Hugo Aaronson	Jiawei Bao
Damiano Abram	Jyotirmoy Basak
Hamza Abusalah	Nirupam Basak
Abtin Afshar	Gabrielle Beck
Siddharth Agarwal	Hugo Beguinet
Navid Alapati	Amit Behera
Miguel Ambrona	Mihir Bellare
Parisa Amiri Eliasi	Tamar Ben David
Ravi Anand	Aner Moshe Ben Efraim
Saikrishna Badrinarayanan	Fabrice Benhamouda
Chen Bai	Tyler Besselman
David Balbás	Tim Beyne
Brieuc Balon	Rishabh Bhadauria
Gustavo Banegas	Divyanshu Bhardwaj
Laasya Bangalore	Shivam Bhasin

Amit Singh Bhati
Loïc Bidoux
Alexander Bienstock
Jan Bobolz
Alexandra Boldyreva
Maxime Bombar
Nicolas Bon
Carl Bootland
Jonathan Bootle
Giacomo Borin
Cecilia Boschini
Jean-Philippe Bossuat
Mariana Botelho da Gama
Christina Boura
Pierre Briaud
Jeffrey Burdges
Fabio Campos
Yibo Cao
Pedro Capitão
Ignacio Cascudo
David Cash
Wouter Castryck
Anirban Chakrabartha
Debasmita Chakraborty
Suvradip Chakraborty
Kanad Chakravarti
Ayantika Chatterjee
Rohit Chatterjee
Jorge Chavez-Saab
Binyi Chen
Bohang Chen
Long Chen
Mingjie Chen
Shiyao Chen
Xue Chen
Yu-Chi Chen
Chen-Mou Cheng
Jiaqi Cheng
Ashish Choudhury
Miranda Christ
Qiaohan Chu
Eldon Chung
Hao Chung
Léo Colisson
Daniel Collins
Jolijn Cottaar
Murilo Coutinho
Eric Crockett
Bibhas Chandra Das
Nayana Das
Pratish Datta
Alex Davidson
Hannah Davis
Leo de Castro
Luca De Feo
Thomas Decru
Giovanni Deligios
Ning Ding
Fangqi Dong
Minxin Du
Qiuyan Du
Jesko Dujmovic
Moumita Dutta
Pranjal Dutta
Duyen
Marius Eggert
Solane El Hirsch
Andre Esser
Hülya Evkan
Sebastian Faller
Yanchen Fan
Niklas Fassbender
Hanwen Feng
Xiutao Feng
Dario Fiore
Scott Fluhrer
Danilo Francati
Shiuan Fu
Georg Fuchsbauer
Shang Gao
Rachit Garg
Gayathri Garimella
Pierrick Gaudry
François Gérard
Paul Gerhart
Riddhi Ghosal
Shibam Ghosh
Ashrujit Ghoshal
Shane Gibbons
Valerie Gilchrist

Xinxin Gong	Yuval Ishai
Lorenzo Grassi	Ryoma Ito
Scott Griffy	Amit Jana
Chaowen Guan	Ashwin Jha
Aurore Guillevic	Xiaoyu Ji
Sam Gunn	Yanxue Jia
Felix Günther	Mingming Jiang
Kanav Gupta	Lin Jiao
Shreyas Gupta	Haoxiang Jin
Kamil Doruk Gur	Zhengzhong Jin
Jincheol Ha	Chris Jones
Hossein Hadipour	Eliran Kachlon
Tovoheray Hajatiana Randrianarisoa	Giannis Kaklamanis
Shai Halevi	Chethan Kamath
Shuai Han	Soumya Kanti Saha
Tobias Handirk	Sabyasachi Karati
Yonglin Hao	Harish Karthikeyan
Zihan Hao	Andes Y. L. Kei
Keisuke Hara	Jean Kieffer
Keitaro Hashimoto	Jiseung Kim
Aditya Hegde	Seongkwang Kim
Andreas Hellenbrand	Sebastian Kolby
Paul Hermouet	Sreehari Kollath
Minki Hhan	Dimitris Kolonelos
Hilder Lima	Venkata Koppula
Taiga Hiroka	Abhiram Kothapalli
Ryo Hiromasa	Stanislav Kruglik
Viet Tung Hoang	Anup Kumar Kundu
Charlotte Hoffmann	Péter Kutas
Clément Hoffmann	Norman Lahr
Man Hou Hong	Qiqi Lai
Wei-Chih Hong	Yi-Fu Lai
Alexander Hoover	Abel Laval
Fumitaka Hoshino	Guirec Lebrun
Patrick Hough	Byeonghak Lee
Yao-Ching Hsieh	Changmin Lee
Chengcong Hu	Hyung Tae Lee
David Hu	Joohee Lee
Kai Hu	Keewoo Lee
Zihan Hu	Yeongmin Lee
Hai Huang	Yongwoo Lee
Mi-Ying Huang	Andrea Lesavourey
Yu-Hsuan Huang	Baiyu Li
Zhicong Huang	Jiangtao Li
Shih-Han Hung	Jianqiang Li

Junru Li
Liran Li
Minzhang Li
Shun Li
Songsong Li
Weihan Li
Wenzhong Li
Yamin Li
Yanan Li
Yu Li
Yun Li
Zeyong Li
Zhe Li
Chuanwei Lin
Fuchun Lin
Yao-Ting Lin
Yunhao Ling
Eik List
Fengrun Liu
Fukang Liu
Hanlin Liu
Hongqing Liu
Rui Liu
Tianren Liu
Xiang Liu
Xiangyu Liu
Zeyu Liu
Paul Lou
George Lu
Zhenghao Lu
Ting-Gian Lua
You Lyu
Jack P. K. Ma
Yiping Ma
Varun Madathil
Lorenzo Magliocco
Avishek Majumder
Nikolaos Makriyannis
Varun Maram
Chloe Martindale
Elisaweta Masserova
Jake Massimo
Loïc Masure
Takahiro Matsuda
Christian Matt
Subhra Mazumdar
Nikolas Melissaris
Michael Meyer
Ankit Kumar Misra
Anuja Modi
Deep Inder Mohan
Charles Momin
Johannes Mono
Hart Montgomery
Ethan Mook
Thorben Moos
Tomoyuki Morimae
Hiraku Morita
Tomoki Moriya
Aditya Morolia
Christian Mouchet
Nicky Mouha
Tamer Mour
Changrui Mu
Arindam Mukherjee
Pratyay Mukherjee
Anne Müller
Alice Murphy
Shyam Murthy
Kohei Nakagawa
Barak Nehoran
Patrick Neumann
Lucien K. L. Ng
Duy Nguyen
Ky Nguyen
Olga Nissenbaum
Anca Nitulescu
Julian Nowakowski
Frederique Oggier
Jean-Baptiste Orfila
Emmanuela Orsini
Tapas Pal
Ying-yu Pan
Roberto Parisella
Aditi Partap
Alain Passelègue
Alice Pellet-Mary
Zachary Pepin
Octavio Perez Kempner
Edoardo Perichetti

Léo Perrin	Sina Schaeffler
Naty Peter	André Schrottenloher
Richard Petri	Jacob Schuldt
Rafael del Pino	Mark Schultz
Federico Pintore	Mahdi Sedaghat
Erik Pohle	Jae Hong Seo
Simon Pohmann	Yannick Seurin
Guru Vamsi Policharla	Aein Shahmirzadi
Daniel Pollman	Girisha Shankar
Yuriy Polyakov	Yixin Shen
Alexander Poremba	Rentaro Shiba
Eamonn Postlethwaite	Ardeshir Shojaeinasab
Sihang Pu	Jun Jie Sim
Luowen Qian	Mark Simkin
Tian Qiu	Jaspal Singh
Rajeev Raghunath	Benjamin Smith
Srinivasan Raghuraman	Yongha Son
Mostafizar Rahman	Fang Song
Mahesh Rajasree	Yongsoo Song
Somindu Chaya Ramanna	Pratik Soni
Simon Rastikian	Pierre-Jean Spaenlehauer
Anik Raychaudhuri	Matthias Johann Steiner
Martin Rehberg	Lukas Stennes
Michael Reichle	Roy Stracovsky
Krijn Reijnders	Takeshi Sugawara
Doreen Riepel	Adam Suhl
Guilherme Rito	Siwei Sun
Matthieu Rivain	Elias Suvanto
Bhaskar Roberts	Koutarou Suzuki
Marc Roeschlin	Erkan Tairi
Michael Rosenberg	Atsushi Takayasu
Paul Rösler	Kaoru Takemure
Arnab Roy	Abdullah Talayhan
Lawrence Roy	Quan Quan Tan
Luigi Russo	Gang Tang
Keegan Ryan	Khai Hanh Tang
Markku-Juhani Saarinen	Tianxin Tang
Éric Sageloli	Yi Tang
Dhiman Saha	Stefano Tessaro
Sayandeep Saha	Sri AravindaKrishnan Thyagarajan
Yusuke Sakai	Yan Bo Ti
Kosei Sakamoto	Jean-Pierre Tillich
Subhabrata Samajder	Toi Tomita
Simona Samardjiska	Aleksei Udovenko
Maria Corte-Real Santos	Arunachalaeswaran V.

Aron van Baarsen	Kyosuke Yamashita
Wessel van Woerden	Jiayun Yan
Michiel Verbauwhede	Yingfei Yan
Corentin Verhamme	Qianqian Yang
Quoc-Huy Vu	Rupeng Yang
Benedikt Wagner	Xinrui Yang
Julian Wälde	Yibin Yang
Hendrik Waldner	Zhaomin Yang
Judy Walker	Yizhou Yao
Alexandre Wallet	Kevin Yeo
Han Wang	Eylon Yogev
Haoyang Wang	Yusuke Yoshida
Jiabo Wang	Aaram Yun
Jiafan Wang	Gabriel Zaid
Liping Wang	Riccardo Zanotto
Mingyuan Wang	Shang Zehua
Peng Wang	Hadas Zeilberger
Weihao Wang	Runzhi Zeng
Yunhao Wang	Bin Zhang
Zhedong Wang	Cong Zhang
Yohei Watanabe	Liu Zhang
Chenkai Weng	Tianwei Zhang
Andreas Weninger	Tianyu Zhang
Stella Wohnig	Xiangyang Zhang
Harry W. H. Wong	Yijian Zhang
Ivy K. Y. Woo	Yinuo Zhang
Tiger Wu	Yuxin Zhang
Yu Xia	Chang-an Zhao
Zejun Xiang	Tianyu Zhao
Yuting Xiao	Yu Zhou
Ning Xie	Yunxiao Zhou
Zhiye Xie	Zhelei Zhou
Lei Xu	Zibo Zhou
Yanhong Xu	Chenzhi Zhu
Haiyang Xue	Ziqi Zhu
Aayush Yadav	Cong Zuo
Saikumar Yadugiri	

Artifact Chair

Rei Ueno

Kyoto University, Japan

Artifact Evaluation Committee

Julien Béguinot	LTCI, Télécom Paris, Institut Polytechnique de Paris, France
Aron Gohr	Independent Researcher
Hosein Hadipour	Graz University of Technology, Austria
Akira Ito	NTT Social Informatics Laboratories, Japan
Haruto Kimura	University of Melbourne, Australia and Waseda University, Japan
Kotaro Matsuoka	Kyoto University, Japan
Florian Mendel	Infineon Technologies, Germany
Hiraku Morita	Aarhus University, University of Copenhagen, Denmark
Prasanna Ravi	Nanyang Technological University, Singapore
Élise Tasso	Tohoku University, Japan

Contents – Part IV

Post-quantum Cryptography

Measure-Rewind-Extract: Tighter Proofs of One-Way to Hiding and CCA Security in the Quantum Random Oracle Model	3
<i>Jiangxia Ge, Heming Liao, and Rui Xue</i>	
A Tight Security Proof for SPHINCS ⁺ , Formally Verified	35
<i>Manuel Barbosa, François Dupressoir, Andreas Hülsing, Matthias Meijers, and Pierre-Yves Strub</i>	
MinRank Gabidulin Encryption Scheme on Matrix Codes	68
<i>Nicolas Aragon, Alain Couvreur, Victor Dyzeryn, Philippe Gaborit, and Adrien Vinçotte</i>	
Tighter Proofs for PKE-to-KEM Transformation in the Quantum Random Oracle Model	101
<i>Jinrong Chen, Yi Wang, Rongmao Chen, Xinyi Huang, and Wei Peng</i>	

Secure Data Structures

Deletions and Dishonesty: Probabilistic Data Structures in Adversarial Settings	137
<i>Mia Filić, Keran Kocher, Ella Kummer, and Anupama Unnikrishnan</i>	
Concurrent Encrypted Multimaps	169
<i>Archita Agarwal, Seny Kamara, and Tarik Moataz</i>	

Lattice-based Cryptography


Verifiable Oblivious Pseudorandom Functions from Lattices: Practical-Ish and Thresholdisable	205
<i>Martin R. Albrecht and Kamil Doruk Gur</i>	
Unbounded ABE for Circuits from LWE, Revisited	238
<i>Valerio Cini and Hoeteck Wee</i>	
Partially Non-interactive Two-Round Lattice-Based Threshold Signatures	268
<i>Rutchathon Chairattana-Apirom, Stefano Tessaro, and Chenzhi Zhu</i>	

Lova: Lattice-Based Folding Scheme from Unstructured Lattices	303
<i>Giacomo Fenzi, Christian Knabenhans, Ngoc Khanh Nguyen, and Duc Tu Pham</i>	
Lattice Assumptions	
On the Spinor Genus and the Distinguishing Lattice Isomorphism Problem	329
<i>Cong Ling, Jingbo Liu, and Andrew Mendelsohn</i>	
Cryptanalysis of Rank-2 Module-LIP with Symplectic Automorphisms	359
<i>Hengyi Luo, Kaijie Jiang, Yanbin Pan, and Anyu Wang</i>	
Dense and Smooth Lattices in Any Genus	386
<i>Wessel van Woerden</i>	
Evasive LWE Assumptions: Definitions, Classes, and Counterexamples	418
<i>Chris Brzuska, Akin Ünäl, and Ivy K. Y. Woo</i>	
Author Index	451

Post-quantum Cryptography



Measure-Rewind-Extract: Tighter Proofs of One-Way to Hiding and CCA Security in the Quantum Random Oracle Model

Jiangxia Ge^{1,2} , Heming Liao^{1,2} , and Rui Xue^{1,2} 

¹ Key Laboratory of Cyberspace Security Defense, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100084, China

{gejiangxia,liaoheming,xuerui}@iie.ac.cn

² School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100049, China

Abstract. The One-Way to Hiding (O2H) theorem, first given by Unruh (J ACM 2015) and then restated by Ambainis et al. (CRYPTO 2019), is a crucial technique for solving the reprogramming problem in the quantum random oracle model (QROM). It provides an upper bound $d \cdot \sqrt{\epsilon}$ for the distinguisher's advantage, where d is the query depth and ϵ denotes the advantage of a one-wayness attacker. Later, in order to obtain a tighter upper bound, Kuchta et al. (EUROCRYPT 2020) proposed the Measure-Rewind-Measure (MRM) technique and then proved the Measure-Rewind-Measure O2H (MRM-O2H) theorem, which provides the upper bound $d \cdot \epsilon$. They also proposed an open question: Can we combine their MRM technique with Ambainis et al.'s semi-classical oracle technique (CRYPTO 2019) or Zhandry's compressed oracle technique (CRYPTO 2019) to prove a new O2H theorem with an upper bound even tighter than $d \cdot \epsilon$?

In this paper, we give an affirmative answer for the above question. We propose a new technique named Measure-Rewind-Extract (MRE) by combining the MRM technique with the semi-classical oracle technique. By using MRE technique, we prove the Measure-Rewind-Extract O2H (MRE-O2H) theorem, which provides the upper bound $\sqrt{d} \cdot \epsilon$.

As an important application of our MRE-O2H theorem, for the FO^\perp , FO_m^\perp , FO^\perp and FO_m^\perp proposed by Hofheinz et al. (TCC 2017), i.e., the key encapsulation mechanism (KEM) variants of the Fujisaki-Okamoto transformation, we prove the following results in the QROM:

- Their IND-CCA security can be reduced to the IND-CPA security of the underlying public key encryption (PKE) scheme without the square-root advantage loss. In particular, compared with the IND-CCA proof of FO^\perp given by Kuchta et al. (EUROCRYPT 2020), ours removes the injectivity assumption and has a tighter security bound.
- Under the assumption that the underlying PKE scheme is unique randomness recoverable, we for the first time prove that their IND-CCA security can be reduced to the OW-CPA security of the underlying PKE scheme without the square-root advantage loss.

Keywords: quantum random oracle model · security proof · Fujisaki-Okamoto transformation · key encapsulation mechanism

1 Introduction

The Fujisaki-Okamoto (FO) transformation [10] is used to construct a public key encryption (PKE) scheme that is secure against the indistinguishability under chosen-ciphertext attacks (IND-CCA) in the random oracle model (ROM) [3]. The PKE scheme constructed by FO is based on a weakly secure PKE scheme, which can only be secure against the indistinguishability under chosen-plaintext attacks (IND-CPA) or the one-wayness under one-way attacks (OW-CPA). Compared to directly constructing an IND-CCA-secure PKE scheme, it is considered easier and more efficient to first construct an IND-CCA-secure key encapsulation mechanism (KEM) scheme and then derive an IND-CCA-secure PKE scheme via the KEM-DEM paradigm [6]. Following this fact, Dent [8] designed the first KEM variant of FO, which can be used to construct IND-CCA-secure KEM schemes in the ROM. Further, Hofheinz et al. [13] designed some KEM variants of FO including FO^{\perp} , FO_m^{\perp} , FO^{\perp} and FO_m^{\perp} . They proved that the KEM schemes constructed by these variants are IND-CCA-secure in the ROM. Indeed, these variants are also called the FO-like transformations, the \perp (resp. \perp) indicates that the variant is implicit (resp. explicit) rejection type, in which a pseudorandom value (resp. an abort symbol \perp) is returned if the ciphertext fails to decapsulate.

The FO-like transformations are widely adopted in the NIST post-quantum cryptography standardisation process [26], and hence their post-quantum security has received much attention. As argued by Boneh et al. [5], to fully assess post-quantum security, the ROM should be lifted to the quantum random oracle model (QROM). This means that having only ROM security proof of FO-like transformations is not enough, and we also need QROM security proof.

Up to now, a long sequence of works [4, 14, 17, 18, 22, 27] have provided the QROM security proofs of FO-like transformations, they all focused on the widely accepted IND-CCA security and gave different security bounds. Simultaneously, all those works used the original One-Way to Hiding (O2H) theorem [1, 28] (or its variant) to solve the reprogramming problem in the QROM. Here the reprogramming problem can be described informally as follows.

- **The reprogramming problem.** To reprogram a random function $G : X \rightarrow Y$ at a subset $S \subseteq X$ is to replace G with a new function H , where $H(x)$ is resampled on $x \in S$ and $H(x) = G(x)$ on $x \notin S$, i.e., G and H only differ on S . The reprogramming problem is, for any distinguisher \mathcal{A} making parallel queries with depth d^1 , bound its distinguishing advantage

$$\text{Adv}(\mathcal{A}) := |\Pr[b = 1 : b \leftarrow \mathcal{A}^G] - \Pr[b = 1 : b \leftarrow \mathcal{A}^H]|. \quad (1)$$

This problem is said to be in the QROM if \mathcal{A} has quantum access to its oracle.

The original O2H theorem [1, 28] designs a one-wayness attacker \mathcal{B}_{ow} , which has oracle access to H and generates its output x by measuring \mathcal{A} 's oracle query. And \mathcal{B}_{ow} 's one-wayness advantage $\text{Adv}(\mathcal{B}_{\text{ow}}) := \Pr[x \in S : x \leftarrow \mathcal{B}_{\text{ow}}^H]$ satisfies $\text{Adv}(\mathcal{A}) \leq 2d \cdot \sqrt{\text{Adv}(\mathcal{B}_{\text{ow}})}$, where d is the query depth of \mathcal{A} .

¹ See Supplementary Material A.1 of our full version [11] for more of parallel query.

From the proof strategies of the long sequence of works [4, 14, 17, 18, 22, 27], one can find that the upper bound of $\text{Adv}(\mathcal{A})$ influences the tightness of their IND-CCA security proofs. Roughly speaking, the tighter the upper bound of $\text{Adv}(\mathcal{A})$, the tighter their IND-CCA security proofs. Since a tighter security proof means more freedom in the parameter selection, many tighter O2H variants have been proved and used in those long sequence of works, and three representative variants are shown in Table 1. As we can see, these variants are all proved by using some novel techniques and giving more “power” (i.e. oracle 1_S or G) to the one-wayness attacker \mathcal{B}_{ow} , and their upper bounds of $\text{Adv}(\mathcal{A})$ are all indeed tighter than the $2d \cdot \sqrt{\text{Adv}(\mathcal{B}_{\text{ow}})}$ proved by the original O2H theorem [1, 28].

Table 1. Three O2H variants. Here \mathcal{A} makes parallel queries to its oracle with query depth d . The $|S|$ denotes the number of elements in set S . The 1_S denotes the indicator function of set S , i.e., $1_S(x) = 1$ if $x \in S$ and 0 otherwise.

O2H theorem	Proved by	$ S $	$\text{Adv}(\mathcal{A}) \leq$	\mathcal{B}_{ow} 's oracle
Original O2H [1, 28]	\backslash semi-classical oracle technique [1]	Arbitrary	$2d \cdot \sqrt{\text{Adv}(\mathcal{B}_{\text{ow}})}$	H
SC-O2H [1]		Arbitrary	$2\sqrt{d \cdot \text{Adv}(\mathcal{B}_{\text{ow}})}$	H and 1_S^a
DS-O2H [4]	compressed oracle technique [31]	One	$2\sqrt{\text{Adv}(\mathcal{B}_{\text{ow}})}$	H and G
MRM-O2H ^b [22]	Measure-Rewind-Measure (MRM) technique [22]	Arbitrary	$4d \cdot \text{Adv}(\mathcal{B}_{\text{ow}})$	H and G

^a The SC-O2H theorem actually requires that \mathcal{B}_{ow} has oracle access to $H \setminus S$. Since $H \setminus S$ can be implemented knowing H and 1_S , we just write H and 1_S here for simplicity.

^b The MRM-O2H theorem additionally requires that the event used by \mathcal{A} to distinguish G and H is efficiently checkable by itself. In fact, as shown in Eq. (1), the distinguisher \mathcal{A} considered in our paper uses the event $b = 1$ to distinguish G and H , which must be efficiently checkable by \mathcal{A} . So we omit this requirement in this table for simplicity.

Although the three O2H variants shown in Table 1 all have tighter upper bounds, there are extra restrictions during their application, respectively. In more detail, the SC-O2H theorem needs the \mathcal{B}_{ow} to have oracle access to both H and 1_S , which means that when we try to use \mathcal{B}_{ow} to attack the underlying hard problem, we need to find a way to simulate the additional 1_S for \mathcal{B}_{ow} . In order to achieve this, it seems that we need to clearly know the set S or at least some values related to S^2 . For the DS-O2H and MRM-O2H theorem, the situation is even worse, as their \mathcal{B}_{ow} even requires oracle access to both H and G . Indeed, since G and H only differ on the set S , requiring oracle access to both H and G seems stronger than to H and 1_S , in the way that one can determine whether $x \in S$ by testing if $G(x) = H(x)$.

² E.g. $S = \{w\}$ and we can get $f(w)$, where f is a public one-way injective function. At this point, we can compute $1_S(x)$ as: $1_S(x) = 1$ if $f(x) = f(w)$ and 0 otherwise.

Fortunately, these restrictions of \mathcal{B}_{ow} are not completely unattainable, at least they can be achieved when proving the IND-CCA security of FO-like transformations in the QROM. Roughly speaking, in the security proof, due to the special properties of the underlying PKE scheme and the structure of FO-like transformations, one can successfully simulate $(H \text{ and } 1_S)/(H \text{ and } G)$ for \mathcal{B}_{ow} . In fact, the (QROM) IND-CCA security proof of FO^\perp provided by Kuchta et al. [22] is done in that way: firstly use the MRM-O2H theorem to obtain the corresponding \mathcal{B}_{ow} , then simulate H and G for \mathcal{B}_{ow} , and finally use \mathcal{B}_{ow} to attack the OW-CPA security of the underlying PKE scheme. One thing we would like to stress is that, since the upper bound provided by the MRM-O2H theorem avoids the square-root advantage loss (see Table 1), Kuchta et al.’s security proof also avoids the square-root advantage loss.

In short, after the long sequence of works [4, 14, 17, 18, 22, 27], a tighter O2H theorem seems necessary if we want to give tighter QROM security proofs of the FO-like transformations. However, it is quite challenging to prove a tighter O2H theorem, Kuchta et al. also proposed the following question in [22]:

Can we combine their MRM technique with the semi-classical oracle technique or the compressed oracle technique to prove a new O2H theorem that is tighter than their MRM-O2H theorem? And can we use this new O2H theorem to give tighter IND-CCA security proofs of the FO-like transformations in the QROM?

1.1 Our Contribution

Our answer to the above question is yes. We propose a new technique named

Measure-Rewind-Extract (MRE)

by combining the MRM technique with the semi-classical oracle technique. Then, by using our MRE technique, we prove a new O2H theorem (Theorem 4) named

Measure-Rewind-Extract O2H (MRE-O2H).

It shows that $\text{Adv}(\mathcal{A}) \leq 4\sqrt{d} \cdot \text{Adv}(\mathcal{B}_{\text{ow}})$, where d is \mathcal{A} ’s query depth and \mathcal{B}_{ow} has oracle access to H, G and 1_S . Note that this upper bound is tighter than the $4d \cdot \text{Adv}(\mathcal{B}_{\text{ow}})$ proved by the MRM-O2H theorem (see Table 1).

Remark 1. Compared with the MRM-O2H theorem, which only requires that \mathcal{B}_{ow} has oracle access to H and G , our MRE-O2H theorem additionally requires the oracle access to 1_S . In fact, this additional requirement is not essential, as we can simulate 1_S by querying H and G : $1_S(x) = 1$ if $H(x) \neq G(x)$ and 0 otherwise. Intuitively speaking, this simulation is not problematic because G and H only differ on the set S . Here we point out that our MRE-O2H theorem still remains 1_S because directly providing 1_S would make the proof of this theorem more concise and understandable. For completeness, in Supplementary Material B of our full version [11], we (roughly) show that every previous work using the MRM-O2H theorem also works with our MRE-O2H theorem. Therefore, compared with the MRM-O2H theorem, there seems to be no more restrictions on the applicability of our MRE-O2H theorem.

In addition, by using our MRE-O2H theorem, we give tighter IND-CCA security proofs of the FO-like transformations FO^\perp , FO_m^\perp , FO^\perp and FO_m^\perp in the QROM, and the detailed security bounds are shown in Table 2.

Table 2. Security bounds of FO-like transformations in the QROM. Here q is the total number of queries to random oracles, d is the query depth of random oracles, and q_D is the total number of queries to the decapsulation oracle. The ‘‘Assumption’’ column shows the property that needs to be satisfied by the underlying PKE scheme. δ and ϵ respectively represent the correctness error and the security bound of the underlying PKE scheme. Here we abbreviate unique randomness recoverable as URR for simplicity.

Transformation	Underlying security	Assumption	Achieved security	Security bound
$\text{FO}^\perp, \text{FO}_m^\perp$ [19]	IND-CPA	\	IND-CCA	$\sqrt{q \cdot \epsilon} + q \cdot \sqrt{\delta}$
FO^\perp [22]	IND-CPA	η -injective	IND-CCA	$d^2 \cdot \epsilon + dq \cdot \delta + q\sqrt{\eta}$
FO_m^\perp [16]	IND-CPA	γ -spread	IND-CCA	$\sqrt{(d + q_D) \cdot \epsilon} + q^2 \cdot \delta + qq_D \cdot \sqrt{2^{-\gamma}}$
$\text{FO}^\perp, \text{FO}_m^\perp$ Corollary 1	IND-CPA	\	IND-CCA	$d^{1.5} \cdot \epsilon + q^2 \cdot \delta$
$\text{FO}^\perp, \text{FO}_m^\perp$ Corollary 2	IND-CPA	γ -spread	IND-CCA	$(d + q_D)^{1.5} \cdot \epsilon + q \cdot \sqrt{\delta} + q_D \cdot \sqrt{2^{-\gamma}}$
$\text{FO}^\perp, \text{FO}_m^\perp$ Corollary 1	OW-CPA	URR	IND-CCA	$d^{0.5} \cdot \epsilon + q^2 \cdot \delta$
$\text{FO}^\perp, \text{FO}_m^\perp$ Corollary 2	OW-CPA	URR	IND-CCA	$(d + q_D)^{0.5} \cdot \epsilon + q \cdot \sqrt{\delta}$

In more detail, our IND-CCA security proofs all avoid the square-root advantage loss incurred in [16, 19]. For the FO^\perp , when the underlying security is IND-CPA, our security proof removes the η -injective assumption used in [22] and achieves a tighter security bound. Moreover, we for the first time prove that the IND-CCA security of FO^\perp , FO_m^\perp , FO^\perp and FO_m^\perp can be reduced to the OW-CPA security of the underlying PKE scheme without the square-root advantage loss. At this point, we introduce an additional assumption of unique randomness recoverable. Roughly speaking, for a public key pk , a plaintext m and a ciphertext c , this assumption assumes that there exists an efficient algorithm Rec such that $\text{Rec}(pk, m, c) = r$ and the encryption of m with the randomness r is exactly c .

Remark 2. As shown in Table 2, compared with [16, 19], although our bounds avoid the square-root advantage loss, the loss related to the query times still exists. For example, if the underlying security is IND-CPA and $d = q$ (i.e. each parallel invoking only makes one query), the security bound of FO^\perp , FO_m^\perp achieved in [19] is $\sqrt{q \cdot \epsilon}$ while ours is $d^{1.5} \epsilon = q^{1.5} \epsilon$ (Corollary 1). At this point, it seems that determining which bound is tighter depends on the actual query times and the concrete underlying problem. Nevertheless, for the massively parallelized attacks, which have low query depth and are the typical methods to deal with high computation costs in practical cryptanalyses, our bound $d^{1.5} \epsilon$ is nearly tight.

1.2 Technique Overview

In this section, for the sake of clarity and understandability, we explain our technique by the following three steps:

- We first introduce a simple distinguisher \mathcal{A} as an example and define some notations that will be used in later explanations.
- Then, based on \mathcal{A} , we give a high-level explanation of our Measure-Rewind-Extract (MRE) technique and how we used it to prove our MRE-O2H theorem.
- Finally, we explain how we use the MRE-O2H theorem to give tighter IND-CCA security proofs of the FO-like transformations in the QROM.

A Simple Distinguisher with Query Depth 2. Recall that $G, H : X \rightarrow Y$ are random functions such that $G(x) = H(x)$ for all $x \notin S$. For the sake of simplicity, we let $S = \{m^* \in X\}$ in the following analysis. That is, there is only one point m^* where G and H differ.

Consider the following simple distinguisher \mathcal{A}^O ($O \in \{G, H\}$) that is aimed to distinguish whether O is G or H :

$$\mathcal{A}^G : \mathbb{M}_{\mathcal{A}} \circ \text{U}_2 \text{O}_G \text{U}_1 \text{O}_G |\psi\rangle, \quad \mathcal{A}^H : \mathbb{M}_{\mathcal{A}} \circ \text{U}_2 \text{O}_H \text{U}_1 \text{O}_H |\psi\rangle.$$

Here $|\psi\rangle$ is the initial state of \mathcal{A} , U_1 and U_2 are the unitary operations performed by \mathcal{A} between its oracle queries O_G/O_H , where $\text{O}_G|x, y\rangle = |x, y \oplus G(x)\rangle$ and $\text{O}_H|x, y\rangle = |x, y \oplus H(x)\rangle$. $\mathbb{M}_{\mathcal{A}} := \{\text{M}_0^{\mathcal{A}}, \text{M}_1^{\mathcal{A}}\}$ is the final projective measurement performed by \mathcal{A} , and its measurement result b (0 or 1) is \mathcal{A} 's final output. Indeed, \mathcal{A}^O considered here is a unitary quantum oracle algorithm that makes parallel queries to O with query depth 2 and query width 1³.

Before giving our explanation, we first perform some pretreatment. Define two states

$$|\psi_H\rangle := \text{U}_2 \text{O}_H \text{U}_1 \text{O}_H |\psi\rangle \text{ and } |\psi_G\rangle := \text{U}_2 \text{O}_G \text{U}_1 \text{O}_G |\psi\rangle.$$

Then, the distinguishing advantage of \mathcal{A} can be computed as follows.

$$\begin{aligned} \text{Adv}(\mathcal{A}) &= |\Pr[b = 1 : b \leftarrow \mathcal{A}^H] - \Pr[b = 1 : b \leftarrow \mathcal{A}^G]| \\ &= \left| \|\text{M}_1^{\mathcal{A}}|\psi_H\rangle\|^2 - \|\text{M}_1^{\mathcal{A}}|\psi_G\rangle\|^2 \right| \\ &\leq \left| (\text{M}_1^{\mathcal{A}}(|\psi_H\rangle - |\psi_G\rangle), \text{M}_1^{\mathcal{A}}(|\psi_H\rangle + |\psi_G\rangle)) \right| && \text{(By Lemma 3)} \\ &= \left| \left((|\psi_H\rangle - |\psi_G\rangle, (\text{M}_1^{\mathcal{A}})^\dagger \text{M}_1^{\mathcal{A}}(|\psi_H\rangle + |\psi_G\rangle)) \right) \right| && \left(\begin{array}{l} \text{Basic property} \\ \text{of inner product} \end{array} \right) \\ &= \left| (|\psi_H\rangle - |\psi_G\rangle, \text{M}_1^{\mathcal{A}}(|\psi_H\rangle + |\psi_G\rangle)) \right|. && \left(\begin{array}{l} \text{M}_1^{\mathcal{A}} \text{ is hermitian} \\ \text{and idempotent} \end{array} \right) \end{aligned} \tag{2}$$

Let $\text{M}_{m^*} := |m^*\rangle\langle m^*|$ be a projector on the oracle's input register, and let I denotes the identity operator.

³ See Supplementary Material A.1 of our full version [11] for the introduction of unitary quantum oracle algorithm and parallel query.

High-Level Explanation of Our MRE Technique. Essentially, in order to compute the upper bound of $\text{Adv}(\mathcal{A})$, our MRE technique performs the following three steps:

• **MRE-Step-1:** In this step, we use the projector $M_{m^*} = |m^*\rangle\langle m^*|$ to divide the state $|\psi_H\rangle - |\psi_G\rangle$. For the state $|\psi_H\rangle$, we have

$$\begin{aligned}
 |\psi_H\rangle &= U_2 O_H U_1 O_H |\psi\rangle \\
 &= U_2 O_H (M_{m^*} + I - M_{m^*}) U_1 O_H |\psi\rangle \\
 &= U_2 O_H M_{m^*} U_1 O_H |\psi\rangle + U_2 O_H (I - M_{m^*}) U_1 O_H |\psi\rangle \\
 &= U_2 O_H M_{m^*} U_1 O_H |\psi\rangle + U_2 O_H (I - M_{m^*}) U_1 O_H (M_{m^*} + I - M_{m^*}) |\psi\rangle \\
 &= U_2 O_H M_{m^*} U_1 O_H |\psi\rangle + U_2 O_H (I - M_{m^*}) U_1 O_H M_{m^*} |\psi\rangle \\
 &\quad + U_2 O_H (I - M_{m^*}) U_1 O_H (I - M_{m^*}) |\psi\rangle.
 \end{aligned}$$

One can see that the main idea of the above partition is to sequentially insert $(M_{m^*} + I - M_{m^*})$ before the query O_H , and then divide the entire state into two parts “ $\dots M_{m^*} \dots |\psi\rangle$ ” and “ $\dots (I - M_{m^*}) \dots |\psi\rangle$ ” by the distributive law. For the first part, we keep it unchanged, and for the second part, we divide it again by inserting $(M_{m^*} + I - M_{m^*})$ before the another query O_H . Similarly, for the state $|\psi_G\rangle$, we have

$$\begin{aligned}
 |\psi_G\rangle &= U_2 O_G M_{m^*} U_1 O_G |\psi\rangle + U_2 O_G (I - M_{m^*}) U_1 O_G M_{m^*} |\psi\rangle \\
 &\quad + U_2 O_G (I - M_{m^*}) U_1 O_G (I - M_{m^*}) |\psi\rangle.
 \end{aligned}$$

Since G and H only differ on the set $S = \{m^*\}$, the operation $O_G(I - M_{m^*})$ must be identical with the operation $O_H(I - M_{m^*})$. Based on this property,

$$\begin{aligned}
 |\psi_H\rangle - |\psi_G\rangle &= U_2 O_H M_{m^*} U_1 O_H |\psi\rangle + U_2 O_H (I - M_{m^*}) U_1 O_H M_{m^*} |\psi\rangle \\
 &\quad + U_2 O_H (I - M_{m^*}) U_1 O_H (I - M_{m^*}) |\psi\rangle \\
 &\quad - U_2 O_G M_{m^*} U_1 O_G |\psi\rangle - U_2 O_G (I - M_{m^*}) U_1 O_G M_{m^*} |\psi\rangle \\
 &\quad - U_2 O_G (I - M_{m^*}) U_1 O_G (I - M_{m^*}) |\psi\rangle \\
 &= U_2 O_H M_{m^*} U_1 O_H |\psi\rangle - U_2 O_G M_{m^*} U_1 O_G |\psi\rangle \\
 &\quad + U_2 O_H (I - M_{m^*}) U_1 O_H M_{m^*} |\psi\rangle - U_2 O_G (I - M_{m^*}) U_1 O_G M_{m^*} |\psi\rangle \\
 &= U_2 (O_H M_{m^*} U_1 O_H |\psi\rangle - O_G M_{m^*} U_1 O_G |\psi\rangle) \\
 &\quad + U_2 O_H (I - M_{m^*}) U_1 (O_H M_{m^*} |\psi\rangle - O_G M_{m^*} |\psi\rangle) \\
 &\stackrel{(a)}{=} U_2 M_{m^*} (O_H U_1 O_H |\psi\rangle - O_G U_1 O_G |\psi\rangle) \\
 &\quad + U_2 O_H (I - M_{m^*}) U_1 M_{m^*} (O_H |\psi\rangle - O_G |\psi\rangle).
 \end{aligned}$$

Here (a) uses the fact that O_G and O_H commute with M_{m^*} , which is obvious since O_G and O_H do not change the state on the oracle’s input register.

• **MRE-Step-2:** Define two states $|\psi_1\rangle := O_H U_1 O_H |\psi\rangle - O_G U_1 O_G |\psi\rangle$ and $|\psi_0\rangle := O_H |\psi\rangle - O_G |\psi\rangle$. One can see that the first step of our MRE technique

actually shows that $|\psi_H\rangle - |\psi_G\rangle = U_2 M_{m^*} |\psi_1\rangle + U_2 O_H (I - M_{m^*}) U_1 M_{m^*} |\psi_0\rangle$. Combine this equation with Eq. (2), we get

$$\begin{aligned} \text{Adv}(\mathcal{A}) &= \left| \langle |\psi_H\rangle - |\psi_G\rangle, M_1^A(|\psi_H\rangle + |\psi_G\rangle) \rangle \right| \\ &= \left| \begin{aligned} &\langle U_2 M_{m^*} |\psi_1\rangle, M_1^A(|\psi_H\rangle + |\psi_G\rangle) \rangle \\ &+ \langle U_2 O_H (I - M_{m^*}) U_1 M_{m^*} |\psi_0\rangle, M_1^A(|\psi_H\rangle + |\psi_G\rangle) \rangle \end{aligned} \right| \\ &\stackrel{(a)}{=} \left| \begin{aligned} &\langle M_{m^*} |\psi_1\rangle, M_{m^*} (U_2)^\dagger M_1^A(|\psi_H\rangle + |\psi_G\rangle) \rangle \\ &+ \langle M_{m^*} |\psi_0\rangle, M_{m^*} (U_1)^\dagger (I - M_{m^*}) O_H (U_2)^\dagger M_1^A(|\psi_H\rangle + |\psi_G\rangle) \rangle \end{aligned} \right|. \end{aligned}$$

Here (a) follows from the basic property of inner product and the fact that M_{m^*} is hermitian and idempotent.

Then, by applying Lemma 4, which guarantees that $|\langle |\alpha\rangle, |\beta\rangle \rangle + \langle |\gamma\rangle, |\delta\rangle \rangle| \leq \sqrt{\|\alpha\|^2 + \|\gamma\|^2} \cdot \sqrt{\|\beta\|^2 + \|\delta\|^2}$ for any states $|\alpha\rangle, |\beta\rangle, |\gamma\rangle, |\delta\rangle$, we rewrite $\text{Adv}(\mathcal{A})$ shown in the above equation into

$$\begin{aligned} \text{Adv}(\mathcal{A}) &\leq \sqrt{\|M_{m^*} |\psi_1\rangle\|^2 + \|M_{m^*} |\psi_0\rangle\|^2} \\ &\quad \cdot \sqrt{\|M_{m^*} (U_2)^\dagger M_1^A(|\psi_H\rangle + |\psi_G\rangle)\|^2 \\ &\quad + \|M_{m^*} (U_1)^\dagger (I - M_{m^*}) O_H (U_2)^\dagger M_1^A(|\psi_H\rangle + |\psi_G\rangle)\|^2}. \quad (3) \\ &\stackrel{(a)}{=} \sqrt{\|M_{m^*} |\psi_1\rangle\|^2 + \|M_{m^*} |\psi_0\rangle\|^2} \\ &\quad \cdot \sqrt{\|M_{m^*} O_H (U_2)^\dagger M_1^A(|\psi_H\rangle + |\psi_G\rangle)\|^2 \\ &\quad + \|M_{m^*} O_H (U_1)^\dagger (I - M_{m^*}) O_H (U_2)^\dagger M_1^A(|\psi_H\rangle + |\psi_G\rangle)\|^2}. \end{aligned}$$

Here (a) uses the fact that O_H is a unitary operation and it commutes with M_{m^*} .

• **MRE-Step-3:** Here, we will relate the above two sum of square norms with the success probabilities of two one-wayness attackers, that is, Eq. (4) and Eq. (5). For the $\|M_{m^*} |\psi_1\rangle\|^2 + \|M_{m^*} |\psi_0\rangle\|^2$, by the superposition oracle trick⁴ given in [4], we can construct two one-wayness attackers \mathcal{B}_1 and \mathcal{B}_2 such that

$$\begin{aligned} 4 \cdot \Pr[m^* \leftarrow \mathcal{B}_1^{G,H}] &= \|M_{m^*} |\psi_1\rangle\|^2 = \|M_{m^*} (O_H U_1 O_H |\psi\rangle - O_G U_1 O_G |\psi\rangle)\|^2, \\ 4 \cdot \Pr[m^* \leftarrow \mathcal{B}_2^{G,H}] &= \|M_{m^*} |\psi_0\rangle\|^2 = \|M_{m^*} (O_H |\psi\rangle - O_G |\psi\rangle)\|^2. \end{aligned}$$

Note that there is an extra constant factor “4” due to the using of superposition oracle trick. Now we can merge \mathcal{B}_1 and \mathcal{B}_2 into \mathcal{B}_3 , which uniformly chooses i from $\{1, 2\}$, runs \mathcal{B}_i and outputs its final output. Obviously, $\Pr[m^* \leftarrow \mathcal{B}_3^{G,H}]$ is equal to $1/2 \cdot \Pr[m^* \leftarrow \mathcal{B}_1^{G,H}] + 1/2 \cdot \Pr[m^* \leftarrow \mathcal{B}_2^{G,H}]$. Hence we obtain

$$4 \cdot 2 \cdot \Pr[m^* \leftarrow \mathcal{B}_3^{G,H}] = \|M_{m^*} |\psi_1\rangle\|^2 + \|M_{m^*} |\psi_0\rangle\|^2. \quad (4)$$

⁴ Roughly speaking, this trick first performs $O_{G,H} := (O_H \otimes |+\rangle\langle +|) + (O_G \otimes |-\rangle\langle -|)$ on $|\psi\rangle|0\rangle$ to obtain the state $1/2(O_H|\psi\rangle - O_G|\psi\rangle)|1\rangle + 1/2(O_H|\psi\rangle + O_G|\psi\rangle)|0\rangle$, and then measures the last qubit with the desired measurement result 1, which makes the whole state to collapse into $O_H|\psi\rangle - O_G|\psi\rangle$ (non-normalized).

For the another sum of the square norms $\|M_{m^*}O_H(U_2)^\dagger M_1^A(|\psi_H\rangle + |\psi_G\rangle)\|^2 + \|M_{m^*}O_H(U_1)^\dagger(I - M_{m^*})O_H(U_2)^\dagger M_1^A(|\psi_H\rangle + |\psi_G\rangle)\|^2$, we first define a one-wayness attacker as:

$\mathcal{B}_4^{G,H,1_S}$: Given oracle access to G , H and 1_S , it works as follows. Here 1_S is the indicator function of $S = \{m^*\}$, that is, $1_S(x) = 1$ if $x = m^*$ and 0 otherwise.

1. Prepare $|\psi_H\rangle + |\psi_G\rangle$ by using the superposition oracle trick [4].
2. Perform the measurement $\mathbb{M}_{\mathcal{A}} = \{M_0^A, M_1^A\}$ with the desired measurement result 1, if the measurement result is 0, abort and output \perp .
3. Apply $O_H(U_2)^\dagger$, then perform projective measurement $\mathbb{M}_{m^*} := \{\chi_0, \chi_1\}$ on the oracle's input register by querying 1_S . Here $\chi_0 = I - M_{m^*}$ and $\chi_1 = M_{m^*}$ ⁵.
 - (a) If the measurement result is 1, measure the oracle's input register to get m^* , then abort and output m^* .
 - (b) If the measurement result is 0, apply $O_H(U_1)^\dagger$ and then perform the measurement \mathbb{M}_{m^*} on the oracle's input register again.
 - i. If the second measurement \mathbb{M}_{m^*} has measurement result 1, measure the oracle's input register to get m^* , then abort and output m^* . Otherwise, abort and output \perp .

Let E_1 be the classical event that the measurement $\mathbb{M}_{\mathcal{A}}$ has result 1 and the next first measurement \mathbb{M}_{m^*} also has result 1. Let E_2 be the classical event that the measurement $\mathbb{M}_{\mathcal{A}}$ has result 1, then the first measurement \mathbb{M}_{m^*} has result 0 and the next second measurement \mathbb{M}_{m^*} has result 1.

At this point, we have a crucial observation that events E_1 and E_2 are mutually exclusive and

$$\begin{aligned} \|M_{m^*}O_H(U_2)^\dagger M_1^A(|\psi_H\rangle + |\psi_G\rangle)\|^2 &= 4 \cdot \Pr[E_1 : \mathcal{B}_4^{G,H,1_S}], \\ \|M_{m^*}O_H(U_1)^\dagger(I - M_{m^*})O_H(U_2)^\dagger M_1^A(|\psi_H\rangle + |\psi_G\rangle)\|^2 &= 4 \cdot \Pr[E_2 : \mathcal{B}_4^{G,H,1_S}]. \end{aligned}$$

Here we have an extra constant factor “4” since our \mathcal{B}_4 uses the superposition oracle trick. Indeed, by the definition, E_1 and E_2 are obviously mutually exclusive. For the $\|M_{m^*}O_H(U_2)^\dagger M_1^A(|\psi_H\rangle + |\psi_G\rangle)\|^2$, the M_1^A and M_{m^*} actually represent that the measurement $\mathbb{M}_{\mathcal{A}}$ and the first measurement \mathbb{M}_{m^*} both have result 1, i.e. E_1 occurs. For the $\|M_{m^*}O_H(U_1)^\dagger(I - M_{m^*})O_H(U_2)^\dagger M_1^A(|\psi_H\rangle + |\psi_G\rangle)\|^2$, the M_1^A represents that the measurement $\mathbb{M}_{\mathcal{A}}$ has result 1, the subsequent $(I - M_{m^*})$ represents that the first measurement \mathbb{M}_{m^*} has result 0, and the final M_{m^*} represents that the second measurement \mathbb{M}_{m^*} has result 1, i.e. E_2 occurs. Consequently, we can compute

⁵ Roughly speaking, to perform $\{I - M_{m^*}, M_{m^*}\}$ on a state $|\phi\rangle := \sum_{x,y} |x,y\rangle$, we first query the oracle 1_S to obtain $\sum_y |m^*, y\rangle|1\rangle + \sum_{x \neq m^*, y} |x,y\rangle|0\rangle$, and then measure the last qubit. If the measurement result is 1, the state $|\phi\rangle$ collapses into $M_{m^*}|\phi\rangle = \sum_y |m^*, y\rangle$ (non-normalized), and we can further measure the first register to get m^* .

$$\begin{aligned}
& 4 \cdot \Pr[m^* \leftarrow \mathcal{B}_4^{G,H,1s}] \stackrel{(a)}{=} 4 \cdot \Pr[E_1 \vee E_2 : \mathcal{B}_4^{G,H,1s}] \\
& \stackrel{(b)}{=} 4 \cdot \Pr[E_1 : \mathcal{B}_4^{G,H,1s}] + 4 \cdot \Pr[E_2 : \mathcal{B}_4^{G,H,1s}] \\
& = \|\mathbb{M}_{m^*} \circ_H (\mathbb{U}_2)^\dagger \mathbb{M}_1^A(|\psi_H\rangle + |\psi_G\rangle)\|^2 + \\
& \quad \|\mathbb{M}_{m^*} \circ_H (\mathbb{U}_1)^\dagger (\mathbb{I} - \mathbb{M}_{m^*}) \circ_H (\mathbb{U}_2)^\dagger \mathbb{M}_1^A(|\psi_H\rangle + |\psi_G\rangle)\|^2.
\end{aligned} \tag{5}$$

Here (a) follows from the definition of our one-wayness attacker \mathcal{B}_4 , (b) uses the fact that events E_1 and E_2 are mutually exclusive. Now, by Eq. (4) and Eq. (5), we can rewrite the $\text{Adv}(\mathcal{A})$ shown in Eq. (3) into

$$\text{Adv}(\mathcal{A}) \leq \sqrt{4 \cdot 2 \cdot \Pr[m^* \leftarrow \mathcal{B}_3^{G,H}]} \cdot \sqrt{4 \cdot \Pr[m^* \leftarrow \mathcal{B}_4^{G,H,1s}]}.$$

Let \mathcal{B} be a one-wayness attacker that runs both \mathcal{B}_3 and \mathcal{B}_4 , outputs m^* if either of these two outputs m^* , and outputs \perp otherwise. Obviously, we have $\max\{\Pr[m^* \leftarrow \mathcal{B}_3^{G,H}], \Pr[m^* \leftarrow \mathcal{B}_4^{G,H,1s}]\} \leq \Pr[m^* \leftarrow \mathcal{B}^{G,H,1s}]$, thus

$$\text{Adv}(\mathcal{A}) \leq 4 \cdot \sqrt{2} \cdot \Pr[m^* \leftarrow \mathcal{B}^{G,H,1s}].$$

That is to say, for the distinguisher \mathcal{A} with query depth $d = 2$, our MRE technique provides an upper bound of $\text{Adv}(\mathcal{A})$ as $4 \cdot \sqrt{d} \cdot \text{Adv}(\mathcal{B})$, where $\text{Adv}(\mathcal{B})$ is the probability that $\mathcal{B}^{G,H,1s}$ successfully finds an element in $S = \{m^*\}$.

Note that \mathcal{B}_4 constructed above has a special structure that first measures (i.e. performs $\mathbb{M}_{\mathcal{A}}$), then rewinds and extracts (i.e. performs *rewinding* operations $\circ_H(\mathbb{U}_2)^\dagger$, $\circ_H(\mathbb{U}_1)^\dagger$ and measurement \mathbb{M}_{m^*} to extract m^*). The same structure is inherited by the final \mathcal{B} since it directly runs \mathcal{B}_4 . Actually, that is precisely why we call our technique described above the Measure-Rewind-Extract (MRE) technique. In addition, when describing our contribution in Sect. 1.1, we mentioned that MRE technique is a combination of the MRM technique [22] and the semi-classical oracle technique [1]. Here, we explain this statement.

- Firstly, our MRE technique follows the framework of MRM technique, which first divides the state $|\psi_H\rangle - |\psi_G\rangle$, then rewrites $\text{Adv}(\mathcal{A})$ into a product of some square norms like Eq. (3), and finally designs a one-wayness attacker \mathcal{B} based on these square norms such that $\text{Adv}(\mathcal{A})$ can be upper bounded by utilizing $\text{Adv}(\mathcal{B})$. However, different from the MRM technique which uses a hybrid argument to divide $|\psi_H\rangle - |\psi_G\rangle$, our MRE technique uses the projector \mathbb{M}_{m^*} to directly divide $|\psi_H\rangle - |\psi_G\rangle$. Note that in the MRM technique, it is this hybrid argument that inevitably introduce a loss of query depth d .
- Secondly, due to using \mathbb{M}_{m^*} to divide $|\psi_H\rangle - |\psi_G\rangle$, we have to construct a one-wayness attacker that performs \mathbb{M}_{m^*} on the oracle's input register, aiming at extracting m^* from \mathcal{A} 's oracle query. In fact, \mathbb{M}_{m^*} is the "semi-classical oracle $\mathcal{O}_{\{m^*\}}^{SC}$ " designed in [1], and the core idea of semi-classical oracle technique is exactly to extract m^* from \mathcal{A} 's oracle query by performing $\mathcal{O}_{\{m^*\}}^{SC}$ (i.e. \mathbb{M}_{m^*}).

Hence, our MRE technique can be viewed as a combination of the MRM technique [22] and the semi-classical oracle technique [1].

Remark 3 (Concern about the measurement \mathbb{M}_{m^}).* Roughly speaking, by using the semi-classical oracle technique, [1] constructed a one-wayness attacker \mathcal{B} and proved that $|\Pr[1 \leftarrow \mathcal{A}^O] - \Pr[1 \leftarrow \mathcal{A}^{O \setminus S}]| \leq \sqrt{O(d)} \cdot \text{Adv}(\mathcal{B})$. In our setting, $O \setminus S$ actually the oracle that first performs the measurement \mathbb{M}_{m^*} on the oracle's input register and then queries O . So this inequality shows that using \mathbb{M}_{m^*} to measure \mathcal{A} 's oracle query will disrupt \mathcal{A} 's computation, resulting in a probability difference of $\sqrt{O(d)} \cdot \text{Adv}(\mathcal{B})$. Note that our one-wayness attacker \mathcal{B}_4 constructed above rewound \mathcal{A} and performed \mathbb{M}_{m^*} , so one might be concerned that \mathcal{B}_4 disrupts \mathcal{A} 's computation and hence will inevitably introduce a loss of $\sqrt{O(d)} \cdot \text{Adv}(\mathcal{B})$. Here, we emphasize that we do not have to concern about this.

- Firstly, our construction of \mathcal{B}_4 does not directly convert \mathcal{A}^O into $\mathcal{A}^{O \setminus S}$. It first runs \mathcal{A}^O , performs \mathcal{A}^O 's final measurement and then rewinds \mathcal{A}^O , and the measurement \mathbb{M}_{m^*} (or $O \setminus S$, intuitively speaking) is only performed during the rewinding. So the inequality $|\Pr[1 \leftarrow \mathcal{A}^O] - \Pr[1 \leftarrow \mathcal{A}^{O \setminus S}]| \leq \sqrt{O(d)} \cdot \text{Adv}(\mathcal{B})$ cannot be directly applied.
- Secondly, what our MRE technique does is, first derive the value

$$p := \|\mathbb{M}_{m^*} O_H(U_2)^\dagger M_1^A(|\psi_H\rangle + |\psi_G\rangle)\|^2 + \|\mathbb{M}_{m^*} O_H(U_1)^\dagger (I - \mathbb{M}_{m^*}) O_H(U_2)^\dagger M_1^A(|\psi_H\rangle + |\psi_G\rangle)\|^2,$$

then construct \mathcal{B}_4 and clearly prove that its success probability $\Pr[m^* \leftarrow \mathcal{B}_4]$ equals to $1/4 \cdot p$ (i.e. Eq. (5)), and finally use this property to prove $\text{Adv}(\mathcal{A}) \leq 4 \cdot \sqrt{d} \cdot \text{Adv}(\mathcal{B})$. That is to say, for our \mathcal{B}_4 , we actually do not care about the ‘‘probability difference’’ between \mathcal{A}^O calculated by \mathcal{B}_4 and the original \mathcal{A}^O , but only focus on whether its success probability $\Pr[m^* \leftarrow \mathcal{B}_4]$ equals to $1/4 \cdot p$. In fact, in Eq. (5), we have clearly calculated that the success probability $\Pr[m^* \leftarrow \mathcal{B}_4]$ of \mathcal{B}_4 is equal to $1/4 \cdot p$, and this calculation, which only utilizes the structure of \mathcal{B}_4 and some basic quantum computation, is actually independent of whether \mathbb{M}_{m^*} disrupts \mathcal{A} 's computation.

Therefore, for our MRE technique, we do not have to concern about performing measurement \mathbb{M}_{m^*} introducing an additional loss of $\sqrt{O(d)} \cdot \text{Adv}(\mathcal{B})$.

Use MRE Technique to Prove Our MRE-O2H Theorem. Although the above explanation of the MRE technique only considers the case where the query depth d is 2, we can directly lift it through induction to account for the case with arbitrary query depth d . Hence, the above explanation of the MRE technique has proved the following fixed version of MRE-O2H theorem.

Theorem 1 (Fixed O2H with MRE, informal). *For a fixed tuple (G, H, S) and a quantum distinguisher \mathcal{A} that makes parallel queries with query depth d , we can construct a quantum one-wayness attacker $\mathcal{B}^{G, H, 1s}$ such that*

$$|\Pr[b = 1 : b \leftarrow \mathcal{A}^G] - \Pr[b = 1 : b \leftarrow \mathcal{A}^H]| \leq 4 \cdot \sqrt{d} \cdot \text{Adv}(\mathcal{B}). \quad (6)$$

Here $\text{Adv}(\mathcal{B})$ is the probability that $\mathcal{B}^{G, H, 1s}$ successfully finds an element in S .

For the random version, where (G, H, S) is sampled from an arbitrary joint distribution D , we can prove it by averaging over $(G, H, S) \leftarrow D$ in Eq. (6).

QROM Security Proofs of FO-Like Transformations. Note that [4, Theorem 5] has shown that FO^\perp (resp. FO^\perp) is as secure as FO_m^\perp (resp. FO_m^\perp) and vice versa. Hence, in our paper, for the FO-like transformations FO^\perp , FO_m^\perp , FO^\perp and FO_m^\perp , we only consider the IND-CCA security of FO^\perp and FO_m^\perp in the QROM.

Our proof outline is shown in Fig. 1. In this outline, we utilize the property that $\text{FO}^\perp = \text{U}^\perp \circ \text{T}$ and $\text{FO}_m^\perp = \text{U}_m^\perp \circ \text{T}$ introduced in [13]. Here, T transforms a randomized PKE (rPKE) scheme into a deterministic PKE (dPKE) scheme, U^\perp and U_m^\perp both transform a dPKE scheme into a KEM scheme.

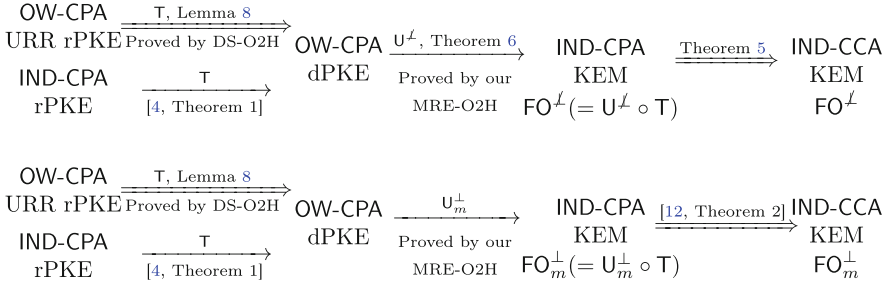


Fig. 1. Proof outline of FO^\perp and FO_m^\perp in the QROM. All the security proofs shown in this outline avoid the square-root advantage loss. The double arrow indicates a tight security proof, while the single arrow indicates a non-tight security proof. In this outline, we abbreviate unique randomness recoverable as URR for the sake of simplicity.

As shown in Fig. 1, by using the DS-O2H theorem [4], we prove that T can tightly transform a OW-CPA-secure and unique randomness recoverable rPKE scheme into a OW-CPA-secure dPKE scheme in the QROM. Based on this proof, we give an IND-CCA security proof of FO^\perp and FO_m^\perp from the OW-CPA security, while avoiding the square-root advantage loss.

In addition, we emphasize that our security proof of U^\perp shown in Fig. 1 (i.e. Theorem 6) does not rely on the η -injective assumption used in [22]. Our observation is that, it is not necessary to require the encryption algorithm dEnc of the underlying dPKE scheme to be nearly injective, we can only need dEnc to satisfy the following weaker property:

For a m^ uniformly sampled from the message space and (pk, sk) generated by the key generation algorithm, there does not exist $m \neq m^*$ such that*

$$\text{dEnc}_{pk}(m) = \text{dEnc}_{pk}(m^*).$$

Indeed, according to [23, Lemma 4], the probability that the underlying dPKE scheme of U^\perp does not satisfy this property is negligible. Hence, we can remove the η -injective assumption in our security proof of U^\perp .

1.3 Related Work

There are also some O2H variants that involve Zhandry’s compressed oracle technique [31]. For example, the [7, Theorem 10], the [21, Theorem C.5], the [14, Theorem 6] and the [12, Theorem 1]. Intuitively speaking, these O2H variants are all obtained by generalizing the SC-O2H theorem [1] to work with the compressed oracle technique. However, they all have a drawback: their final upper bound suffers from the square-root advantage loss.

We note that in [27, 29], the authors proved that the IND-CCA security of the transformation SXY (also known as U_m^ℓ) can be tightly reduced to the DS-IND security of the underlying dPKE scheme in the QROM, where DS-IND is a non-standard security assumption. Indeed, although they provided a tight QROM security proof of U_m^ℓ , which is used to construct the FO-like transformation $FO_m^\ell (= U_m^\ell \circ T)$ [13], the cost is that they used a non-standard security assumption and the underlying PKE scheme is restricted to a dPKE scheme.

In our QROM security proofs of the FO-like transformations, when the security of the underlying PKE scheme is OW-CPA, we introduce an addition assumption named unique randomness recoverable. This assumption is actually a stronger variant of the assumption named randomness recoverable, which, as far as we know, was first introduced in [9, 24] to achieve a tight ROM security proof of the transformation T. According to the definition of unique randomness recoverable given in Definition 6 of our full version [11], we find that it is not a security assumption but just a constraint on the encryption algorithm. Meanwhile, we find that the NTRU-based PKE schemes generally satisfy the assumption of unique randomness recoverable, and we also provide a rough explanation in Supplementary Material C of our full version [11] for completeness.

In a concurrent work, under the assumption that the underlying PKE scheme is unique randomness recoverable, Bao et al. [2] introduced a variant of the DS-O2H theorem [4] and then used it to give a tight security proof of T in the QROM. Their security bound $4 \cdot \epsilon$ is even tighter than the security bound $10 \cdot \epsilon$ achieved by our security proof of T (Lemma 8) in the QROM. Here ϵ is the security bound of the underlying PKE scheme.

2 Preliminaries

2.1 Notation

By $[x = y]$ we denote a bit that is 1 if $x = y$ and otherwise 0. For a finite set S , $x \stackrel{\$}{\leftarrow} S$ denotes that x is an element uniformly sampled from set S . For a distribution D , $x \leftarrow D$ denotes that x is chosen according to distribution D . For a game \mathbf{G} in the security proof, $1 \leftarrow \mathbf{G}$ denotes that \mathbf{G} finally returns 1. $\Pr[A : B, C]$ (or $\Pr_C[A : B]$, $\Pr_{B,C}[A]$ for short) is the probability that the predicate A keeps true where all variables in A are conditioned according to predicates B and C . For an algorithm \mathcal{A} , we use $\mathcal{T}_{\mathcal{A}}$ to denote its running time.

2.2 Quantum Background

We refer to [25] for detailed basics of quantum computation.

2.3 Quantum Random Oracle Model

The random oracle model (ROM) is an ideal model in which a uniformly random function H is selected, and all parties have access to H . In real schemes, the random oracle is implemented using a suitable hash function. In the quantum setting, the ROM should be lifted into the quantum random oracle model (QROM) [5], where all parties have quantum access to the random oracle. In the QROM, we take the random oracle H as a unitary operation $O_H : |x, y\rangle \mapsto |x, y \oplus H(x)\rangle$.

Here, we state the following two lemmas that are used throughout this paper.

Lemma 1 (Simulate the QROM [30, Theorem 6.1]). *Let O be a random oracle, H be a function uniformly sampled from the set of $2q$ -wise independent functions. For any algorithm \mathcal{A} that makes at most q quantum queries, we have*

$$\Pr[b = 1 : b \leftarrow \mathcal{A}^H] = \Pr[b = 1 : b \leftarrow \mathcal{A}^O].$$

Remark 4. This lemma shows that we can use a $2q$ -wise independent function to perfectly simulate a quantum accessible random oracle with query bound q . Indeed, as stated in [27, Section 2.2], this simulation has an $O(q^2)$ running time increase since it has to compute a $2q$ -wise independent function for each query.

Lemma 2 (Generic quantum distinguishing problem with bounded probabilities [15, Lemma 2.9]). *Let $\delta \in [0, 1]$ and \mathcal{M} be a finite set. Let $N_1 : \mathcal{M} \rightarrow \{0, 1\}$ be a random function such that, for each $m \in \mathcal{M}$, $N_1(m) = 1$ with probability δ_m ($\delta_m \leq \delta$), and $N_1(m) = 0$ with probability $1 - \delta_m$. Let $N_2 : \mathcal{M} \rightarrow \{0, 1\}$ be a constant function such that $N_2(m) = 0$ for all $m \in \mathcal{M}$. For any algorithm \mathcal{A} that makes at most q quantum queries, we have*

$$|\Pr[b = 1 : b \leftarrow \mathcal{A}^{N_1}] - \Pr[b = 1 : b \leftarrow \mathcal{A}^{N_2}]| \leq 8(q + 1)^2 \cdot \delta.$$

Now, we recall the Measure-Rewind-Measure One-Way to Hiding (MRM-O2H) theorem introduced in [22].

Theorem 2 (MRM-O2H [22, Lemma 3.3]). *Let $G, H : X \rightarrow Y$ be random functions, $S \subseteq X$ be a random set and $z \in Z$ be a random bitstring. The tuple (G, H, S, z) may have arbitrary joint distribution D and satisfies that $\forall x \notin S$, $G(x) = H(x)$. Let \mathcal{A}^O ($O \in \{G, H\}$) be a quantum oracle algorithm that makes parallel queries with query depth d and query width n . Define*

$$P_{\text{left}} := \Pr_{(G, H, S, z) \leftarrow D} [b = 1 : b \leftarrow \mathcal{A}^H(z)], P_{\text{right}} := \Pr_{(G, H, S, z) \leftarrow D} [b = 1 : b \leftarrow \mathcal{A}^G(z)].$$

Then, we can construct an algorithm $\mathcal{D}^{G, H}(z)$ such that

- Let $\text{Adv}(\mathcal{D}) := \Pr [T_{\mathcal{D}} \cap S \neq \emptyset : T_{\mathcal{D}} \leftarrow \mathcal{D}^{G, H}(z), (G, H, S, z) \leftarrow D]$, then

$$|P_{\text{left}} - P_{\text{right}}| \leq 4d \cdot \text{Adv}(\mathcal{D}).$$

- $\mathcal{D}^{G, H}(z)$ makes parallel queries to G and H both with query depth at most $3d$ and query width n . Its running time can be bounded as $\mathcal{T}_{\mathcal{D}} \lesssim 3 \cdot \mathcal{T}_{\mathcal{A}}$.

3 O2H with Measure-Rewind-Extract (MRE)

In this section, we focus on the tuple (G, H, S, z) , where G, H are functions with domain X and codomain Y , S is a subset of X and G, H, S satisfy that $\forall x \notin S, G(x) = H(x)$, $z \in Z$ is a bitstring that can depend on G, H, S . Let 1_S denote the indicator function of the set S , that is, $1_S(x) = 1$ if $x \in S$ and 0 otherwise.

Here we introduce the following two lemmas that will be used later, and their proofs can be found in Supplementary Material A.3 of our full version [11].

Lemma 3 ([22, Lemma 3.1]). *For any states $|\phi_1\rangle$ and $|\phi_2\rangle$, we have*

$$| \|\phi_1\|^2 - \|\phi_2\|^2 | \leq |(|\phi_1\rangle - |\phi_2\rangle, |\phi_1\rangle + |\phi_2\rangle)|.$$

Lemma 4. *For any states $|\varphi_1\rangle, \dots, |\varphi_n\rangle$ and $|\phi_1\rangle, \dots, |\phi_n\rangle$, we have*

$$\sum_{i=1}^n |(|\varphi_i\rangle, |\phi_i\rangle)| \leq \sqrt{\sum_{i=1}^n \|\varphi_i\|^2} \cdot \sqrt{\sum_{i=1}^n \|\phi_i\|^2}.$$

Now we prove our new O2H theorem. Same as [22], we first prove the fixed version, where the tuple (G, H, S, z) is fixed. Then, we extend it to the random version, where the tuple (G, H, S, z) can have an arbitrary joint distribution.

Theorem 3 (Fixed O2H with MRE). *Let $G, H : X \rightarrow Y$ be fixed functions, $S \subseteq X$ be a fixed set and $z \in Z$ be a fixed bitstring. The tuple (G, H, S, z) satisfies that $\forall x \notin S, G(x) = H(x)$. Let \mathcal{A}^O ($O \in \{G, H\}$) be a quantum oracle algorithm that makes parallel queries with query depth d and query width n . Define*

$$P_{\text{left}}^{GHSz} := \Pr[b = 1 : b \leftarrow \mathcal{A}^H(z)], \quad P_{\text{right}}^{GHSz} := \Pr[b = 1 : b \leftarrow \mathcal{A}^G(z)].$$

Then, we can construct an algorithm $\mathcal{D}^{G,H,1_S}(z)$ which has the following two properties:

- Let $\text{Adv}(\mathcal{D}) := \Pr[T_{\mathcal{D}} \cap S \neq \emptyset : T_{\mathcal{D}} \leftarrow \mathcal{D}^{G,H,1_S}(z)]$, then

$$|P_{\text{left}}^{GHSz} - P_{\text{right}}^{GHSz}| \leq 4\sqrt{d} \cdot \text{Adv}(\mathcal{D}). \quad (7)$$

- $\mathcal{D}^{G,H,1_S}(z)$ makes parallel queries to G, H and 1_S all with query depth at most $3d$ and query width n . Its running time can be bounded as $\mathcal{T}_{\mathcal{D}} \lesssim 3 \cdot \mathcal{T}_{\mathcal{A}}$.

Proof. Following the proof of [22, Lemma 3.2], we denote $O_G^{\otimes n}$ (resp. $O_H^{\otimes n}$) as the n -width parallel quantum oracle for G (resp. H). Then, we define a new quantum oracle

$$O_{G,H}^{\otimes n} := (O_H^{\otimes n} \otimes |+\rangle\langle +|) + (O_G^{\otimes n} \otimes |-\rangle\langle -|).$$

Here $|+\rangle := (|0\rangle + |1\rangle)/\sqrt{2}$ and $|-\rangle := (|0\rangle - |1\rangle)/\sqrt{2}$. Indeed, the oracle $O_{G,H}^{\otimes n}$ uses an auxiliary single quantum bit as the controlling bit: if the state of this controlling bit is $|+\rangle$ (resp. $|-\rangle$), the oracle $O_H^{\otimes n}$ (resp. $O_G^{\otimes n}$) will be queried. As

analyzed in [4, 22], $O_{G,H}^{\otimes n}$ can be efficiently implemented by applying a Hadamard gate before and after a conditional operation, which queries $O_H^{\otimes n}$ (resp. $O_G^{\otimes n}$) if the controlling bit is in the state $|0\rangle$ (resp. $|1\rangle$).

Based on the above notations, we introduce the following lemma that will be used later. It shows that we can use $O_{G,H}^{\otimes n}$ to get a uniform superposition of the sum and difference of the state generated by $O_G^{\otimes n}$ and $O_H^{\otimes n}$, and those states are entangled with the controlling bit of $O_{G,H}^{\otimes n}$. This lemma can be easily proved by induction, and we omit the detailed proof for the sake of simplicity.

Lemma 5. *Let V_1, \dots, V_t ($t \in \mathbb{N}^+$) be any unitary operation that can be applied between $O_G^{\otimes n}/O_H^{\otimes n}$ queries and $|\phi\rangle$ be any appropriate initial state. Let the controlling bit of $O_{G,H}^{\otimes n}$ be in the initial state $|0\rangle$. Then*

$$\begin{aligned} \prod_{i=1}^t [V_i O_{G,H}^{\otimes n} (|\phi\rangle|0\rangle)] &= \frac{1}{2} \left(\prod_{i=1}^t [V_i O_H^{\otimes n} |\phi\rangle] + \prod_{i=1}^t [V_i O_G^{\otimes n} |\phi\rangle] \right) \otimes |0\rangle \\ &\quad + \frac{1}{2} \left(\prod_{i=1}^t [V_i O_H^{\otimes n} |\phi\rangle] - \prod_{i=1}^t [V_i O_G^{\otimes n} |\phi\rangle] \right) \otimes |1\rangle. \end{aligned}$$

Here $\prod_{i=1}^t [V_i O_{G,H}^{\otimes n} (|\phi\rangle|0\rangle)] := V_t O_{G,H}^{\otimes n} V_{t-1} O_{G,H}^{\otimes n} \dots V_2 O_{G,H}^{\otimes n} V_1 O_{G,H}^{\otimes n} (|\phi\rangle|0\rangle)$, and analogously for $\prod_{i=1}^t [V_i O_H^{\otimes n} |\phi\rangle]$ and $\prod_{i=1}^t [V_i O_G^{\otimes n} |\phi\rangle]$.

Since any quantum oracle algorithm can be efficiently transformed into a unitary quantum oracle algorithm with the same query times and query depth (i.e. Fact 1 in Supplementary Material A.1 of our full version [11]), we assume $\mathcal{A}^O(z)$ to be unitary without loss of generality. Now, for $O \in \{G, H\}$, denote $|\psi_z\rangle$ as the initial state of $\mathcal{A}^O(z)$, and denote U_1, \dots, U_d as the unitary operations performed by $\mathcal{A}^O(z)$ between its (parallel) oracle queries. Then, the joint state of $\mathcal{A}^H(z)$ (resp. $\mathcal{A}^G(z)$) just before performing the final binary projective measurement $\mathbb{M}_{\mathcal{A}} := \{M_0^{\mathcal{A}}, M_1^{\mathcal{A}}\}$ can be written as

$$|\psi_H\rangle := \prod_{i=1}^d [U_i O_H^{\otimes n} |\psi_z\rangle] \quad (\text{resp. } |\psi_G\rangle := \prod_{i=1}^d [U_i O_G^{\otimes n} |\psi_z\rangle]). \quad (8)$$

Since the measurement result of $\mathbb{M}_{\mathcal{A}}$ is the final output of \mathcal{A} , we can compute

$$\begin{aligned} |P_{\text{left}}^{GHSz} - P_{\text{right}}^{GHSz}| &= |\Pr[b = 1 : b \leftarrow \mathcal{A}^H(z)] - \Pr[b = 1 : b \leftarrow \mathcal{A}^G(z)]| \\ &= \left| \|M_1^{\mathcal{A}} |\psi_H\rangle\|^2 - \|M_1^{\mathcal{A}} |\psi_G\rangle\|^2 \right| \\ &\stackrel{(a)}{\leq} \left| (M_1^{\mathcal{A}} (|\psi_H\rangle - |\psi_G\rangle), M_1^{\mathcal{A}} (|\psi_H\rangle + |\psi_G\rangle)) \right| \\ &\stackrel{(b)}{=} \left| (|\psi_H\rangle - |\psi_G\rangle, (M_1^{\mathcal{A}})^\dagger M_1^{\mathcal{A}} (|\psi_H\rangle + |\psi_G\rangle)) \right| \\ &\stackrel{(c)}{=} \left| (|\psi_H\rangle - |\psi_G\rangle, M_1^{\mathcal{A}} (|\psi_H\rangle + |\psi_G\rangle)) \right|. \end{aligned} \quad (9)$$

Here (a) is obtained by using Lemma 3. (b) uses the fact that $(A|\phi_1\rangle, B|\phi_2\rangle) = (|\phi_1\rangle, A^\dagger B|\phi_2\rangle)$ for any operators A, B and states $|\phi_1\rangle, |\phi_2\rangle$. (c) uses the fact that the projector $M_1^{\mathcal{A}}$ is Hermitian and idempotent.

Next, we focus on the states $|\psi_H\rangle$ and $|\psi_G\rangle$. We will give them a decomposition (i.e., the following Eq. (13) and Eq. (14)) according to a projector on the oracle's input register. In the following, we first define this projector which we denote as $M_{S^{\oplus n}}$, and then introduce some properties of it.

- **The definition of projector $M_{S^{\oplus n}}$.** Since \mathcal{A} makes parallel queries with query width n , the query state of \mathcal{A} can be written as

$$|query\rangle := \sum_{in, out, aux} \alpha_{in, aux}^{\text{out}} |in_1\rangle |out_1\rangle \cdots |in_n\rangle |out_n\rangle |aux\rangle.$$

Here $in := (in_1, \dots, in_n) \in X^{\otimes n}$, $out := (out_1, \dots, out_n) \in Y^{\otimes n}$ and $aux \in \{0, 1\}^*$. $|in_1\rangle \cdots |in_n\rangle$ (resp. $|out_1\rangle \cdots |out_n\rangle$) is the basis state of the oracle's input register IN (resp. oracle's output register OUT), $|aux\rangle$ is the basis state of some auxiliary registers that may be entangled with IN and OUT . Furthermore, we have

$$\begin{aligned} O_G^{\otimes n} |query\rangle &= \sum_{in, out, aux} \alpha_{in, aux}^{\text{out}} |in_1\rangle |out_1 \oplus G(in_1)\rangle \cdots |in_n\rangle |out_n \oplus G(in_n)\rangle |aux\rangle, \\ O_H^{\otimes n} |query\rangle &= \sum_{in, out, aux} \alpha_{in, aux}^{\text{out}} |in_1\rangle |out_1 \oplus H(in_1)\rangle \cdots |in_n\rangle |out_n \oplus H(in_n)\rangle |aux\rangle. \end{aligned}$$

Define set

$$S^{\oplus n} := \{(in_1, \dots, in_n) | in_1, \dots, in_n \in X, \exists i \in \{1, \dots, n\} \text{ s.t. } in_i \in S\}.$$

Then, we define a projector on the oracle's input register IN as

$$M_{S^{\oplus n}} := \sum_{(in_1, \dots, in_n) \in S^{\oplus n}} |in_1\rangle \cdots |in_n\rangle \langle in_1| \cdots \langle in_n|. \quad (10)$$

Let I_{IN} be the identity operator on the oracle's input register IN , we have

$$I_{IN} - M_{S^{\oplus n}} = \sum_{(in_1, \dots, in_n) \notin S^{\oplus n}} |in_1\rangle \cdots |in_n\rangle \langle in_1| \cdots \langle in_n|.$$

- **Properties satisfied by $M_{S^{\oplus n}}$, $O_G^{\otimes n}$ and $O_H^{\otimes n}$.** Using the fact that $G(x) = H(x)$ for all $x \notin S$, it is easy to see that

$$O_H^{\otimes n} (I_{IN} - M_{S^{\oplus n}}) |query\rangle = O_G^{\otimes n} (I_{IN} - M_{S^{\oplus n}}) |query\rangle. \quad (11)$$

Note that querying the oracles $O_G^{\otimes n}$ and $O_H^{\otimes n}$ does not change the state on the oracle's input register IN , we also have

$$O_G^{\otimes n} M_{S^{\oplus n}} = M_{S^{\oplus n}} O_G^{\otimes n}, \quad O_H^{\otimes n} M_{S^{\oplus n}} = M_{S^{\oplus n}} O_H^{\otimes n}. \quad (12)$$

That is, $O_G^{\otimes n}$ and $O_H^{\otimes n}$ both commute with $M_{S^{\oplus n}}$.

In particular, we introduce the following lemma about $M_{S^{\oplus n}}$. It shows that we can implement the projective measurement $\{I_{IN} - M_{S^{\oplus n}}, M_{S^{\oplus n}}\}$ on the oracle's input register IN by quantum querying the 1_S . The proof of this lemma is very simple and is given in Supplementary Material A.4 of our full version [11].

Lemma 6. *Recall that 1_S is the indicator function of the set S , that is, $1_S(x) = 1$ if $x \in S$ and 0 otherwise. Let $\chi_0 := I_{IN} - M_{S^{\oplus n}}$ and $\chi_1 := M_{S^{\oplus n}}$. Then, the binary projective measurement $M_{S^{\oplus n}} := \{\chi_0, \chi_1\}$ on the oracle's input register IN can be performed by making two parallel queries to 1_S with query width n .*

Now, we define the following states

$$|\psi_H^j\rangle := \prod_{i=1}^j [U_i O_H^{\otimes n}] |\psi_z\rangle \quad (1 \leq j \leq d) \quad \text{and} \quad |\psi_H^0\rangle := |\psi_z\rangle.$$

Let $\prod_{i=1}^0 [U_i O_H^{\otimes n}] := I_{\mathcal{A}}$, where $I_{\mathcal{A}}$ denotes the identity operator on \mathcal{A} 's whole register. Then, for $1 \leq j \leq d$, we can compute

$$\begin{aligned} |\psi_H^j\rangle &= \prod_{i=1}^j [U_i O_H^{\otimes n}] |\psi_z\rangle = U_j O_H^{\otimes n} (M_{S^{\oplus n}} + I_{IN} - M_{S^{\oplus n}}) \prod_{i=1}^{j-1} [U_i O_H^{\otimes n}] |\psi_z\rangle \\ &= U_j O_H^{\otimes n} (M_{S^{\oplus n}} + I_{IN} - M_{S^{\oplus n}}) |\psi_H^{j-1}\rangle \\ &= U_j O_H^{\otimes n} M_{S^{\oplus n}} |\psi_H^{j-1}\rangle + U_j O_H^{\otimes n} (I_{IN} - M_{S^{\oplus n}}) |\psi_H^{j-1}\rangle. \end{aligned}$$

Hence, by induction, it is not hard to obtain

$$\begin{aligned} |\psi_H\rangle = |\psi_H^d\rangle &= \prod_{i=1}^d [U_i O_H^{\otimes n} (I_{IN} - M_{S^{\oplus n}})] |\psi_z\rangle + U_d O_H^{\otimes n} M_{S^{\oplus n}} |\psi_H^{d-1}\rangle \\ &\quad + \sum_{k=1}^{d-1} \prod_{j=k+1}^d [U_j O_H^{\otimes n} (I_{IN} - M_{S^{\oplus n}})] (U_k O_H^{\otimes n} M_{S^{\oplus n}}) |\psi_H^{k-1}\rangle. \end{aligned} \tag{13}$$

Similarly, for $|\psi_G\rangle$, we can derive the following equation using the definitions $|\psi_G^j\rangle := \prod_{i=1}^j [U_i O_G^{\otimes n}] |\psi_z\rangle$ ($1 \leq j \leq d$) and $|\psi_G^0\rangle := |\psi_z\rangle$.

$$\begin{aligned} |\psi_G\rangle = |\psi_G^d\rangle &= \prod_{i=1}^d [U_i O_G^{\otimes n} (I_{IN} - M_{S^{\oplus n}})] |\psi_z\rangle + U_d O_G^{\otimes n} M_{S^{\oplus n}} |\psi_G^{d-1}\rangle \\ &\quad + \sum_{k=1}^{d-1} \prod_{j=k+1}^d [U_j O_G^{\otimes n} (I_{IN} - M_{S^{\oplus n}})] (U_k O_G^{\otimes n} M_{S^{\oplus n}}) |\psi_G^{k-1}\rangle. \end{aligned} \tag{14}$$

Note that I_{IN} denotes the identity operator on the oracle's input register IN , and $I_{\mathcal{A}}$ denotes the identity operator on \mathcal{A} 's whole register. Then, based on the states $|\psi_H^j\rangle$ and $|\psi_G^j\rangle$ ($0 \leq j \leq d$) defined above, we define the following notations that will be used later:

$$\begin{aligned}
 \chi_0 &:= I_{IN} - M_{S^{\oplus n}}, \chi_1 := M_{S^{\oplus n}}, \\
 |\psi_{H+G}\rangle &:= M_1^A(|\psi_H\rangle + |\psi_G\rangle), \\
 |\psi_{H-G}^i\rangle &:= O_H^{\otimes n}|\psi_H^{i-1}\rangle - O_G^{\otimes n}|\psi_G^{i-1}\rangle \quad (1 \leq i \leq d), \\
 |\psi_{H-G}^{S,i}\rangle &:= O_H^{\otimes n}M_{S^{\oplus n}}|\psi_H^{i-1}\rangle - O_G^{\otimes n}M_{S^{\oplus n}}|\psi_G^{i-1}\rangle \\
 &\stackrel{(a)}{=} M_{S^{\oplus n}}O_H^{\otimes n}|\psi_H^{i-1}\rangle - M_{S^{\oplus n}}O_G^{\otimes n}|\psi_G^{i-1}\rangle \quad (1 \leq i \leq d), \\
 U_{d \leftarrow k+1}^{\text{non-}S} &:= \prod_{j=k+1}^d [U_j O_H^{\otimes n} (I_{IN} - M_{S^{\oplus n}})] \quad (1 \leq k \leq d-1), \\
 U_{d \leftarrow d+1}^{\text{non-}S} &:= I_A.
 \end{aligned} \tag{15}$$

Here (a) follows from Eq. (12).

By using Eq. (11), it is not hard to check that

$$\prod_{i=1}^d [U_i O_H^{\otimes n} (I_{IN} - M_{S^{\oplus n}})] |\psi_z\rangle = \prod_{i=1}^d [U_i O_G^{\otimes n} (I_{IN} - M_{S^{\oplus n}})] |\psi_z\rangle. \tag{16}$$

Then, combining Eq. (11) with Eq. (13) to Eq. (16), we obtain $|\psi_H\rangle - |\psi_G\rangle = \sum_{k=1}^d U_{d \leftarrow k+1}^{\text{non-}S} U_k |\psi_{H-G}^{S,k}\rangle$. Combine this equation with Eq. (9), we can compute

$$\begin{aligned}
 &|P_{\text{left}}^{GHSz} - P_{\text{right}}^{GHSz}| \\
 &\leq |(|\psi_H\rangle - |\psi_G\rangle, M_1^A(|\psi_H\rangle + |\psi_G\rangle))| \stackrel{(a)}{=} |(|\psi_H\rangle - |\psi_G\rangle, |\psi_{H+G}\rangle)| \\
 &= \left| \sum_{k=1}^d \left(U_{d \leftarrow k+1}^{\text{non-}S} U_k |\psi_{H-G}^{S,k}\rangle, |\psi_{H+G}\rangle \right) \right| \\
 &\stackrel{(b)}{\leq} \sum_{k=1}^d \left| \left(U_{d \leftarrow k+1}^{\text{non-}S} U_k |\psi_{H-G}^{S,k}\rangle, |\psi_{H+G}\rangle \right) \right| \\
 &\stackrel{(c)}{=} \sum_{k=1}^d \left| \left(|\psi_{H-G}^{S,k}\rangle, (U_k)^\dagger (U_{d \leftarrow k+1}^{\text{non-}S})^\dagger |\psi_{H+G}\rangle \right) \right| \\
 &\stackrel{(d)}{=} \sum_{k=1}^d \left| \left(\chi_1 |\psi_{H-G}^{S,k}\rangle, (U_k)^\dagger (U_{d \leftarrow k+1}^{\text{non-}S})^\dagger |\psi_{H+G}\rangle \right) \right| \\
 &\stackrel{(e)}{=} \sum_{k=1}^d \left| \left(\chi_1 |\psi_{H-G}^{S,k}\rangle, \chi_1 (U_k)^\dagger (U_{d \leftarrow k+1}^{\text{non-}S})^\dagger |\psi_{H+G}\rangle \right) \right| \\
 &\stackrel{(f)}{=} \sum_{k=1}^d \left| \left(|\psi_{H-G}^{S,k}\rangle, \chi_1 (U_k)^\dagger (U_{d \leftarrow k+1}^{\text{non-}S})^\dagger |\psi_{H+G}\rangle \right) \right| \\
 &\stackrel{(g)}{\leq} \sqrt{\sum_{k=1}^d \| |\psi_{H-G}^{S,k}\rangle \|^2} \cdot \sqrt{\sum_{k=1}^d \left\| \chi_1 (U_k)^\dagger (U_{d \leftarrow k+1}^{\text{non-}S})^\dagger |\psi_{H+G}\rangle \right\|^2}.
 \end{aligned} \tag{17}$$

Here (a) follows the definition of $|\psi_{H+G}\rangle$ given in Eq. (15). (b) uses the triangle inequality. (c) uses the basic property of inner product. (d) and (f) use the fact that $|\psi_{H-G}^{S,i}\rangle$ defined in Eq. (15) satisfies $|\psi_{H-G}^{S,i}\rangle = \chi_1 |\psi_{H-G}^{S,i}\rangle$ for $1 \leq i \leq d$. (e) uses the fact that χ_1 is Hermitian and idempotent. (g) uses Lemma 4.

Recall that $|\psi_z\rangle$ is the initial state of $\mathcal{A}^O(z)$ ($O \in \{G, H\}$), U_1, \dots, U_d are the unitary operations performed by $\mathcal{A}^O(z)$ between its parallel oracle queries, and $\mathbb{M}_{\mathcal{A}} = \{\mathbb{M}_0^{\mathcal{A}}, \mathbb{M}_1^{\mathcal{A}}\}$ is the final projective measurement performed by $\mathcal{A}^O(z)$. Now, we define the following algorithms $\mathcal{B}_i^{G,H}(z)$ ($1 \leq i \leq d$), $\mathcal{B}^{G,H}(z)$, $\mathcal{C}^{G,H,1_S}(z)$ and $\mathcal{D}^{G,H,1_S}(z)$, all with the aim to extract an element from the set S :

- Algorithm $\mathcal{B}_i^{G,H}(z)$ ($1 \leq i \leq d$). This algorithm has initial pure state $|\psi_z\rangle|0\rangle$, where $|0\rangle$ is the state of the controlling bit of $O_{G,H}^{\otimes n}$. As mentioned earlier, this controlling bit is used by $O_{G,H}^{\otimes n}$ to determine whether to query $O_G^{\otimes n}$ or $O_H^{\otimes n}$. $\mathcal{B}_i^{G,H}(z)$ first applies $O_{G,H}^{\otimes n} \prod_{j=1}^{i-1} [U_j O_{G,H}^{\otimes n}]$ (here we let $\prod_{j=1}^0 [U_j O_{G,H}^{\otimes n}] = \mathbb{I}_{\mathcal{A}}$), then measures the controlling bit of $O_{G,H}^{\otimes n}$ in the computational basis:
 1. If the measurement result is 1, $\mathcal{B}_i^{G,H}(z)$ measures the oracle's input register IN in the computational basis, and outputs the result $T_{\mathcal{B}_i}$.
 2. If the measurement result is 0, $\mathcal{B}_i^{G,H}(z)$ outputs \perp .
- Algorithm $\mathcal{B}^{G,H}(z)$. This algorithm first uniformly chooses i from $\{1, \dots, d\}$, and then runs $\mathcal{B}_i^{G,H}(z)$ directly. $\mathcal{B}^{G,H}(z)$ finally outputs $\mathcal{B}_i^{G,H}(z)$'s output and we denote it as $T_{\mathcal{B}}$ if it is not \perp .
- Algorithm $\mathcal{C}^{G,H,1_S}(z)$. This algorithm has initial pure state $|\psi_z\rangle|0\rangle$, where $|0\rangle$ is the state of the controlling bit of $O_{G,H}^{\otimes n}$. $\mathcal{C}^{G,H,1_S}(z)$ first applies the operation $\prod_{i=1}^d [U_i O_{G,H}^{\otimes n}]$, then performs the projective measurement $\mathbb{M}_{\mathcal{A}} = \{\mathbb{M}_0^{\mathcal{A}}, \mathbb{M}_1^{\mathcal{A}}\}$ and measures the controlling bit of $O_{G,H}^{\otimes n}$ in the computational basis:
 1. If the measurement result of $\mathbb{M}_{\mathcal{A}}$ is 0 or the measurement result of the controlling bit is 1, $\mathcal{C}^{G,H,1_S}(z)$ outputs \perp .
 2. If the measurement result of $\mathbb{M}_{\mathcal{A}}$ is 1 and the measurement result of the controlling bit is 0, $\mathcal{C}^{G,H,1_S}(z)$ performs the following operations:
 - (a) Initially, let $i = d$.
 - (b) Apply (rewinding operation) $O_H^{\otimes n}(U_i)^\dagger$, then perform the projective measurement $\mathbb{M}_{S^{\oplus n}} = \{\chi_0, \chi_1\}$ on the oracle's input register IN by querying 1_S (Lemma 6). If the measurement result of $\mathbb{M}_{S^{\oplus n}}$ is 1, abort, measure IN in the computational basis and output the result $T_{\mathcal{C}}$.
 - (c) If the measurement result of $\mathbb{M}_{S^{\oplus n}}$ is 0 and $i > 1$, let $i = i - 1$ and repeat the above step. If the measurement result of $\mathbb{M}_{S^{\oplus n}}$ is 0 and $i = 1$, abort and output \perp .
- Algorithm $\mathcal{D}^{G,H,1_S}(z)$. This algorithm runs $\mathcal{B}^{G,H}(z)$ and $\mathcal{C}^{G,H,1_S}(z)$, and it outputs \perp if they both outputs \perp . Otherwise, $\mathcal{D}^{G,H,1_S}(z)$ outputs $T_{\mathcal{D}}$, where
 - $T_{\mathcal{D}} = T_{\mathcal{B}}$ if $\mathcal{B}^{G,H}(z)$ outputs $T_{\mathcal{B}}$ and $\mathcal{C}^{G,H,1_S}(z)$ outputs \perp .

- $T_{\mathcal{D}} = T_{\mathcal{C}}$ if $\mathcal{B}^{G,H}(z)$ outputs \perp and $\mathcal{C}^{G,H,1_S}(z)$ outputs $T_{\mathcal{C}}$.
- $T_{\mathcal{D}} = T_{\mathcal{B}} \cup T_{\mathcal{C}}$ if $\mathcal{B}^{G,H}(z)$ outputs $T_{\mathcal{B}}$ and $\mathcal{C}^{G,H,1_S}(z)$ outputs $T_{\mathcal{C}}$.

As for the running time, one can easily check that $T_{\mathcal{B}} \lesssim T_{\mathcal{A}}$ and $\mathcal{B}^{G,H}(z)$ makes parallel queries to G and H both with query depth at most d and query width n . Since $\mathcal{C}^{G,H,1_S}(z)$ performs the rewinding operations, we have $T_{\mathcal{C}} \lesssim 2 \cdot T_{\mathcal{A}}$ and $\mathcal{C}^{G,H,1_S}(z)$ makes parallel queries to G , H and 1_S all with query depth at most $2d$ and query width n . By the definition of $\mathcal{D}^{G,H,1_S}(z)$, we can conclude that $T_{\mathcal{D}} \lesssim 3 \cdot T_{\mathcal{A}}$, and $\mathcal{D}^{G,H,1_S}(z)$ makes parallel queries to G , H and 1_S all with query depth at most $3d$ and query width n .

We define

$$\begin{aligned} \text{Adv}(\mathcal{B}_i) &:= \Pr[T_{\mathcal{B}_i} \cap S \neq \emptyset : T_{\mathcal{B}_i} \leftarrow \mathcal{B}_i^{G,H}(z)] \quad i \in \{1, \dots, d\}, \\ \text{Adv}(\mathcal{B}) &:= \Pr[T_{\mathcal{B}} \cap S \neq \emptyset : T_{\mathcal{B}} \leftarrow \mathcal{B}^{G,H}(z)], \\ \text{Adv}(\mathcal{C}) &:= \Pr[T_{\mathcal{C}} \cap S \neq \emptyset : T_{\mathcal{C}} \leftarrow \mathcal{C}^{G,H,1_S}(z)], \\ \text{Adv}(\mathcal{D}) &:= \Pr[T_{\mathcal{D}} \cap S \neq \emptyset : T_{\mathcal{D}} \leftarrow \mathcal{D}^{G,H,1_S}(z)]. \end{aligned}$$

For the algorithm $\mathcal{B}_i^{G,H}(z)$ ($i \in \{1, \dots, d\}$), since we let $\prod_{j=1}^0 [U_j O_{G,H}^{\otimes n}] := I_{\mathcal{A}}$, by Lemma 5 and the definition of states $|\psi_{H-G}^i\rangle$ and $|\psi_{H-G}^{S,i}\rangle$ given in Eq. (15), it is not hard to check that

$$\text{Adv}(\mathcal{B}_i) = \left\| \frac{|\psi_{H-G}^{S,i}\rangle}{\|\psi_{H-G}^i\rangle} \right\|^2 \cdot \left\| \frac{1}{2} |\psi_{H-G}^i\rangle \right\|^2 = \frac{1}{4} \cdot \|\psi_{H-G}^{S,i}\rangle\|^2.$$

Then, by the definition of algorithm $\mathcal{B}^{G,H}(z)$, we have

$$\text{Adv}(\mathcal{B}) = \sum_{i=1}^d \frac{1}{d} \text{Adv}(\mathcal{B}_i) = \sum_{i=1}^d \frac{1}{4d} \cdot \|\psi_{H-G}^{S,i}\rangle\|^2. \quad (18)$$

For the algorithm $\mathcal{C}^{G,H,1_S}(z)$, by Lemma 5 and the definition of $|\psi_H\rangle$ and $|\psi_G\rangle$ given in Eq. (8), we can write the state of $\mathcal{C}^{G,H,1_S}(z)$ just before performing the $\mathbb{M}_{\mathcal{A}} = \{M_0^{\mathcal{A}}, M_1^{\mathcal{A}}\}$ and the measurement of the controlling bit of $O_{G,H}^{\otimes n}$ as

$$\frac{1}{2} (|\psi_H\rangle + |\psi_G\rangle) \otimes |0\rangle + \frac{1}{2} (|\psi_H\rangle - |\psi_G\rangle) \otimes |1\rangle.$$

The right half, $|0\rangle$ and $|1\rangle$, is the state of the controlling bit of $O_{G,H}^{\otimes n}$. Since $|\psi_{H+G}\rangle := M_1^{\mathcal{A}}(|\psi_H\rangle + |\psi_G\rangle)$ (i.e. Eq. (15)), the probability that $\mathbb{M}_{\mathcal{A}}$ has result 1 and the measurement of the controlling bit of $O_{G,H}^{\otimes n}$ has result 0 is $\frac{1}{4} \|\psi_{H+G}\rangle\|^2$. Further, the state of $\mathcal{C}^{G,H,1_S}(z)$ will collapse into $|\psi_{H+G}\rangle / \|\psi_{H+G}\rangle\|$ ⁶.

After $\mathbb{M}_{\mathcal{A}}$ obtains result 1 and the measurement of the controlling bit of $O_{G,H}^{\otimes n}$ obtains result 0, $\mathcal{C}^{G,H,1_S}(z)$ will rewind U_1, \dots, U_d and perform $\mathbb{M}_{S \oplus n} = \{\chi_0, \chi_1\}$

⁶ In this notation, we omit the state of the controlling bit of $O_{G,H}^{\otimes n}$, since this bit is no longer used by the subsequent operations of $\mathcal{C}^{G,H,1_S}(z)$.

to extract an element from the set S . We refer to this step of $\mathcal{C}^{G,H,1_S}(z)$ as the “rewind-extract” process, and in fact, we can restate the “rewind-extract” process as:

$$\underline{\mathbb{M}}_{S^{\oplus n}} O_H^{\otimes n}(U_1)^\dagger \underline{\mathbb{M}}_{S^{\oplus n}} O_H^{\otimes n}(U_2)^\dagger \cdots \underline{\mathbb{M}}_{S^{\oplus n}} O_H^{\otimes n}(U_{d-1})^\dagger \underline{\mathbb{M}}_{S^{\oplus n}} O_H^{\otimes n}(U_d)^\dagger \frac{|\psi_{H+G}\rangle}{\|\psi_{H+G}\|}.$$

Here $|\psi_{H+G}\rangle/\|\psi_{H+G}\|$ is the initial pure state just before the “rewind-extract” process, and $\underline{\mathbb{M}}_{S^{\oplus n}}$ denotes the following conditional operation:

Perform $\mathbb{M}_{S^{\oplus n}}$ on the oracle’s input register IN . If the measurement result is 1, measure IN in the computational basis and output the result T_C . If the measurement result is 0, proceed with the subsequent operations.

$\underline{\mathbb{M}}_{S^{\oplus n}}$ is the same as $\mathbb{M}_{S^{\oplus n}}$, except that it directly outputs \perp if the measurement result of $\mathbb{M}_{S^{\oplus n}}$ is 0. Obviously, $\mathcal{C}^{G,H,1_S}(z)$ performs $\mathbb{M}_{S^{\oplus n}}$ at most d times.

Recall that $\chi_1 := \mathbb{M}_{S^{\oplus n}}$ (i.e. Eq. (15)). By the definition of the projector $\mathbb{M}_{S^{\oplus n}}$ given in Eq. (10), we can conclude that T_C must satisfy $T_C \cap S \neq \emptyset$ if T_C is obtained by measuring the oracle’s input register IN (in the computational basis) under the condition that the measurement result of $\mathbb{M}_{S^{\oplus n}}$ just performed was 1. This means that, as long as one of the $\mathbb{M}_{S^{\oplus n}}$ performed by $\mathcal{C}^{G,H,1_S}(z)$ in the “rewinding-extract” process yields a measurement result of 1, $\mathcal{C}^{G,H,1_S}(z)$ will output a set T_C such that $T_C \cap S \neq \emptyset$.

Now, we define the following mutually exclusive events that may be occurred in the “rewinding-extract” process of $\mathcal{C}^{G,H,1_S}(z)$:

E_i : *The measurement result of the first $i - 1$ measurements $\mathbb{M}_{S^{\oplus n}}$ are all 0, and the measurement result of the i -th measurement $\mathbb{M}_{S^{\oplus n}}$ is 1. ($1 \leq i \leq d$)*

According to the definition of the operation $U_{d \leftarrow k+1}^{\text{non-}S}$ ($1 \leq k \leq d$) given in Eq. (15), one can check that

$$\Pr[E_i] = \left\| \chi_1 O_H^{\otimes n}(U_k)^\dagger \left(U_{d \leftarrow k+1}^{\text{non-}S} \right)^\dagger |\psi_{H+G}\rangle \right\|^2 \frac{1}{\|\psi_{H+G}\|^2} \quad (1 \leq i \leq d, i+k = d+1).$$

Then, we can compute

$$\begin{aligned} \text{Adv}(\mathcal{C}) &= \frac{1}{4} \cdot \|\psi_{H+G}\|^2 \cdot \sum_{i=1}^d \Pr[E_i] \\ &= \frac{1}{4} \cdot \|\psi_{H+G}\|^2 \cdot \frac{\sum_{k=1}^d \left\| \chi_1 O_H^{\otimes n}(U_k)^\dagger \left(U_{d \leftarrow k+1}^{\text{non-}S} \right)^\dagger |\psi_{H+G}\rangle \right\|^2}{\|\psi_{H+G}\|^2} \\ &\stackrel{(a)}{=} \frac{1}{4} \cdot \sum_{k=1}^d \left\| O_H^{\otimes n} \chi_1(U_k)^\dagger \left(U_{d \leftarrow k+1}^{\text{non-}S} \right)^\dagger |\psi_{H+G}\rangle \right\|^2 \\ &\stackrel{(b)}{=} \frac{1}{4} \cdot \sum_{k=1}^d \left\| \chi_1(U_k)^\dagger \left(U_{d \leftarrow k+1}^{\text{non-}S} \right)^\dagger |\psi_{H+G}\rangle \right\|^2. \end{aligned} \tag{19}$$

Here (a) follows from the fact that $\chi_1 = M_{S^{\oplus n}}$ and $M_{S^{\oplus n}}$ commutes with $O_H^{\otimes n}$ (i.e. Eq. (12)), (b) uses the fact that $O_H^{\otimes n}$ is a unitary operation.

Combine Eq. (17), Eq. (18) with Eq. (19), we get

$$|P_{\text{left}}^{GHSz} - P_{\text{right}}^{GHSz}| \leq \sqrt{4d \cdot \text{Adv}(\mathcal{B})} \cdot \sqrt{4 \cdot \text{Adv}(\mathcal{C})}.$$

Since we have $\text{Adv}(\mathcal{D}) \geq \max\{\text{Adv}(\mathcal{B}), \text{Adv}(\mathcal{C})\}$ by the definition of the algorithm $\mathcal{D}^{G,H,1_S}(z)$, we finally obtain $|P_{\text{left}}^{GHSz} - P_{\text{right}}^{GHSz}| \leq 4\sqrt{d} \cdot \text{Adv}(\mathcal{D})$. \square

Theorem 4 (Random O2H with MRE). *Let $G, H : X \rightarrow Y$ be random functions, $S \subseteq X$ be a random set and $z \in Z$ be a random bitstring. The tuple (G, H, S, z) may have arbitrary joint distribution D and satisfies that $\forall x \notin S, G(x) = H(x)$. Let \mathcal{A}^O ($O \in \{G, H\}$) be a quantum oracle algorithm that makes parallel queries with query depth d and query width n . Define*

$$P_{\text{left}} := \Pr_{(G,H,S,z) \leftarrow D} [b = 1 : b \leftarrow \mathcal{A}^H(z)], P_{\text{right}} := \Pr_{(G,H,S,z) \leftarrow D} [b = 1 : b \leftarrow \mathcal{A}^G(z)].$$

Then, we can construct an algorithm $\mathcal{D}^{G,H,1_S}(z)$ which has the following two properties:

- Let $\text{Adv}(\mathcal{D}) := \Pr[T_{\mathcal{D}} \cap S \neq \emptyset : T_{\mathcal{D}} \leftarrow \mathcal{D}^{G,H,1_S}(z), (G, H, S, z) \leftarrow D]$, then

$$|P_{\text{left}} - P_{\text{right}}| \leq 4\sqrt{d} \cdot \text{Adv}(\mathcal{D}).$$

- $\mathcal{D}^{G,H,1_S}(z)$ makes parallel queries to G, H and 1_S all with query depth at most $3d$ and query width n . Its running time can be bounded as $\mathcal{T}_{\mathcal{D}} \lesssim 3 \cdot \mathcal{T}_{\mathcal{A}}$.

Proof. Based on Eq. (7) in Theorem 3, we can directly prove this theorem by averaging over $(G, H, S, z) \leftarrow D$. \square

Remark 5. In the proof of Theorem 3, we assume that \mathcal{A} is a unitary quantum oracle algorithm by the well-known fact Fact 1 in Supplementary Material A.1 of our full version [11], which shows that any quantum oracle algorithm can be efficiently transformed into a unitary one with the same query times and query depth. However, as mentioned in [20, 32], that transformation has a linear space⁷ expansion with the running time of the quantum oracle algorithm, since we need to use unitary operations to simulate the non-unitary computations. Indeed, both the MRM-O2H theorem [22] and our MRE-O2H theorem (Theorem 4) involve this linear space expansion. In our paper, we stress that we do not view the space expansion as a dominant factor since it is only linear and not exponential, and we only view the advantage loss and the running time as crucial factors.

⁷ Here the ‘‘space’’ refers to the number of quantum bits used by an algorithm.

4 Tighter IND-CCA Proofs of FO-Like Transformations

In this section, we consider the IND-CCA security of FO-like transformations FO^\perp , FO_m^\perp , FO^\perp and FO_m^\perp in the QROM. Note that [4, Theorem 5] has shown that FO^\perp (resp. FO^\perp) is as secure as FO_m^\perp (resp. FO_m^\perp) and vice versa. Hence, we only prove the IND-CCA security of FO^\perp and FO_m^\perp in the QROM. Our proof idea consists of the following two steps:

1. We first prove that the IND-CCA security of FO^\perp and FO_m^\perp can be reduced to its IND-CPA security.
2. Then, by using our MRE-O2H theorem (Theorem 4), we prove that the IND-CPA security of FO^\perp and FO_m^\perp can be reduced to the IND-CPA/OW-CPA security of the underlying PKE scheme.

The advantage of our proof idea is that, for the FO^\perp , the additional injectivity assumption assumed in the proof of [22, Theorem 4.6] can be removed. All the relevant security notions can be found in Supplementary Material A.2 of our full version [11].

Before proving our results, we first review the transformation T designed in [13] and introduce three lemmas about T that will be used later.

Transformation T : Let $\mathsf{P} = (\text{Gen}, \text{Enc}, \text{Dec})$ be a randomized PKE (rPKE) scheme with message space \mathcal{M} and randomness space \mathcal{R} . Let $H : \mathcal{M} \rightarrow \mathcal{R}$ be a hash function. We associate deterministic PKE (dPKE) scheme $\mathsf{T}[\mathsf{P}, H] := (\text{Gen}, \text{Enc}', \text{Dec}')$. The constituting algorithms of $\mathsf{T}[\mathsf{P}, H]$ are shown in Fig. 2.

Gen	$\text{Enc}'_{pk}(m \in \mathcal{M})$	$\text{Dec}'_{sk}(c)$
1 : $(pk, sk) \leftarrow \text{Gen}$	1 : $c := \text{Enc}_{pk}(m; H(m))$	1 : $m' := \text{Dec}_{sk}(c)$
2 : return (pk, sk)	2 : return c	2 : if $m' = \perp \vee c \neq \text{Enc}_{pk}(m'; H(m'))$
		3 : return \perp
		4 : else return m'

Fig. 2. Deterministic Public Key Encryption $\mathsf{T}[\mathsf{P}, H]$.

Lemma 7 (Security of T from IND-CPA [4, Theorem 1]). *Let P be an rPKE scheme with message space \mathcal{M} . Let \mathcal{A} be a OW-CPA adversary against $\mathsf{T}[\mathsf{P}, H]$, making parallel quantum queries to the random oracle H with query depth d_H and query width n . Let $q_H := d_H \cdot n$.*

Then, we can construct an IND-CPA adversary \mathcal{B} against P such that

$$\text{Adv}_{\mathsf{T}[\mathsf{P}, H]}^{\text{OW-CPA}}(\mathcal{A}) \leq (d_H + 2) \cdot \left(\text{Adv}_{\mathsf{P}}^{\text{IND-CPA}}(\mathcal{B}) + \frac{8(q_H + 1)}{|\mathcal{M}|} \right)$$

and $\mathcal{T}_{\mathcal{B}} \approx \mathcal{T}_{\mathcal{A}} + O(q_H^2)$.

Remark 6. The [4, Theorem 1] actually claims $\mathcal{T}_{\mathcal{B}} \approx \mathcal{T}_{\mathcal{A}}$. In the above lemma, we give a more detailed running time of \mathcal{B} as $\mathcal{T}_{\mathcal{B}} \approx \mathcal{T}_{\mathcal{A}} + O(q_H^2)$. The additional $O(q_H^2)$ is because \mathcal{B} needs to use a $2q_H$ -wise independent function to simulate a quantum accessible random oracle with query bound q_H .

The following lemma also focuses on the OW-CPA security of T in the QROM, but the difference is that the following lemma does not require the underlying rPKE scheme to be IND-CPA-secure.

Lemma 8 (Security of T from OW-CPA). *Let $\mathsf{P} = (\text{Gen}, \text{Enc}, \text{Dec})$ be a δ -correct rPKE scheme, and assume P is unique randomness recoverable with the recover algorithm Rec . Let \mathcal{A} be a OW-CPA adversary against $\mathsf{T}[\mathsf{P}, H]$, making parallel quantum queries to the random oracle H with query depth d_H and query width n . Let $q_H := d_H \cdot n$.*

Then, we can construct a OW-CPA adversary \mathcal{B} against P such that

$$\text{Adv}_{\mathsf{T}[\mathsf{P}, H]}^{\text{OW-CPA}}(\mathcal{A}) \leq 10 \cdot \text{Adv}_{\mathsf{P}}^{\text{OW-CPA}}(\mathcal{B}) + 16 \cdot \delta$$

and $\mathcal{T}_{\mathcal{B}} \approx \mathcal{T}_{\mathcal{A}} + O(q_H^2) + O(q_H) \cdot (\mathcal{T}_{\text{Enc}} + \mathcal{T}_{\text{Rec}})$.

Proof. See Supplementary Material A.5 of our full version [11]. □

Lemma 9 ([23, Lemma 4]). *Let $\mathsf{P} = (\text{Gen}, \text{Enc}, \text{Dec})$ be an rPKE scheme with message space \mathcal{M} and randomness space \mathcal{R} . Define a set w.r.t fixed (pk, sk) and function $H : \mathcal{M} \rightarrow \mathcal{R}$ as*

$$S_{pk, sk, H}^{\text{collision}} := \{m \in \mathcal{M} \mid \exists m' \neq m \text{ s.t. } \text{Enc}_{pk}(m'; H(m')) = \text{Enc}_{pk}(m; H(m))\}.$$

Let Ω_H be the set of all functions $H : \mathcal{M} \rightarrow \mathcal{R}$. Then, if P is δ -correct, we have

$$\Pr \left[m^* \in S_{pk, sk, H}^{\text{collision}} : (pk, sk) \leftarrow \text{Gen}, H \stackrel{\$}{\leftarrow} \Omega_H, m^* \stackrel{\$}{\leftarrow} \mathcal{M} \right] \leq 2 \cdot \delta.$$

4.1 FO-Like Transformation FO^{\perp}

FO-like transformation FO^{\perp} . Let $\mathsf{P} = (\text{Gen}, \text{Enc}, \text{Dec})$ be an rPKE scheme with message space \mathcal{M} , randomness space \mathcal{R} and ciphertext space \mathcal{C} . For a given set \mathcal{K} , let $H : \mathcal{M} \rightarrow \mathcal{R}$, $G : \mathcal{M} \times \mathcal{C} \rightarrow \mathcal{K}$ be hash functions, let $F : \mathcal{K}^{prf} \times \mathcal{C} \rightarrow \mathcal{K}$ be a pseudorandom function (PRF) with key space \mathcal{K}^{prf} . We associate KEM scheme

$$\text{KEM}^{\perp} := \text{FO}^{\perp}[\mathsf{P}, H, G, F] = (\text{Gen}^{\perp}, \text{Enca}, \text{Deca}^{\perp})$$

that has key space \mathcal{K} . The constituting algorithms of KEM^{\perp} are given in Fig. 3.

We first prove the following theorem. It shows that in the QROM, the IND-CPA security of KEM^{\perp} implies its IND-CCA security.

Gen^\perp	$\text{Enca}(pk)$	$\text{Deca}^\perp(sk' = (sk, s), c)$
1: $(pk, sk) \leftarrow \text{Gen}$	1: $m \xleftarrow{\$} \mathcal{M}$	1: $m' := \text{Dec}_{sk}(c)$
2: $s \xleftarrow{\$} \mathcal{K}^{\text{prf}}$	2: $c := \text{Enc}_{pk}(m; H(m))$	2: if $m' = \perp \vee c \neq \text{Enc}_{pk}(m'; H(m'))$
3: $sk' := (sk, s)$	3: $K := G(m, c)$	3: return $K := F(s, c)$
4: return (pk, sk')	4: return (K, c)	4: else return $K := G(m', c)$

Fig. 3. Key Encapsulation Mechanism KEM^\perp .

Theorem 5 (IND-CPA of $\text{KEM}^\perp \stackrel{\text{QROM}}{\Rightarrow}$ IND-CCA of KEM^\perp). *Let $r\text{PKE}$ scheme $\text{P} = (\text{Gen}, \text{Enc}, \text{Dec})$ be δ -correct. Let \mathcal{A} be an IND-CCA adversary against $\text{KEM}^\perp = \text{FO}^\perp[\text{P}, H, G, F]$, making q_D classical queries to the decapsulation oracle, making parallel quantum queries to the random oracle H (resp. G) with query depth d_H (resp. d_G) and query width n . Let $q_H := d_H \cdot n$ and $q_G := d_G \cdot n$.*

Then, we can construct the following two adversaries:

- A PRF-adversary \mathcal{B}_1 against F making at most q_D classical queries. The running time of \mathcal{B}_1 is $\mathcal{T}_{\mathcal{B}_1} \approx \mathcal{T}_A + q_D \cdot (\mathcal{T}_{\text{Enc}} + \mathcal{T}_{\text{Dec}}) + O(q_H^2 + q_G^2)$.
- An IND-CPA adversary \mathcal{B}_2 against KEM^\perp in the QROM. \mathcal{B}_2 makes parallel quantum queries to the random oracle H (resp. G) with query depth at most $d_H + d_G$ (resp. d_G) and query width n . The running time of \mathcal{B}_2 is $\mathcal{T}_{\mathcal{B}_2} \approx \mathcal{T}_A + O(q_G) \cdot \mathcal{T}_{\text{Enc}} + O(q_G^2 + q_D^2)$.

Adversaries \mathcal{B}_1 and \mathcal{B}_2 satisfy the following:

$$\text{Adv}_{\text{KEM}^\perp}^{\text{IND-CCA}}(\mathcal{A}) \leq \text{Adv}_F^{\text{PRF}}(\mathcal{B}_1) + \text{Adv}_{\text{KEM}^\perp}^{\text{IND-CPA}}(\mathcal{B}_2) + 16(2q_H + 2q_G + 1)^2 \cdot \delta.$$

Proof. The proof of this theorem is similar with the proof of [17, Theorem 1], and we present it in Supplementary Material A.6 of our full version [11]. \square

Next, we focus on the IND-CPA security of KEM^\perp in the QROM. As introduced in [13], the KEM^\perp satisfies that

$$\text{KEM}^\perp = \text{FO}^\perp[\text{P}, H, G, F] = \text{U}^\perp[\text{T}[\text{P}, H], G, F]. \quad (20)$$

Here transformation U^\perp transforms a dPKE scheme into a KEM scheme. For the U^\perp , we can prove the following theorem, which shows that in the QROM, the IND-CPA security of U^\perp can be reduced to the OW-CPA security of the underlying dPKE scheme without the square-root advantage loss.

Theorem 6 (OW-CPA of dPKE $\stackrel{\text{QROM}}{\Rightarrow}$ IND-CPA of $\text{U}^\perp[\text{dPKE}, G, F]$). *For a dPKE scheme $\text{dPKE} = (\text{Gen}, \text{dEnc}, \text{dDec})$ with message space \mathcal{M} , let \mathcal{A} be an IND-CPA adversary against $\text{U}^\perp[\text{dPKE}, G, F]$, making parallel quantum queries to the random oracle G with query depth d_G and query width n . Let $q_G := d_G \cdot n$.*

Then, we can construct a OW-CPA adversary \mathcal{A}_1 against dPKE such that

$$\text{Adv}_{\text{U}^\perp[\text{dPKE}, G, F]}^{\text{IND-CPA}}(\mathcal{A}) \leq 2\sqrt{d_G} \cdot \text{Adv}_{\text{dPKE}}^{\text{OW-CPA}}(\mathcal{A}_1) + 2\sqrt{d_G} \cdot \Pr[E_{\text{dPKE}}]$$

and $\mathcal{T}_{\mathcal{A}_1} \lesssim 3 \cdot \mathcal{T}_{\mathcal{A}} + O(q_G) \cdot \mathcal{T}_{\text{dEnc}} + O(q_G^2)$. Here E_{dPKE} is the following event:

$$(pk, sk) \leftarrow \text{Gen}, m^* \stackrel{\S}{\leftarrow} \mathcal{M}, \exists m \neq m^* \text{ such that } \text{dEnc}_{pk}(m) = \text{dEnc}_{pk}(m^*).$$

Proof. We prove this theorem by applying Theorem 4, and we present the detailed proof in Supplementary Material A.8 of our full version [11]. \square

Combining Theorem 6 with Lemma 7 and Lemma 8, we can prove the following result for the IND-CPA security of KEM^\perp in the QROM.

Theorem 7 (IND-CPA/OW-CPA of $\text{P} \stackrel{\text{QROM}}{\Rightarrow} \text{IND-CPA of } \text{KEM}^\perp$). *Let $\text{P} = (\text{Gen}, \text{Enc}, \text{Dec})$ be a δ -correct rPKE scheme with message space \mathcal{M} . Let \mathcal{A} be an IND-CPA adversary against $\text{KEM}^\perp = \text{FO}^\perp[\text{P}, H, G, F]$, making parallel quantum queries to the random oracle H (resp. G) with query depth d_H (resp. d_G) and query width n . Let $q_H := d_H \cdot n$ and $q_G := d_G \cdot n$.*

Then, we can construct an IND-CPA adversary \mathcal{B} against P such that

$$\begin{aligned} \text{Adv}_{\text{KEM}^\perp}^{\text{IND-CPA}}(\mathcal{A}) &\leq 2\sqrt{d_G}(6d_G + d_H + 3) \cdot \text{Adv}_{\text{P}}^{\text{IND-CPA}}(\mathcal{B}) + 4\sqrt{d_G} \cdot \delta \\ &\quad + 16\sqrt{d_G}(6d_G + d_H + 3) \frac{(6q_G + 2q_H + 1)}{|\mathcal{M}|}. \end{aligned}$$

and $\mathcal{T}_{\mathcal{B}} \lesssim 3 \cdot \mathcal{T}_{\mathcal{A}} + O(q_G^2 + q_H^2) + O(q_G) \cdot \mathcal{T}_{\text{Enc}}$. If P is also unique randomness recoverable with the recover algorithm Rec , we can also construct a OW-CPA adversary \mathcal{B}_1 against P such that

$$\text{Adv}_{\text{KEM}^\perp}^{\text{IND-CPA}}(\mathcal{A}) \leq 20\sqrt{d_G} \cdot \text{Adv}_{\text{P}}^{\text{OW-CPA}}(\mathcal{B}_1) + 36\sqrt{d_G} \cdot \delta$$

and $\mathcal{T}_{\mathcal{B}_1} \lesssim 3 \cdot \mathcal{T}_{\mathcal{A}} + O(q_G^2 + q_H^2) + O(q_G + q_H) \cdot (\mathcal{T}_{\text{Enc}} + \mathcal{T}_{\text{Rec}})$.

Proof. This theorem can be easily proved by Eq. (20) and Lemma 9. We present the detailed proof in Supplementary Material A.9 of our full version [11]. \square

Combine Theorem 5 with Theorem 7, we finally obtain the following corollary. It shows that, in the QROM, the IND-CCA security of KEM^\perp can be reduced to the IND-CPA/OW-CPA security of the underlying rPKE scheme without the square-root advantage loss.

Corollary 1 (IND-CPA/OW-CPA of $\text{P} \stackrel{\text{QROM}}{\Rightarrow} \text{IND-CCA of } \text{KEM}^\perp$). *Let $\text{P} = (\text{Gen}, \text{Enc}, \text{Dec})$ be a δ -correct rPKE scheme with message space \mathcal{M} . Let \mathcal{A} be an IND-CCA adversary against $\text{KEM}^\perp = \text{FO}^\perp[\text{P}, H, G, F]$, making q_D classical queries to the decapsulation oracle, making parallel quantum queries to the random oracle H (resp. G) with query depth d_H (resp. d_G) and query width n . Let $q_H := d_H \cdot n$ and $q_G := d_G \cdot n$.*

Then, we can construct the following two adversaries:

- A PRF-adversary \mathcal{B}_1 against F making at most q_D classical queries. The running time of \mathcal{B}_1 is $\mathcal{T}_{\mathcal{B}_1} \lesssim \mathcal{T}_{\mathcal{A}} + q_D \cdot (\mathcal{T}_{\text{Enc}} + \mathcal{T}_{\text{Dec}}) + O(q_H^2 + q_G^2)$.

- An IND-CPA adversary \mathcal{B}_2 against P . The running time of \mathcal{B}_2 is $\mathcal{T}_{\mathcal{B}_2} \lesssim 3 \cdot \mathcal{T}_{\mathcal{A}} + O(q_G) \cdot \mathcal{T}_{\text{Enc}} + O(q_G^2 + q_H^2 + q_D^2)$.

Adversaries \mathcal{B}_1 and \mathcal{B}_2 satisfy the following:

$$\begin{aligned} \text{Adv}_{\text{KEM}^\perp}^{\text{IND-CCA}}(\mathcal{A}) &\leq \text{Adv}_{\mathsf{F}}^{\text{PRF}}(\mathcal{B}_1) + 2\sqrt{d_G}(7d_G + d_H + 3) \cdot \text{Adv}_{\mathsf{P}}^{\text{IND-CPA}}(\mathcal{B}_2) \\ &\quad + 16(2q_H + 2q_G + 1)^2 \cdot \delta + 4\sqrt{d_G} \cdot \delta \\ &\quad + 16\sqrt{d_G}(7d_G + d_H + 3) \frac{(8q_G + 2q_H + 1)}{|\mathcal{M}|}. \end{aligned}$$

If P is also unique randomness recoverable with the recover algorithm Rec , we can also construct following adversary:

- A OW-CPA adversary \mathcal{B}_3 against P . The running time of \mathcal{B}_3 is $\mathcal{T}_{\mathcal{B}_3} \lesssim 3 \cdot \mathcal{T}_{\mathcal{A}} + O(q_G^2 + q_H^2 + q_D^2) + O(q_G + q_H) \cdot (\mathcal{T}_{\text{Enc}} + \mathcal{T}_{\text{Rec}})$.

Adversaries \mathcal{B}_1 and \mathcal{B}_3 satisfy the following:

$$\begin{aligned} \text{Adv}_{\text{KEM}^\perp}^{\text{IND-CCA}}(\mathcal{A}) &\leq \text{Adv}_{\mathsf{F}}^{\text{PRF}}(\mathcal{B}_1) + 20\sqrt{d_G} \cdot \text{Adv}_{\mathsf{P}}^{\text{OW-CPA}}(\mathcal{B}_3) + 36\sqrt{d_G} \cdot \delta \\ &\quad + 16(2q_H + 2q_G + 1)^2 \cdot \delta. \end{aligned}$$

4.2 FO-Like Transformation FO_m^\perp

Similar to Sect. 4.1, we use the following two steps to prove the IND-CCA security of FO_m^\perp in the QROM:

1. First, we introduce [12, Theorem 2], which shows that the IND-CCA security of FO_m^\perp can be reduced to its IND-CPA security.
2. Then, by using our MRE-O2H theorem (Theorem 4), we prove that the IND-CPA security of FO_m^\perp can be reduced to the IND-CPA/OW-CPA security of the underlying PKE scheme.

We present the detailed proofs in Supplementary Material A.10 of our full version [11], and we directly give the main corollary of this section in the following.

Corollary 2 (IND-CPA/OW-CPA of $\mathsf{P} \stackrel{\text{QROM}}{\Rightarrow} \text{IND-CCA of } \text{KEM}_m^\perp$). *Let $r\text{PKE}$ scheme $\mathsf{P} = (\text{Gen}, \text{Enc}, \text{Dec})$ with message space \mathcal{M} be δ -correct and weakly γ -spread. Let \mathcal{A} be an IND-CCA adversary against $\text{KEM}_m^\perp = \text{FO}_m^\perp[\mathsf{P}, H, G]$, making q_D classical queries to the decapsulation oracle, making parallel quantum queries to the random oracle H (resp. G) with query depth d_H (resp. d_G) and query width n . Let $q_H := d_H \cdot n$ and $q_G := d_G \cdot n$.*

Then, we can construct an IND-CPA adversary \mathcal{B}_1 against P such that

$$\begin{aligned} \text{Adv}_{\text{KEM}_m^\perp}^{\text{IND-CCA}}(\mathcal{A}) &\leq 2\sqrt{d_G + q_D}(6d_G + 2d_H + 6q_D + 3) \cdot \text{Adv}_{\mathsf{P}}^{\text{IND-CPA}}(\mathcal{B}_1) \\ &\quad + 4\sqrt{d_G + q_D} \cdot \delta + 8\sqrt{q_H(q_H + 1)} \cdot \delta \\ &\quad + (64q_H + 2) \cdot \delta + 40q_D \cdot 2^{-\gamma/2} \\ &\quad + 16\sqrt{d_G + q_D}(6d_G + 2d_H + 6q_D + 3) \frac{(8q_G + 8q_D + 4q_H + 1)}{|\mathcal{M}|} \end{aligned}$$

and $\mathcal{T}_{\mathcal{B}_1} \lesssim 3 \cdot \mathcal{T}_{\mathcal{A}} + O(q_G) \cdot \mathcal{T}_{\text{Enc}} + O(q_G^2 + q_H^2 + q_H q_D)$. If \mathcal{P} is also unique randomness recoverable with the recover algorithm Rec , we can also construct an OW-CPA adversary \mathcal{B}_2 against \mathcal{P} such that⁸

$$\begin{aligned} \text{Adv}_{\text{KEM}_m^{\text{IND-CCA}}}(\mathcal{A}) \leq & 20\sqrt{d_G + q_D} \cdot \text{Adv}_{\mathcal{P}}^{\text{OW-CPA}}(\mathcal{B}_2) + 36\sqrt{d_G + q_D} \cdot \delta \\ & + 8\sqrt{q_H(q_H + 1)} \cdot \delta + (64q_H + 2) \cdot \delta + 40q_D \cdot |\mathcal{R}|^{-1/2} \end{aligned}$$

and $\mathcal{T}_{\mathcal{B}_2} \lesssim 3 \cdot \mathcal{T}_{\mathcal{A}} + O(q_G^2 + q_H^2 + q_H q_D) + O(q_G + q_H) \cdot (\mathcal{T}_{\text{Enc}} + \mathcal{T}_{\text{Rec}})$.

Acknowledgments. We thank Shujiao Cao, Tianshu Shan, Kexin Gao and Jiawei Bao for their helpful suggestions, and thank all anonymous reviewers of Asiacrypt 2024 for their helpful comments. Particularly, we are very grateful to Reviewer C of Asiacrypt 2024 for providing many insightful suggestions to improve our writing. We will revise our full version [11] according to these suggestions later. This work is supported by the National Natural Science Foundation of China (Grants No. 62172405) and the Key Research Program of the Chinese Academy of Sciences (Grant NO. ZDRW-XX-2022-1).

References

1. Ambainis, A., Hamburg, M., Unruh, D.: Quantum security proofs using semi-classical oracles. In: Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part II. pp. 269–295. Springer (2019). https://doi.org/10.1007/978-3-030-26951-7_10
2. Bao, J., Ge, J., Xue, R.: Double-sided: Tight proofs for guessing games in the quantum random oracle model. unpublished manuscript (2024)
3. Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: CCS '93, Proceedings of the 1st ACM Conference on Computer and Communications Security, Fairfax, Virginia, USA, November 3-5, 1993. pp. 62–73. ACM (1993). <https://doi.org/10.1145/168588.168596>
4. Bindel, N., Hamburg, M., Hövelmanns, K., Hülsing, A., Persichetti, E.: Tighter proofs of CCA security in the quantum random oracle model. In: Theory of Cryptography Conference. pp. 61–90. Springer (2019). https://doi.org/10.1007/978-3-030-36033-7_3
5. Boneh, D., Dagdelen, Ö., Fischlin, M., Lehmann, A., Schaffner, C., Zhandry, M.: Random oracles in a quantum world. In: Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings. pp. 41–69. Springer (2011). https://doi.org/10.1007/978-3-642-25385-0_3
6. Cramer, R., Shoup, V.: Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. SIAM J. Comput. **33**(1), 167–226 (2003). <https://doi.org/10.1137/S0097539702403773>
7. Czajkowski, J., Majenz, C., Schaffner, C., Zur, S.: Quantum lazy sampling and game-playing proofs for quantum indistinguishability. IACR Cryptol. ePrint Arch. p. 428 (2019), <https://eprint.iacr.org/2019/428>

⁸ Here, we implicitly use a property that \mathcal{P} must be weakly $\log_2 |\mathcal{R}|$ -spread if \mathcal{P} is unique randomness recoverable.






8. Dent, A.W.: A designer's guide to kems. In: IMA International Conference on Cryptography and Coding. pp. 133–151. Springer (2003). https://doi.org/10.1007/978-3-540-40974-8_12
9. Duman, J., Hövelmanns, K., Kiltz, E., Lyubashevsky, V., Seiler, G., Unruh, D.: A thorough treatment of highly-efficient NTRU instantiations. In: Public-Key Cryptography - PKC 2023 - 26th IACR International Conference on Practice and Theory of Public-Key Cryptography, Atlanta, GA, USA, May 7-10, 2023, Proceedings, Part I. pp. 65–94. Springer (2023). https://doi.org/10.1007/978-3-031-31368-4_3
10. Fujisaki, E., Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. *J. Cryptol.* **26**(1), 80–101 (2013). <https://doi.org/10.1007/s00145-011-9114-1>
11. Ge, J., Liao, H., Xue, R.: Measure-rewind-extract: Tighter proofs of one-way to hiding and CCA security in the quantum random oracle model. *Cryptology ePrint Archive*, Paper 2024/777 (2024), <https://eprint.iacr.org/2024/777>
12. Ge, J., Shan, T., Xue, R.: Tighter qcca-secure key encapsulation mechanism with explicit rejection in the quantum random oracle model. In: Advances in Cryptology - CRYPTO 2023 - 43rd Annual International Cryptology Conference, CRYPTO 2023, Santa Barbara, CA, USA, August 20-24, 2023, Proceedings, Part V. pp. 292–324. Springer (2023). https://doi.org/10.1007/978-3-031-38554-4_10
13. Hofheinz, D., Hövelmanns, K., Kiltz, E.: A modular analysis of the fujisaki-okamoto transformation. In: Theory of Cryptography Conference. pp. 341–371. Springer (2017). https://doi.org/10.1007/978-3-319-70500-2_12
14. Hövelmanns, K., Hülsing, A., Majenz, C.: Failing gracefully: Decryption failures and the fujisaki-okamoto transform. In: Advances in Cryptology - ASIACRYPT 2022 - 28th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, December 5-9, 2022, Proceedings, Part IV. pp. 414–443. Springer (2022). https://doi.org/10.1007/978-3-031-22972-5_15
15. Hövelmanns, K., Kiltz, E., Schäge, S., Unruh, D.: Generic authenticated key exchange in the quantum random oracle model. In: Public-Key Cryptography - PKC 2020 - 23rd IACR International Conference on Practice and Theory of Public-Key Cryptography, Edinburgh, UK, May 4-7, 2020, Proceedings, Part II. pp. 389–422. Springer (2020). https://doi.org/10.1007/978-3-030-45388-6_14
16. Hövelmanns, K., Majenz, C.: A note on failing gracefully: Completing the picture for explicitly rejecting fujisaki-okamoto transforms using worst-case correctness. *IACR Cryptol. ePrint Arch.* p. 1811 (2023), <https://eprint.iacr.org/2023/1811>
17. Jiang, H., Zhang, Z., Chen, L., Wang, H., Ma, Z.: Ind-cca-secure key encapsulation mechanism in the quantum random oracle model, revisited. In: Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part III. pp. 96–125. Springer (2018). https://doi.org/10.1007/978-3-319-96878-0_4
18. Jiang, H., Zhang, Z., Ma, Z.: Key encapsulation mechanism with explicit rejection in the quantum random oracle model. In: Public-Key Cryptography - PKC 2019 - 22nd IACR International Conference on Practice and Theory of Public-Key Cryptography, Beijing, China, April 14-17, 2019, Proceedings, Part II. pp. 618–645. Springer (2019). https://doi.org/10.1007/978-3-030-17259-6_21
19. Jiang, H., Zhang, Z., Ma, Z.: Tighter security proofs for generic key encapsulation mechanism in the quantum random oracle model. In: Post-Quantum Cryptography - 10th International Conference, PQCrypto 2019, Chongqing, China, May 8-10, 2019 Revised Selected Papers. pp. 227–248. Springer (2019). https://doi.org/10.1007/978-3-030-25510-7_13

20. Jiang, H., Zhang, Z., Ma, Z.: On the non-tightness of measurement-based reductions for key encapsulation mechanism in the quantum random oracle model. In: *Advances in Cryptology - ASIACRYPT 2021 - 27th International Conference on the Theory and Application of Cryptology and Information Security*, Singapore, December 6-10, 2021, Proceedings, Part I. pp. 487–517. Springer (2021). https://doi.org/10.1007/978-3-030-92062-3_17
21. Katsumata, S., Kwiatakowski, K., Pintore, F., Prest, T.: Scalable ciphertext compression techniques for post-quantum kems and their applications. In: *Advances in Cryptology - ASIACRYPT 2020 - 26th International Conference on the Theory and Application of Cryptology and Information Security*, Daejeon, South Korea, December 7-11, 2020, Proceedings, Part I. pp. 289–320. Springer (2020). https://doi.org/10.1007/978-3-030-64837-4_10
22. Kuchta, V., Sakzad, A., Stehlé, D., Steinfeld, R., Sun, S.: Measure-rewind-measure: Tighter quantum random oracle model proofs for one-way to hiding and CCA security. In: *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Zagreb, Croatia, May 10-14, 2020, Proceedings, Part III. pp. 703–728. Springer (2020). https://doi.org/10.1007/978-3-030-45727-3_24
23. Liu, X., Wang, M.: Qcca-secure generic key encapsulation mechanism with tighter security in the quantum random oracle model. In: *Public-Key Cryptography - PKC 2021 - 24th IACR International Conference on Practice and Theory of Public Key Cryptography*, Virtual Event, May 10-13, 2021, Proceedings, Part I. pp. 3–26. Springer (2021). https://doi.org/10.1007/978-3-030-75245-3_1
24. Lyubashevsky, V., Seiler, G.: NTTRU: truly fast NTRU using NTT. *IACR Trans. Cryptogr. Hardw. Embed. Syst.* **2019**(3), 180–201 (2019). <https://doi.org/10.13154/TCHES.V2019.I3.180-201>
25. Nielsen, M.A., Chuang, I.L.: *Quantum Computation and Quantum Information* (10th Anniversary edition). Cambridge University Press (2016)
26. NIST: National institute for standards and technology. post quantum crypto project. <https://csrc.nist.gov/projects/post-quantum-cryptography> (2017)
27. Saito, T., Xagawa, K., Yamakawa, T.: Tightly-secure key-encapsulation mechanism in the quantum random oracle model. In: *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part III. pp. 520–551. Springer (2018). https://doi.org/10.1007/978-3-319-78372-7_17
28. Unruh, D.: Revocable quantum timed-release encryption. *J. ACM* **62**(6), 49:1–49:76 (2015). <https://doi.org/10.1145/2817206>
29. Xagawa, K., Yamakawa, T.: (tightly) qcca-secure key-encapsulation mechanism in the quantum random oracle model. In: *Post-Quantum Cryptography - 10th International Conference, PQCrypto 2019, Chongqing, China, May 8-10, 2019 Revised Selected Papers*. pp. 249–268. Springer (2019). https://doi.org/10.1007/978-3-030-25510-7_14
30. Zhandry, M.: Secure identity-based encryption in the quantum random oracle model. In: *Advances in Cryptology - CRYPTO 2012 - 32nd Annual Cryptology Conference*, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings. pp. 758–775. Springer (2012). https://doi.org/10.1007/978-3-642-32009-5_44

31. Zhandry, M.: How to record quantum queries, and applications to quantum indistinguishability. In: *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference*, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part II. pp. 239–268. Springer (2019). https://doi.org/10.1007/978-3-030-26951-7_9
32. Zhandry, M.: The space-time cost of purifying quantum computations. In: *15th Innovations in Theoretical Computer Science Conference, ITCS 2024*, January 30 to February 2, 2024, Berkeley, CA, USA. pp. 102:1–102:22. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2024). <https://doi.org/10.4230/LIPICS.ITCS.2024.102>



A Tight Security Proof for SPHINCS⁺, Formally Verified

Manuel Barbosa^{1,2} , François Dupressoir³ , Andreas Hülsing^{4,5} ,
Matthias Meijers⁴ , and Pierre-Yves Strub⁶ 

¹ University of Porto (FCUP) and INESC TEC, Porto, Portugal
`fv-sphincsplus@mmeijers.com`

² Max Planck Institute for Security and Privacy, Bochum, Germany

³ University of Bristol, Bristol, UK

⁴ Eindhoven University of Technology, Eindhoven, The Netherlands
`research@mmeijers.com`

⁵ SandboxAQ, Palo Alto, USA

⁶ PQShield, Paris, France

Abstract. SPHINCS⁺ is a post-quantum signature scheme that, at the time of writing, is being standardized as SLH-DSA. It is the most conservative option for post-quantum signatures, but the original tight proofs of security were flawed—as reported by Kudinov, Kiktenko and Fedorov in 2020. In this work, we formally prove a tight security bound for SPHINCS⁺ using the EasyCrypt proof assistant, establishing greater confidence in the general security of the scheme and that of the parameter sets considered for standardization. To this end, we reconstruct the tight security proof presented by Hülsing and Kudinov (in 2022) in a modular way. A small but important part of this effort involves a complex argument relating four different games at once, of a form not yet formalized in EasyCrypt (to the best of our knowledge). We describe our approach to overcoming this major challenge, and develop a general formal verification technique aimed at this type of reasoning.

Enhancing the set of reusable EasyCrypt artifacts previously produced in the formal verification of stateful hash-based cryptographic constructions, we (1) improve and extend the existing libraries for hash functions and (2) develop new libraries for fundamental concepts related to hash-based cryptographic constructions, including Merkle trees. These enhancements, along with the formal verification technique we develop, further ease future formal verification endeavors in EasyCrypt, especially those concerning hash-based cryptographic constructions.

Keywords: SPHINCS⁺ · Post-Quantum Cryptography · EasyCrypt · Formal Verification · Machine-Checked Proofs · Computer-Aided Cryptography

1 Introduction

The advent of sufficiently powerful quantum computers would jeopardize essentially all of the currently deployed public-key cryptography [10]. Albeit it is still uncertain if and when such computers will be practically realized, ongoing advancements and current prospects in the field lead many experts to believe that the likelihood of this happening in the near future is quite substantial [15, 23]. Together with the potentially disastrous ramifications, this suggests that adequate preparation is paramount and urgent. Therefore, in 2016, the National Institute of Standards and Technology (NIST) started a process aimed at the standardization of post-quantum cryptography — cryptography that is executable on classical computers but provides security against attacks from both classical and quantum computers [10, 24]. In 2022, NIST announced the initial four cryptographic constructions to be standardized as a result of this process: CRYSTALS-Kyber for key encapsulation, and CRYSTALS-Dilithium, Falcon, and SPHINCS⁺ for digital signatures [25]. Interestingly, two years prior, NIST already standardized two post-quantum digital signature schemes — XMSS and LMS (as well as their multi-tree variants) — independently from the ongoing standardization process [12]. Although their maturity justified the standardization, these schemes are challenging to deploy in many contexts due to the required state management [12, 22]. Hence, they do not suffice to fully replace contemporary digital signature schemes, which is the rationale for additionally standardizing the schemes from the standardization process.

During the above-mentioned standardization process, Kudinov, Kiktenko and Fedorov discovered an error in the tight security proof for a variant of the Winternitz One-Time Signature (WOTS) scheme, WOTS⁺ [14, 21]. As this scheme is (implicitly) a fundamental component of XMSS and SPHINCS⁺, the tight security proofs for the latter two schemes used similar erroneous reasoning and, hence, were invalid as well [9, 20]. Following this discovery, Hülsing and Kudinov remediated the error for the case of SPHINCS⁺ by explicitly specifying the employed variant of WOTS — called WOTS-TW — defining (and proving) a specific security notion for this variant, and proving the tight security of SPHINCS⁺ using this security notion [17]. Sadly, this approach did not directly translate to the case of XMSS due to the data processed by WOTS-TW being adversarially controlled (while it is user controlled in SPHINCS⁺) [6]. Nevertheless, building on the work by Hülsing and Kudinov [17], Barbosa, Dupressoir, Grégoire, Hülsing, Meijers, and Strub later constructed a novel tight security proof for XMSS; moreover, they formally verified this security proof using the EasyCrypt proof assistant [6]. Unfortunately, in that work, the formal verification of the security proof for SPHINCS⁺ in [17] was considered out of scope and left as future work. Given that the error in the original SPHINCS⁺ security proof was only detected after several years of intense scrutiny, an increase in confidence regarding the novel security proof and its guarantees — as could be accomplished by, e.g., the formal verification of the proof — is no frivolous luxury.

As it is referred to above, formal verification (of cryptography) is an endeavor belonging to the field of computer-aided cryptography. This field aims to address

the ever-increasing complexity of constructing and evaluating cryptography by employing computers to make these processes more rigorous and streamlined [5]. Certainly, this is especially valuable in the context of complex cryptography that is still relatively novel, such as most of the post-quantum cryptography considered for standardization today. Over time, many tools and frameworks have been developed and proven effective in the construction and evaluation of progressively involved and significant cryptographic applications. For instance, as discussed before, EasyCrypt has been used to formally verify the novel security proof for XMSS [6], but also to formally verify the correctness and security of Saber’s Public-Key Encryption (PKE) scheme [18]. Moreover, in combination with Jasmin, EasyCrypt has been used to construct and verify functionally correct, constant-time and efficient implementations of ChaCha20-Poly1305 [1], SHA-3 [3], and the aforementioned CRYSTALS-Kyber [2]. Further examples using different tools include the formal verification of Hybrid Public-Key Encryption (HPKE) using CryptoVerif [4], as well as the formal verification of Transport Layer Security (TLS) 1.3 [13] and (the key establishment of) Signal using Tamarin [11]. A more thorough and systematic overview of computer-aided cryptography with additional examples and success stories is provided in [5].

Our Contribution. In this work, we aim to renew or boost the confidence in the security of (the parameter sets considered for) SPHINCS⁺. Crudely put, we achieve this goal by formally verifying the novel tight security proof for SPHINCS⁺ from [17]. However, we commence this endeavor by reconstructing the entire proof, essentially obtaining a modular version that is significantly more detailed. This reconstruction allows us to somewhat manage the complexity of the formal verification, and reuse some of the artifacts produced in the formal verification of the new tight security proof for XMSS [6]. Nevertheless, the formal verification poses significant, novel challenges that we overcome, including the formal analysis of the considered few-time signature scheme and hypertree structure. Furthermore, one of the modular components we formally verify constitutes a generic relation between variants of the multi-target PREimage resistance (PRE), Target Collision Resistance (TCR), and Decisional Second-Preimage Resistance (DSPR) properties. This statement is comparable to Theorem 38 in [8], the proof of which employs non-standard reasoning. Correspondingly, the proof for the statement we consider is similarly non-standard. Loosely speaking, instead of utilizing a standard approach such as (a sequence of) reductions between pairs of games, this proof simultaneously compares four games through an extremely granular case analysis on the associated success probabilities. In the process of understanding and developing a proof technique aimed at this kind of reasoning, we formally verify the simpler of the fundamental theorems in [8], Theorem 25, that relates the (standard) PRE, SPR, and DSPR properties — allowing us to try out the arguments in a simpler context.

Due to the nature of the considered proof and the artifacts we build on, we opt to employ EasyCrypt — a powerful and expressive tool primarily aimed at the formal verification of code-based, game-playing security proofs in the computational model [7] — for this work. As part of this work’s contribution,

we facilitate future formal verification endeavors in two ways. First, we extend EasyCrypt by creating and enhancing libraries based on the features required in this work. Specifically, we construct libraries containing (generic) definitions and properties for binary trees and Merkle trees; furthermore, we enhance the libraries for hash functions — originally produced in [6] — by adding new properties and adjusting some of the definitions to be easier to use in different scenarios. Second, we develop a general formal verification technique targeting the type of non-standard reasoning required for the proof of the aforementioned relation between (variants of) the PRE, TCR, and DSPR properties. To the best of our knowledge, this is a novelty in the context of EasyCrypt. Although this paper only covers some of these artifacts in more detail, all of them can be found in the repository associated with this work, located at <https://github.com/MM45/FV-SPHINCSPLUS-EC>, or in the standard library of EasyCrypt.

Overview. The remainder of this paper is organized as follows. First, Section 2 introduces the fundamental structure and concepts underlying SPHINCS⁺ and its formal verification. Second, Section 3 provides an overview of the formal verification. Lastly, Section 4, 5, and 6 discuss several aspects of the formal verification in detail.

2 Preliminaries

In the ensuing, we introduce the schemes and concepts considered throughout the paper. Most of the fundamentals directly coincide with those of previous works [6,9]; however, we still provide them here for completeness.

SPHINCS⁺. On a high level, a SPHINCS⁺ instance consists of (1) an instance of a hypertree-based signature scheme akin to XMSS^{MT} [16,19], and (2) an instance of a forest-based — i.e., considering a sequence of individual trees — signature scheme for each leaf of the hypertree. This latter scheme is a few-time signature scheme, called Forest of Random Subsets (FORS), which was introduced together with SPHINCS⁺ [9]. In the hypertree construction, each “node” constitutes an instance of a Merkle signature scheme similar to XMSS with WOTS-TW, a variant of WOTS introduced in [17], as One-Time Signature (OTS) scheme. To sign an arbitrary-length message m with SPHINCS⁺, the message is initially processed in a way that results in a fixed-length message m_c and an index i pointing to a leaf of the hypertree. Subsequently, the FORS instance associated with this leaf is used to sign m_c ; in turn, the hypertree construction is used to sign the public key of this FORS instance. Then, the SPHINCS⁺ signature on m consists of the information used to obtain m_c and i from m , the FORS signature on m_c , and the hypertree signature on the public key of the employed FORS instance. Intuitively, the FORS signature can be seen as the actual signature on the message, while the signature of the hypertree construction can be seen as a proof that the FORS instance used to sign the message is actually part of the considered SPHINCS⁺ instance.

$\text{Game}_{\mathcal{A}, \text{KHF}, \text{MAP}}^{\text{ITSR}}$ <hr/> 1 : OITSR.Init() 2 : $(k, x) \leftarrow \mathcal{A}^{\text{OITSR.Query}}.\text{Find}()$ 3 : return $\text{IB}_{\text{KHF}}^{\text{MAP}}(k, x, \text{OITSR}_{\text{KHF}}.\mathcal{T})$
--

Fig. 1. ITSR game.

OITSR <hr/> vars \mathcal{T} Init() <hr/> 1 : $\mathcal{T} \leftarrow []$ Query (x) <hr/> 1 : $k \leftarrow \mathcal{U}(\mathcal{K})$ 2 : $\mathcal{T} \leftarrow \mathcal{T} \parallel (k, x)$ 3 : return k

Fig. 2. Oracle employed in ITSR game.

Keyed Hash Functions. A Keyed Hash Function (KHF) is a function $\text{KHF} : \mathcal{K} \times \mathcal{M} \rightarrow \mathcal{Y}$ where *key space* \mathcal{K} , *message space* \mathcal{M} , and *digest space* \mathcal{Y} respectively denote sets of keys, messages, and digests. In practice, these spaces are essentially all sets of bitstrings. However, in specifications, each of these spaces may also be left abstract or be instantiated with any set relevant in the considered context — e.g., the set of integers within a certain range. Occasionally, we interpret and refer to a KHF as a family of hash functions indexed by keys from the key space.

For KHFs, we consider the *Interleaved Target Subset Resilience* (ITSR) and *Pseudo-Random Function family* (PRF) properties. Intuitively, a KHF is a PRF if querying an unknown, randomly selected hash function from the family defined by the KHF is computationally indistinguishable from querying an actual random function.¹ Formally, the ITSR and PRF properties for KHFs are respectively defined as the games in Figures 1 and 3; the oracles employed in these games are specified in Figures 2 and 4. In the ITSR game, $\text{IB}_{\text{KHF}}^{\text{MAP}}$ is a predicate validating whether its arguments constitute an ITSR break (with respect to KHF and MAP); more precisely, this predicate is defined as follows.

$$\text{IB}_{\text{KHF}}^{\text{MAP}}(k, x, \mathcal{T}) = (k, x) \notin \mathcal{T} \wedge \text{MAP}(\text{KHF}(k, x)) \in \bigcup_{i=0}^{|\mathcal{T}|-1} \text{MAP}(\text{KHF}(\mathcal{T}[i][0], \mathcal{T}[i][1]))$$

Then, the advantage of any adversary \mathcal{A} against ITSR is defined as given below.

$$\text{Adv}_{\text{KHF}, \text{MAP}}^{\text{ITSR}}(\mathcal{A}) = \Pr \left[\text{Game}_{\mathcal{A}, \text{KHF}, \text{MAP}}^{\text{ITSR}} = 1 \right]$$

Moreover, the advantage of any adversary \mathcal{A} against PRF is defined as follows.

$$\text{Adv}_{\text{KHF}}^{\text{PRF}}(\mathcal{A}) = \left| \Pr \left[\text{Game}_{\mathcal{A}, \text{KHF}}^{\text{PRF}}(0) = 1 \right] - \Pr \left[\text{Game}_{\mathcal{A}, \text{KHF}}^{\text{PRF}}(1) = 1 \right] \right|$$

Tweakable Hash Functions. A Tweakable Hash Function (THF) is a function $\text{THF} : \mathcal{P} \times \mathcal{T} \times \mathcal{M} \rightarrow \mathcal{Y}$ where (*public*) *parameter space* \mathcal{P} , *tweak space* \mathcal{T} , *message space* \mathcal{M} , and *digest space* \mathcal{Y} denote sets of (*public*) parameters, tweaks, messages, and digests, respectively. As for KHFs, in practice, these spaces are

¹ Unlike the PRF property, the ITSR property is specifically designed for SPHINCS⁺ and does not admit as much of an intuitive interpretation out of context.

$\text{Game}_{\mathcal{A}, \text{KHF}}^{\text{PRF}}(b)$
1 : $\text{OPRF}_{\text{KHF}}.\text{Init}(b)$ 2 : $b' \leftarrow \mathcal{A}^{\text{OPRF}_{\text{KHF}}.\text{Query}}.\text{Distinguish}()$ 3 : return b'

Fig. 3. PRF game.

OPRF_{KHF}
vars b, k, m Init (bi)
1 : $b, m \leftarrow \text{bi}, \text{emptymap}$ 2 : $k \leftarrow \mathcal{U}(\mathcal{K})$
Query (x)
1 : if b then 2 : if $m.[x] = \perp$ then 3 : $y \leftarrow \mathcal{U}(\mathcal{Y})$ 4 : $m.[x] \leftarrow y$ 5 : $y \leftarrow m.[x]$ 6 : else 7 : $y \leftarrow \text{KHF}(k, x)$ 8 : return y

Fig. 4. Oracle employed in PRF game.

essentially all sets of bitstrings. In specifications, they may also be left abstract or be instantiated with any set relevant in the considered context. Nevertheless, throughout this work, the message and digest space of any THF are, respectively, the set of arbitrary-length bitstrings (i.e., $\{0, 1\}^*$) and a set of fixed-length bitstrings (i.e., $\{0, 1\}^k$ for some $k > 0$). Conceptually, THFs extend KHF by explicitly considering contextual data in the form of tweaks, primarily serving the purpose of mitigating multi-target attacks. At times, we view and refer to a THF as a family of hash functions (mapping tweaks and messages to digests) indexed by (public) parameters from the (public) parameter space.

Alongside individual THFs, we consider collections of such functions — a concept introduced by the authors of SPHINCS⁺ [9] — containing a single THF for each possible length of the input messages. Alternatively stated, a collection of THFs constitutes a set $\text{THFC} = \{\text{THF}_\lambda : \mathcal{P} \times \mathcal{T} \times \mathcal{M} \rightarrow \mathcal{Y}\}_{\lambda \in \Lambda}$ where Λ is the index set comprising the possible input lengths.²

For THFs, the properties we are concerned with in this work are the *Single-function*, *Multi-target*, *Distinct-Tweak* variants of *Target-Collision Resistance* (SM-DT-TCR), *Decisional Second-Preimage Resistance* (SM-DT-DSPR), and *Opening-Preimage Resistance* (SM-DT-OpenPRE). Additionally, we consider an extension of SM-DT-TCR, denoted by SM-DT-TCR-C, that takes the relevant THF collection into account. As their names suggest, all of these properties model a similar scenario where (1) a single, uniformly random (public) parameter is considered throughout (*single-function*); (2) the attack’s targets, of which there may be multiple (*multi-target*), must be specified before the parameter is revealed; and (3) the tweaks used in the attack’s targets must be

² Technically, we could restrict the message space of each THF in a collection to only contain messages of the relevant length, but this does not yield significant advantages.

Game ^{SM-DT-TCR(-C)} _{$\mathcal{A}, \mathcal{T}, \mathcal{X}, \text{THFC}, t$}
1 : $p \leftarrow \mathcal{U}(\mathcal{P})$
2 : $\text{OTCR}_{\text{THF}}.\text{Init}(p)$
3 : $\text{OC}_{\text{THFC}}.\text{Init}(p)$
4 : $\mathcal{A}^{\text{OTCR}_{\text{THF}}.\text{Query}, \text{OC}_{\text{THFC}}.\text{Query}}.\text{Pick}()$
5 : $i, x' \leftarrow \mathcal{A}.\text{Find}(p)$
6 : $\text{tw}, x \leftarrow \text{OTCR}_{\text{THF}}.\mathcal{T}[i], \text{OTCR}_{\text{THF}}.\mathcal{X}[i]$
7 : return $x \neq x' \wedge \text{THF}(p, \text{tw}, x) = \text{THF}(p, \text{tw}, x')$ $\wedge \text{VQS}_t(i, \text{OTCR}_{\text{THF}}.\mathcal{T}, \text{OC}_{\text{THFC}}.\mathcal{T})$

Fig. 5. SM-DT-TCR(-C) game. Outlined code is only considered in SM-DT-TCR-C.

OTCR _{THF}
vars $p, \mathcal{T}, \mathcal{X}$
Init(pi)
1 : $p, \mathcal{T}, \mathcal{X} \leftarrow \text{pi}, [], []$
Query(tw, x)
1 : $y \leftarrow \text{THF}(p, \text{tw}, x)$
2 : $\mathcal{T}, \mathcal{X} \leftarrow \mathcal{T} \parallel \text{tw}, \mathcal{X} \parallel x$
3 : return y

Fig. 6. Challenge oracle employed in SM-DT-TCR(-C) game.

OC _{THFC}
vars p, \mathcal{T}
Init(pi)
1 : $p, \mathcal{T} \leftarrow \text{pi}, []$
Query(tw, x)
1 : $y \leftarrow \text{THFC}_{ x }(p, \text{tw}, x)$
2 : $\mathcal{T} \leftarrow \mathcal{T} \parallel \text{tw}$
3 : return y

Fig. 7. Collection oracle employed in games for tweakable hash functions.

distinct (*distinct-tweak*). Unsurprisingly, this scenario shares quite some similarities with the manner in which SPHINCS⁺ operates; in particular, SPHINCS⁺ uses the same (public) parameter — which is sampled uniformly at random during setup — and a unique tweak for each THF evaluation. For a more in-depth discussion and analysis of these properties, see [9, 17].

The considered THF properties are formalized through the games and oracles in Figures 5, 6, and 7 (SM-DT-TCR(-C) game, challenge oracle, and collection oracle); Figures 8 and 9 (SM-DT-DSPR games and challenge oracle); and Figures 10 and 11 (SM-DT-OpenPRE game and challenge oracle). In these games, t denotes the upper bound on the number of targets, SPE_{THF} is a predicate that indicates whether there exists a second-preimage of the given message under THF (when the first two arguments to THF are the given parameter and tweak), and VQS_t is a predicate that validates the adversary’s behavior by checking whether (1) the number of targets is less than or equal to t , (2) the provided index i is a valid index into the target list(s), and (3) the target tweaks are distinct from each other and, in case the relevant collection is considered, from the tweaks issued to the collection oracle. Moreover, for the (non-standard) advantage definition of SM-DT-DSPR, we need to define SM-DT-SPprob, a game that essentially repre-

$\text{Game}_{\mathcal{A}, \text{THF}, t}^{\text{SM-DT-DSPR}-\text{SPprob}}$ <hr/> 1: $p \leftarrow \$ \mathcal{U}(\mathcal{P})$ 2: $\text{ODSPR}_{\text{THF}}.\text{Init}(p)$ 3: $\mathcal{A}^{\text{ODSPR}_{\text{THF}}.\text{Query}}.\text{Pick}()$ 4: $i, b \leftarrow \mathcal{A}.\text{Find}(p)$ 5: $\text{tw}, x \leftarrow \text{ODSPR}_{\text{THF}}.\mathcal{T}[i], \text{ODSPR}_{\text{THF}}.\mathcal{X}[i]$ 6: return $\text{SPE}_{\text{THF}}(p, \text{tw}, x) \stackrel{\text{b}}{=} \bar{b}$ $\wedge \text{VQS}_t(i, \text{ODSPR}_{\text{THF}}.\mathcal{T})$

Fig. 8. SM-DT-DSPR (blue) and SM-DT-SPprob (yellow) game. Non-outlined code is considered in both games. (Color figure online)

$\text{ODSPR}_{\text{THF}}$ <hr/> vars $p, \mathcal{T}, \mathcal{X}$ Init (pi) <hr/> 1: $p, \mathcal{T}, \mathcal{X} \leftarrow \text{pi}, [], []$ Query (tw, x) <hr/> 1: $y \leftarrow \text{THF}(p, \text{tw}, x)$ 2: $\mathcal{T}, \mathcal{X} \leftarrow \mathcal{T} \parallel \text{tw}, \mathcal{X} \parallel x$ 3: return y
--

Fig. 9. Challenge oracle employed in SM-DT-DSPR and SM-DT-SPprob game.

$\text{Game}_{\mathcal{A}, \text{THF}, t}^{\text{SM-DT-OpenPRE}}$ <hr/> 1: $p \leftarrow \$ \mathcal{U}(\mathcal{P})$ 2: $\text{tws} \leftarrow \mathcal{A}.\text{Pick}()$ 3: $\text{ys} \leftarrow \text{OOPRE}_{\text{THF}}.\text{Init}(p, \text{tws})$ 4: $i, x \leftarrow \mathcal{A}^{\text{OOPRE}_{\text{THF}}.\text{Open}}.\text{Find}(p, \text{ys})$ 5: $\text{tw}, y \leftarrow \text{tws}[i], \text{ys}[i]$ 6: return $\text{THF}(p, \text{tw}, x) = y$ $\wedge i \notin \text{OOPRE}_{\text{THF}}.\mathcal{O}$ $\wedge \text{VQS}_t(i, \text{tws})$
--

Fig. 10. SM-DT-OpenPRE game for tweakable hash functions.

$\text{OOPRE}_{\text{THF}}$ <hr/> vars $p, \mathcal{X}, \mathcal{O}$ Init (pi, twsi) <hr/> 1: $p, \mathcal{X}, \mathcal{O}, \text{ys} \leftarrow \text{pi}, [], [], []$ 2: for tw in twsi do 3: $x \leftarrow \$ \mathcal{U}(\mathcal{M})$ 4: $\mathcal{X} \leftarrow \mathcal{X} \parallel x$ 5: $\text{ys} \leftarrow \text{ys} \parallel \text{THF}(p, \text{tw}, x)$ 6: return ys <hr/> Open (i) <hr/> 1: $\mathcal{O} \leftarrow \mathcal{O} \parallel i$ 2: return $\mathcal{X}[i]$
--

Fig. 11. Challenge oracle employed in SM-DT-OpenPRE game.

sents the trivial attack against SM-DT-DSPR. Then, we define the advantage of any adversary \mathcal{A} against $\text{Prop} \in \{\text{SM-DT-TCR}, \text{SM-DT-OpenPRE}\}$ as follows.

$$\text{Adv}_{\text{THF}, t}^{\text{Prop}}(\mathcal{A}) = \Pr \left[\text{Game}_{\mathcal{A}, \text{THF}, t}^{\text{Prop}} = 1 \right]$$

For the remaining THF properties, the corresponding advantages are given below, where $p = \Pr \left[\text{Game}_{\mathcal{A}, \text{THF}, t}^{\text{SM-DT-DSPR}} = 1 \right]$ and $q = \Pr \left[\text{Game}_{\mathcal{A}, \text{THF}, t}^{\text{SM-DT-SPprob}} = 1 \right]$.

$$\text{Adv}_{\text{THF}, t}^{\text{SM-DT-DSPR}}(\mathcal{A}) = \max(0, p - q)$$

$$\text{Adv}_{\text{THF}, \text{THFC}, t}^{\text{SM-DT-TCR-C}}(\mathcal{A}) = \Pr \left[\text{Game}_{\mathcal{A}, \text{THF}, \text{THFC}, t}^{\text{SM-DT-TCR-C}} = 1 \right]$$

Hash Addresses. An instance of SPHINCS⁺ employs the same collection of THFs throughout its entire execution; furthermore, it invariably uses the same

(public) parameter to index the THFs. Thus, to mitigate multi-target attacks, SPHINCS⁺ uses a unique, fixed tweak in each THF evaluation. For the construction of these tweaks, SPHINCS⁺ utilizes a specific addressing scheme. In this scheme, an address essentially encodes (uniquely) identifying information for the THF evaluation in which the address is used. More precisely, each address constitutes a fixed-length sequence of nonnegative integers encoding the location and purpose of a THF evaluation within the virtual structure of a SPHINCS⁺ instance. Naturally, not every (fixed-length) sequence of nonnegative integers constitutes a valid address in this scheme. Furthermore, because we approach the analysis of SPHINCS⁺ in a modular manner, parts of the addresses may be irrelevant at certain points;³ in such cases, we disregard the irrelevant part of the addresses. Throughout this paper, we use “address” to refer to a fixed-length sequence of nonnegative integers that constitutes (the relevant part of) a valid SPHINCS⁺ address in the considered context. Additional clarification on address validity will be provided as necessary.

EasyCrypt. EasyCrypt is an interactive proof assistant designed to check the validity of code-based, game-playing proofs for concrete security statements in the computational model. Schemes, experiments/games, oracles, and reductions are all expressed as probabilistic programs written in a simple probabilistic `While` language, and claims (e.g., about successive games) are proved by demonstrating equivalences between such programs — if necessary, quantifying over an additional program modeling the adversary. Moreover, it is possible to quantify over programs modeling parts of the (honest) system that are left abstract; this can, for example, be used to state and prove results about generic compositions or transformations.

In EasyCrypt, an equivalence proof is typically performed by establishing a probabilistic coupling between the processes defined by the considered programs, rather than reasoning directly about the distributions those processes sample from. The relational (principal) logic supports conditional equivalences (capturing simulation failures), and can be combined with a non-relational logic to bound the computational distance between two imperfectly equivalent games.⁴ In cases where an equivalence cannot be reasoned about by coupling — e.g., when a sampling operation is split into two or more sampling operations — EasyCrypt supports reasoning directly about distributions, as long as the distributions are fully defined by the programs without adversarial interaction.

3 Approach

The primary objective of this work is to renew or increase confidence in the (parameter sets considered for) SPHINCS⁺, which we achieve via the formal

³ For example, in a modular part that exclusively operates on a single layer of the virtual structure, the part of the addresses that indicates this layer is irrelevant.

⁴ Here, “relational” refers to simultaneous reasoning about two programs; correspondingly, “non-relational” refers to reasoning about a single program.

verification of a tight security proof. To this end, we initially reconstruct the (handwritten) tight security proof for SPHINCS⁺ from [17] in a modular manner, adding a significant amount of detail in the process. Utilizing this reconstructed proof as a guideline for the subsequent formal verification facilitates the overall process in several ways. First, the modularity reduces the complexity (of the formal verification) of individual statements by limiting their scope. Second, the modularity enables the reuse of certain artifacts previously produced in the formal verification of the novel tight security proof for XMSS [6]. Lastly, the increased granularity is a prerequisite for the formal verification, where each reasoning step must be carried out explicitly and in full.

Figure 12 presents a high-level overview of (the proofs underlying) our formal verification. In this figure, each node represents a property of a cryptographic construction, KHF, or THF; each edge indicates an implication between properties, i.e., from origin nodes to destination nodes.

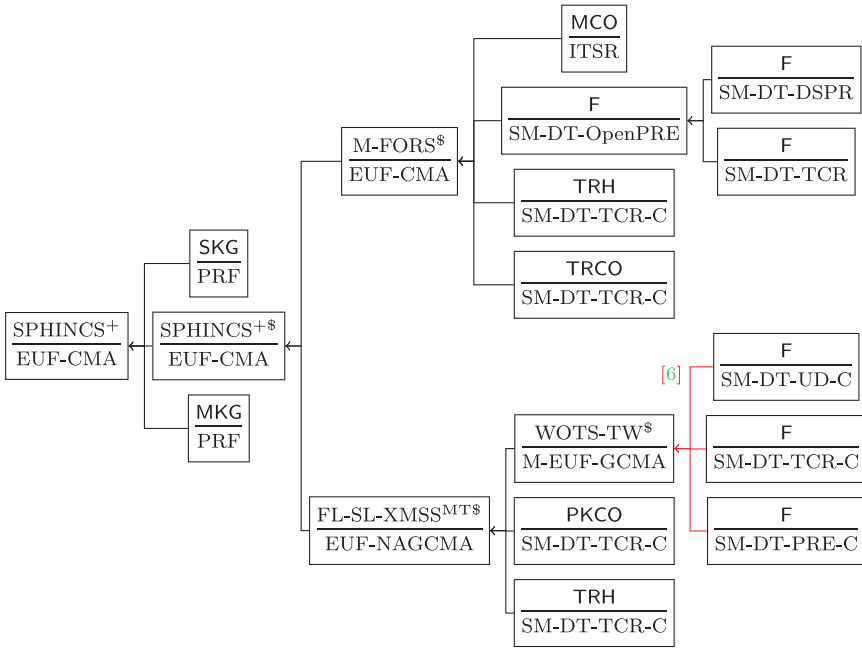


Fig. 12. High-level overview of (the proofs underlying) our formal verification. Nodes represent properties of cryptographic constructions or functions: the text above the line indicates the construction or function; the text below the line indicates the property. Edges represent implications between properties: the property denoted by the destination node is implied by the conjunction of properties denoted by the origin nodes. Red edges represent implications that have been formally verified in previous work (specified in an edge label), the results of which are reused here.

The leftmost node in Figure 12 signifies the primary objective of this work: The formal verification of the *Existential UnForgeability under Chosen-Message Attacks* (EUF-CMA) security of SPHINCS⁺. As depicted, we show that this property is implied by (1) the PRF property of SKG and MKG, KHF's used for the generation of secret keys and message compression keys, respectively; and (2) the EUF-CMA security of SPHINCS^{+\$}, a variant of SPHINCS⁺ that employs actual randomness rather than pseudorandomness. In essence, these initial reductions replace all pseudorandomness used throughout the construction by actual randomness. Consequently, in the remaining (modular) parts of the proof, we can immediately consider variants of the sub-constructions that use actual randomness (without further PRF-related reasoning).⁵

Next, we exhibit that the EUF-CMA security of SPHINCS^{+\$} follows from (1) the EUF-CMA security of M-FORS^{\$}, a multi-instance variant of FORS that employs actual randomness; and (2) the EUF-NAGCMA property — a non-adaptive, generic version of the EUF-CMA property — of FL-SL-XMSS^{MT\$}, a fixed-length, stateless variant of XMSS^{MT} that uses actual randomness.

Proceeding in a modular fashion, we demonstrate that the EUF-CMA security of M-FORS^{\$} can be based on (1) the ITSR property of MCO, a KHF used for the compression of (arbitrary-length) messages; (2) the SM-DT-OpenPRE property of F, a THF employed to generate Merkle tree leaves from secret key elements; and (3) the SM-DT-TCR-C property of TRH and TRCO, THFs used for the construction of Merkle trees from their leaves and, respectively, the compression of Merkle tree roots. In turn, we establish that the SM-DT-OpenPRE property of F is implied by its own SM-DT-DSPR and SM-DT-TCR properties. Interestingly, this implication can be considered a THF analog of Theorem 38 in [8], which states a comparable implication for KHF's. Correspondingly, the proofs require similar non-standard reasoning which, to the best of our knowledge, is unprecedented in EasyCrypt. Employing Theorem 25 in [8] — the proof of which only requires a relatively basic form of this reasoning — as an initial case study, we develop a formal verification technique aimed at this kind of reasoning. Building on this, we formally verify the implication required for SPHINCS⁺.

Then, for FL-SL-XMSS^{MT\$}, we show that its EUF-NAGCMA security is implied by (1) the M-EUF-GCMA property — a multi-instance, generic version of EUF-CMA specifically devised for the purpose of recovering the SPHINCS⁺ proof [17] — of WOTS-TW^{\$}, a variant of WOTS-TW that employs actual randomness; and (2) the SM-DT-TCR-C property of PKCO and TRH, THFs respectively employed for the compression of WOTS-TW^{\$} public keys and the construction of Merkle trees from their leaves.⁶ At this point, a single implication remains: The implication from several properties of THF F to the M-EUF-GCMA security of WOTS-TW^{\$}. Fortunately, in [6], this implication has already been

⁵ Nevertheless, at the expense of brevity (and code size), it should be relatively straightforward to extract a proof in which the PRF-related reasoning is localized, i.e., moved to the (modular) parts of the proof concerning the sub-constructions.

⁶ Indeed, TRH is the same function in both M-FORS^{\$} and FL-SL-XMSS^{MT\$}.

formally verified in a way that facilitates reuse. We capitalize on this and do not formally verify this implication anew.

Finally, combining all modular parts, we formally verify that the EUF-CMA security of SPHINCS⁺ can solely be based on the properties of the employed KHF_s and THF_s, as desired.

In the ensuing sections, we discuss the formal verification process more extensively, going by the cryptographic (sub-)constructions. Specifically, in order, we go over M-FORS[§], FL-SL-XMSS^{MT§}, and SPHINCS^{+§}/SPHINCS⁺. Throughout this discussion, we do not include any material directly from the produced formal verification artifacts in the interest of space. Instead, we cover the proofs underlying the formal verification in a way that allows for a near-verbatim translation to EasyCrypt, thus accurately representing the formally verified material.

Development. Before proceeding to the detailed discussion about (the formal verification of) the proof, we go over some meta-information in an attempt to provide additional insight into the development process and produced artifacts.

Excluding any reused artifacts, the development comprises approximately 17000 lines of code. However, the vast majority of this code is entirely verified by the tool. Furthermore, as hinted at before, the code that should be manually verified mostly constitutes a straightforward translation from the corresponding handwritten material (presented in this paper). In total, the project spanned approximately seven months,⁷ with one primary developer and four others (mainly) providing guidance.

Reflecting on the development process, we discern two key ways to facilitate future projects of a similar scale or nature. First, constructing EasyCrypt libraries for common cryptographic concepts could significantly reduce both specification and proof efforts. This work contributes to that end by developing and improving libraries relevant to hash-based cryptography. Additionally, extending EasyCrypt with features that automate frequently occurring reasoning steps in cryptographic proofs — particularly those that currently still require significant boilerplate and effort — could further streamline the process. A specific example of this would be PRF-related reductions.

4 M-FORS[§]

FORS was first introduced in [9] as the few-time signature scheme used in SPHINCS⁺. In practice, FORS is used with pseudorandom keys. However, our proof performs a PRF-related step on the level of SPHINCS⁺ that replaces all pseudorandom values with random values. Thus, when analyzing the security of FORS, we actually analyze FORS[§], a version of FORS that operates using actual randomness. In fact, it turns out that considering a multi-instance variant of FORS[§], M-FORS[§], is convenient for the proof because SPHINCS⁺ and the ITSR property (inherently) consider multiple instances of FORS[§].

⁷ Again, not including the development of any reused artifacts.

Intuitively, the (virtual) structure of a FORS^S instance is a sequence of Merkle trees, the leaves of which are digests of secret key values. The public key of such an instance is a single digest obtained by compressing the roots of the Merkle trees. A FORS^S signature comprises, for each Merkle tree, a single secret key value and the corresponding authentication path (defined below). The selection of secret key values in the signature is derived from the message.

A FORS^S instance is defined with respect to three parameters: k , a , and n . These parameters denote the number of Merkle trees (k), the height of each Merkle tree (a), and the byte-length (n) of the secret key elements, the public key, and the (values associated with the) nodes of the Merkle trees. From a , we compute the number of leaves for each Merkle tree as $t = 2^a$. Furthermore, FORS^S employs the THFs F, TRH, and TRCO. These functions have the same (public) parameter space and tweak space — referred to as the *public seed space* \mathcal{PS} and *address space* \mathcal{AD} — as well as the same message space $\{0, 1\}^*$ and digest space $\{0, 1\}^{8 \cdot n}$.

An instance of M-FORS^S essentially manages multiple FORS^S instances divided into sequences, where the number of sequences and the size of each sequence are respectively determined by parameters s and l' . M-FORS^S utilizes the KHF MCO to process arbitrary-length messages, obtaining (1) a fixed-length message — processable by a FORS^S instance — and (2) an index uniquely identifying a specific FORS^S instance. Moreover, it uses a random function MKG^S to generate a fresh indexing key for each message compression. Lastly, to guarantee a unique address for each THF evaluation in M-FORS^S's operations, we require that addresses have a corresponding *xtree index* (*xtri*), *keypair index* (*kpi*), *type index* (*typei*), *ftree height index* (*frhi*), and *ftree breadth index* (*frbi*). These indices are nonnegative integers that indicate, in the given order, the sequence of FORS^S instances, the FORS^S instance within the sequence, the type of operation (tree hashing or tree root compression), the height of the node (in the FORS^S instance), and the breadth of the node (in the FORS^S instance).⁸ Here, the breadth and height indices are only relevant for tree hashing operations.

In essence, provided with a public seed ps and an address ad , the key pair of a FORS^S instance is constructed as follows. Initially, a FORS^S secret key $sk = sk_0 \dots sk_{k \cdot t - 1} — sk_i \in \{0, 1\}^{8 \cdot n}$ for $0 \leq i < k \cdot t$ — is sampled uniformly at random. To obtain the corresponding public key, first, a sequence of $k \cdot t$ Merkle tree leaves is computed from the secret key by processing each element with F. The resulting sequence contains k non-overlapping subsequences of t leaves, each uniquely defining a Merkle tree of height a . The roots of these trees can be obtained by iteratively computing the layers of each tree. Specifically, in the construction of the layer at height h in the j -th Merkle tree, the node at breadth b can be computed from its children c_l and c_r as $\text{TRH}(ps, ad_{j,h,b}, c_l || c_r)$, where $ad_{j,h,b}$ denotes the unique address for this evaluation of TRH (obtained by appropriately adjusting ad based on j , h and b). Hereafter, we denote the

⁸ For the *ftree breadth index*, we do not consider a single tree, but rather the full sequence of trees in a FORS^S instance. This way, addresses are unique even for nodes in different trees but at the same height and breadth of their respective tree.

Listing 1 FORS^S Primary

```

1: procedure FORSS.KeyGen(ps, ad)
2:   skF ←  $\mathcal{U}(\{0, 1\}^{8 \cdot n})^{k \cdot t}$ 
3:   lvs ← FORSS.SkFToLvs(skF, ps, ad)
4:   rts ← []
5:   ad.typei ← ftrhType
6:   for  $i = 0, \dots, k - 1$  do
7:     lvsst ← lvs[ $i \cdot t : (i + 1) \cdot t$ ]
8:     rt ← LvsToRta(ps, ad, lvsst, i)
9:     rts ← rts || rt
10:  ad.typei ← ftrcType
11:  pkF ← TRCO(ps, ad, flatten(rts))
12:  return (pkF, ps, ad), (skF, ps, ad)
13: procedure FORSS.Sign(sk := (skF, ps, ad), m)
14:   lvs ← FORSS.SkFToLvs(skF, ps, ad)
15:   sig ← []
16:   ad.typei ← ftrhType
17:   for  $i = 0, \dots, k - 1$  do
18:      $j \leftarrow \text{toint}(m[i \cdot a : (i + 1) \cdot a])$ 
19:     skele ← skF[ $i \cdot t + j$ ]
20:     lvsst ← lvs[ $i \cdot t : (i + 1) \cdot t$ ]
21:     ap ← ConsAPa(ps, ad, lvsst, j, i)
22:     sig ← sig || (skele, ap)
23:   return sig
24: procedure FORSS.Verify(pk := (pkF, ps, ad),
25:   m, sig)
26:   pkF' ← FORSS.SigToPkF(m, sig, ps, ad)
   return pkF' = pkF

```

Listing 2 FORS^S Auxiliary

```

1: procedure FORSS.SkFToLvs(skF, ps, ad)
2:   lvs ← []
3:   ad.typei, ad.ftrhi ← ftrhType, 0
4:   for  $i = 0, \dots, k \cdot t - 1$  do
5:     ad.ftrhb ← i
6:     lf ← F(ps, ad, skF[i])
7:     lvs ← lvs || lf
8:   return lvs
9: procedure FORSS.SigToPkF(m, sig, ps,
10:  ad)
11:   rts ← []
12:   ad.typei ← ftrhType
13:   for  $i = 0, \dots, k - 1$  do
14:     skele, ap ← sig[i]
15:      $j \leftarrow \text{toint}(m[i \cdot a : (i + 1) \cdot a])$ 
16:     ad.ftrhi, ad.ftrbi ← 0,  $i \cdot t + j$ 
17:     lf ← F(ps, ad, skele)
18:     rt ← APToRta(ps, ad, ap, lf, j, i)
19:     rts ← rts || rt
20:   ad.typei ← ftrcType
21:   pkF ← TRCO(ps, ad, flatten(rts))
   return pkF

```

operator that performs this computation for Merkle trees of height h (i.e., for lists of leaves of length 2^h) by LvsToRt_h . Lastly, after computing the Merkle tree roots, the FORS^S public key pk is obtained by compressing the concatenation of these roots using TRCO. Since signing and verifying require the public seed and address that were used in key generation, we include them in both the public and secret key for convenience.

Given a FORS^S key pair, a message $m \in \{0, 1\}^{k \cdot a}$ is signed and verified in the following manner. Initially, m is split into k bitstrings of length a , each of which is interpreted as the big-endian binary representation of an integer in $[0, 2^a - 1]$. This gives rise to a k -tuple of integers (i_0, \dots, i_{k-1}) . Next, for every i_j , $0 \leq j < k$, a so-called *authentication path* is constructed for the i_j -th leaf of the j -th Merkle tree in the FORS^S instance. This path is the sequence comprising, in order, the sibling nodes along the path from the root to the considered leaf. Indeed, this path can be computed from the list of leaves and the index of the leaf. Throughout the remainder, we denote the operator that constructs these paths for Merkle trees of height h by ConsAP_h . Then, the FORS^S signature on m is a k -tuple of pairs $(\text{sk}_{i_j}, \text{ap}_{i_j})$, $0 \leq j < k$, where sk_{i_j} and ap_{i_j} are the secret key element and authentication path corresponding to the i_j -th leaf of the j -th Merkle tree. Verification of a signature on m is performed by, initially, extracting the integers (i_0, \dots, i_{k-1}) from m in the same way as before. Subsequently, the

secret key elements in the signature are transformed into the corresponding leaves via F . Combining each of these leaves with the associated authentication path in the signature, the root of each Merkle tree in the FORS^S instance is computed. This is achieved by iteratively reconstructing the path from the leaf to the root using the sibling nodes in the authentication path. For instance, if the i_j -th leaf is a right child, the second node on the path is computed as $n_1 = \text{TRH}(\text{ps}, \text{ad}_{j,1,x}, \text{ap}_{i_j}[a-1] \parallel \text{lf}_{i_j})$, where $x = \lfloor i_j/2 \rfloor$, $\text{ad}_{j,1,x}$ is the unique address for this evaluation of TRH, and lf_{i_j} is the i_j -th leaf; then, if n_1 is a left child, the third node on the path equals $\text{TRH}(\text{ps}, \text{ad}_{j,2,y}, n_2 \parallel \text{ap}_{i_j}[a-2])$, where $y = \lfloor x/2 \rfloor$; and so forth.⁹ Henceforth, we denote the operator that performs this computation for Merkle trees of height h (i.e., for authentication paths of length $\log_2 h$) by ApToRt_h . Finally, the produced roots are compressed using TRCO to obtain a candidate public key. If and only if this candidate public key matches the original public key, verification succeeds.

Following the foregoing descriptions, Listing 1 provides the specification of FORS^S's key generation, signing, and verification algorithm. These algorithms employ auxiliary procedures for the computation of (1) a sequence of Merkle tree leaves corresponding to a FORS^S secret key, and (2) a FORS^S public key corresponding to a FORS^S signatures. For reuse purposes, we specify these auxiliary procedures separately in Listing 2. In the specifications, $l[i : j]$ denotes the slice of list l from index i (including) up to index j (excluding), $\text{flatten}(l)$ denotes the sequential concatenation of all elements in list l , and $\text{toint}(s)$ denotes the integer corresponding to bitstring s (assuming big-endian binary representation).

At this point, it is rather straightforward to specify M-FORS^S, as it essentially constitutes a collection of FORS^S instances combined with a way to compress messages and select which instance to use for signing and verification. Listing 3 provides the specification of M-FORS^S's algorithms.

Security Property. For M-FORS^S, we effectively consider a slight variant of the customary EUF-CMA security property that accounts for the fact that M-FORS^S expects to be provided with a public seed and an address. Furthermore, for the usage of the THFs to be secure (with respect to their assumed properties), this public seed should be sampled uniformly at random. The game and oracle formalizing this security property are respectively provided in Figures 13 and 14. Here, ad_z denotes an arbitrary address used for initialization.

Formal Verification. As illustrated in Figure 12, we demonstrate that the EUF-CMA security of M-FORS^S is implied by the ITSR property of MCO, the SM-DT-OpenPRE property of F , and the SM-DT-TCR-C property of TRH and TRCO. For the ITSR property of MCO, we instantiate MAP (see Figure 1) with CM, a function that maps $(m_c, i) \in \{0, 1\}^{k \cdot a} \times [0, s \cdot l - 1]$ —i.e., outputs from MCO—to the set $S = \{(i, j, \text{toint}(m_c[j \cdot a : (j+1) \cdot a]) \mid 0 \leq j < k\}$. Intuitively, a tuple (x, y, z) from this set can be interpreted as an index pointing to the z -th

⁹ Whether the nodes along the reconstructed path are left or right children can be computed from the value of i_j .

Listing 3 M-FORS[§]

```

1: procedure M-FORS§.KeyGen(ps, ad)
2:   pkMF, skMF  $\leftarrow$  [], []
3:   for  $i = 0, \dots, s \cdot l' - 1$  do
4:     ad.xtri, ad.kpi  $\leftarrow$   $\lfloor i/l' \rfloor, i \bmod l'$ 
5:     (pkF, _, _) , (skF, _, _)  $\leftarrow$  FORS§.KeyGen(ps, ad)
6:     pkMF, skMF  $\leftarrow$  pkMF || pkF, skMF || skF
7:   return (pkMF, ps, ad), (skMF, ps, ad)
8: procedure M-FORS§.Sign(sk := (skMF, ps, ad), m)
9:   mk  $\leftarrow$  MKG§(m)
10:  mc, i  $\leftarrow$  MCO(mk, m)
11:  ad.xtri, ad.kpi  $\leftarrow$   $\lfloor i/l' \rfloor, i \bmod l'$ 
12:  sigF  $\leftarrow$  FORS§.Sign((skMF[i], ps, ad), mc)
13:  return mk, sigF
14: procedure M-FORS§.Verify(pk := (pkMF, ps, ad), m, sig := (mk, sigF))
15:  mc, i  $\leftarrow$  MCO(mk, m)
16:  ad.xtri, ad.kpi  $\leftarrow$   $\lfloor i/l' \rfloor, i \bmod l'$ 
17:  isValid  $\leftarrow$  FORS§.Verify((pkMF[i], ps, ad), mc, sigF)
18:  return isValid

```

Game ^{EUF-CMA} _{A, M-FORS[§]}
<pre> 1 : ad \leftarrow ad_z 2 : ps \leftarrow $\\$U(\mathcal{P}S) 3 : (pk, sk) \leftarrow M-FORS[§].KeyGen(ps, ad) 4 : [O]_{M-FORS[§]}.Init(sk) 5 : m', sig' \leftarrow A^{[O]_{M-FORS[§]}.Query}.Forge(pk) 6 : isValid \leftarrow M-FORS[§].Verify(pk, m, sig) 7 : isFresh \leftarrow m' \notin [O]_{M-FORS[§]}..\mathcal{M} 8 : return isValid \wedge isFresh </pre>

Fig. 13. EUF-CMA game for M-FORS[§].

[O] _{M-FORS[§]}
<pre> vars sk, \mathcal{M} Init(ski) <hr/> 1 : sk, $\mathcal{M} \leftarrow$ ski, [] <hr/> Query(m) <hr/> 1 : sig \leftarrow M-FORS[§].Sign(sk, m) 2 : $\mathcal{M} \leftarrow$ $\mathcal{M} \parallel m$ 3 : return </pre>

Fig. 14. Oracle employed in EUF-CMA game for M-FORS[§].

leaf of the y -th Merkle tree in the x -th FORS[§] instance. Formally, the security theorem we consider is the following.

Theorem 1 (EUF-CMA for M-FORS[§]). *For any adversary \mathcal{A} , there exist adversaries $\mathcal{B}_0, \mathcal{B}_1, \mathcal{B}_2$, and \mathcal{B}_3 — each with approximately the same running time as \mathcal{A} — such that the following inequality holds.*

$$\begin{aligned}
\text{Adv}_{\text{M-FORS}^{\S}}^{\text{EUF-CMA}}(\mathcal{A}) &\leq \text{Adv}_{\text{MCO, CM}}^{\text{ITSR}}(\mathcal{B}_0) + \text{Adv}_{\text{F}, t_f}^{\text{SM-DT-OpenPRE}}(\mathcal{B}_1) \\
&\quad + \text{Adv}_{\text{TRH, THFC}, t_{\text{trh}}}^{\text{SM-DT-TCR-C}}(\mathcal{B}_2) \\
&\quad + \text{Adv}_{\text{TRCO, THFC}, t_{\text{trco}}}^{\text{SM-DT-TCR-C}}(\mathcal{B}_3)
\end{aligned}$$

Here, THFC denotes an arbitrary THF collection containing F, TRH, and TRCO. Furthermore, $t_f = s \cdot l' \cdot k \cdot t$, $t_{\text{trh}} = s \cdot l' \cdot k \cdot (t - 1)$, and $t_{\text{trco}} = s \cdot l'$.

In essence, the formal verification of Theorem 1 proceeds by an exhaustive case analysis on the situation where \mathcal{A} wins Game^{EUF-CMA}_{A, M-FORS[§]}. This case analysis com-

prises four distinct cases; for each of these cases, the probability is bounded by exactly one of the advantage terms on the right-hand side of the theorem's inequality. In the following, $G_{\mathcal{A}}^{\top}$ signifies the event $\text{Game}_{\mathcal{A}, \text{M-FORS}^{\mathfrak{s}}}^{\text{EUF-CMA}} = 1$.

Case Distinction for $G_{\mathcal{A}}^{\top}$. First, note that a valid EUF-CMA forgery for M-FORS^s consists of a message m' and a signature $\text{sig}' = (\text{mk}', \text{sigF}')$ such that m' is fresh and sig' is a valid signature on m' under the considered public key $\text{pk} = (\text{pkMF}, \text{ps}, \text{ad})$. Here, recall that sig' is only valid if the FORS^s candidate public key pkF' , computed from $(m'_c, i') = \text{MCO}(\text{mk}', m')$ and sigF' , equals $\text{pkF} = \text{pkMF}[i']$. By the nature of the computations, validity of the forgery implies that, at some point during the construction of pkF' , the considered values must coincide with the corresponding values in the original construction of pkF .

Harnessing the above observation, the first case we distinguish is one where the compression of m' (using MCO indexed on mk') results in the selection of a set of secret key elements from a FORS^s instance such that all of these values were already revealed as part of (the replies to) the issued signature queries.¹⁰ Alternatively stated, the set $\text{CM}(m'_c, i')$ is contained in the union of the analogous sets for the key/message pairs corresponding to the issued signature queries. As m' is fresh, it follows that the pair (mk', m') can be used to break ITSR. In the remaining, E_M denotes the event that captures this case.

If the first case does not occur ($\neg E_M$), the forgery contains at least one secret key element skele' not revealed during the game. Then, the second case we distinguish concerns the leaf lf' produced from this secret key element equaling the corresponding leaf lf in the computation of pkF . In this case, skele' is a preimage of lf under F and, hence, can be used to break SM-DT-OpenPRE (for F). Hereafter, E_F signifies the event capturing this case (within $\neg E_M$).

If both the first and second case do not happen ($\neg E_M \wedge \neg E_F$), the forgery contains a secret key element skele' that (1) was not revealed during the game and (2) produces a leaf lf' that is different from the one in the original construction of pkF . As such, the third case we distinguish regards the Merkle tree root computed from lf' and the associated authentication path ap' (from the same pair in the forgery) equaling the corresponding Merkle tree root in the original computation of pkF . Here, it must be the case that, at a certain point, the values on the reconstructed path (using lf' and ap') coincide with the corresponding values in the original Merkle tree. So, because the initial node(s) on these paths *are not* equal, the first node for which the paths converge must be obtained by applying TRH on different inputs. These inputs form a collision for TRH and, thus, can be used to break SM-DT-TCR-C (for TRH). Henceforth, we denote the event that captures this case (within $\neg E_M \wedge \neg E_F$) by E_T .

Finally, if all of the foregoing cases do not transpire ($\neg E_M \wedge \neg E_F \wedge \neg E_T$), it must be the case that one of the Merkle tree roots provided as (part of the) input to TRCO to produce pkF' does not equal the corresponding root used

¹⁰ The values need not all be revealed in (the reply to) a single signature query. They may have been revealed over (the replies to) any number of signature queries.

in the original computation of pkF. Therefore, in this case, the (concatenated) Merkle tree roots used to compute pkF' and pkF form a collision for TRCO and can be used to break SM-DT-TCR-C (for TRCO).

Bound on $\Pr[G_A^\top \wedge E_M]$. If the compression of m' (using mk') indicates a set of secret key elements already revealed in (the responses to) the issued signature queries, we construct a reduction adversary \mathcal{R}^A playing in $\text{Game}_{\mathcal{A}, \text{MCO}, \text{CM}}^{\text{ITSR}}$ that straightforwardly simulates an execution of $\text{Game}_{\mathcal{A}, \text{M-FORS}^\S}^{\text{EUF-CMA}}$ but, instead of sampling, uses $[\text{OITSR}]$ to obtain message keys for the compression of messages contained in queries by \mathcal{A} . Upon receiving the forgery from \mathcal{A} , \mathcal{R}^A directly extracts and returns (mk', m') , winning its own game. As a result, we can bound $\Pr[G_A^\top \wedge E_M]$ by $\text{Adv}_{\text{MCO}, \text{CM}}^{\text{ITSR}}(\mathcal{R}^A)$.

Bound on $\Pr[G_A^\top \wedge \neg E_M \wedge E_F]$. In case the compression of m' indicates an unprecedented secret key element for which the image under F coincides with the corresponding original Merkle tree leaf, we construct the following reduction adversary playing in $\text{Game}_{\mathcal{R}^A, F, t_f}^{\text{SM-DT-OpenPRE}}$. In its first stage, \mathcal{R}^A constructs and returns a list containing every address used to create Merkle tree leaves from secret key elements in M-FORS[§]. Then, in its second stage, \mathcal{R}^A utilizes the given public seed and Merkle tree leaves to compute the corresponding M-FORS[§] public key and runs \mathcal{A} with this public key, the public seed, and the initialization address. During the execution of \mathcal{A} , the reduction adversary answers signature queries in accordance with M-FORS[§].Sign, acquiring any necessary secret key elements via $[\text{OOPRE}]_F$.Open. Upon receiving the forgery from \mathcal{A} , \mathcal{R}^A finds the secret key element not revealed in (responses to) the issued signature queries, and returns this element together with the associated index. By construction, the reduction adversary did not query any indices corresponding to secret key elements not included in (responses to) the issued signature queries. Consequently, \mathcal{R}^A wins its own game and we can bound $\Pr[G_A^\top \wedge \neg E_M \wedge E_F]$ by $\text{Adv}_{F, t_f}^{\text{SM-DT-OpenPRE}}(\mathcal{R}^A)$.

Bound on $\Pr[G_A^\top \wedge \neg E_M \wedge \neg E_F \wedge E_T]$. If the leaf obtained from the unprecedented secret key element in the forgery does not equal the corresponding leaf in the original Merkle tree, but the root computed based on the associated authentication path does coincide with the root of the original Merkle tree, we construct the ensuing reduction adversary playing in $\text{Game}_{\mathcal{R}^A, \text{TRH}, \text{THFC}, t_{\text{trh}}}^{\text{SM-DT-TCR-C}}$. In its first stage, \mathcal{R}^A constructs a key pair in line with M-FORS[§].KeyGen by utilizing the provided oracles. Specifically, for each FORS[§] instance, the reduction adversary samples the secret key, computes the Merkle tree leaves by querying the collection oracle on the secret key elements, computes the Merkle tree roots by querying the challenge oracle on the (concatenation of) sibling nodes — specifying these as targets — and computes the FORS[§] public keys by querying the collection oracle on the (concatenation of) Merkle tree roots. Then, in its second stage, \mathcal{R}^A runs \mathcal{A} with the previously generated public key, the received public seed, and the initialization address. Since \mathcal{R}^A constructed the considered key pair itself, it can trivially simulate the signing oracle for \mathcal{A} . Upon receiving the forgery from \mathcal{A} , \mathcal{R}^A computes the non-matching leaf from the unprecedented secret key element; extracts the collision based on this leaf, the associated

authentication path, and the original Merkle tree; and returns the extracted collision and the associated index, winning its own game. As such, we can bound $\Pr[G_{\mathcal{A}}^{\top} \wedge \neg E_M \wedge \neg E_F \wedge E_T]$ by $\text{Adv}_{\text{TRH,THFC},t_{\text{trh}}}^{\text{SM-DT-TCR-C}}(\mathcal{R}^{\mathcal{A}})$.

Bound on $\Pr[G_{\mathcal{A}}^{\top} \wedge \neg E_M \wedge \neg E_F \wedge \neg E_T]$. Finally, if none of the previous cases occurs, we construct a reduction adversary playing in $\text{Game}_{\mathcal{R}^{\mathcal{A}},\text{TRCO,THFC},t_{\text{trco}}}^{\text{SM-DT-TCR-C}}$ that, in essence, is extremely similar to the one considered in the preceding case. Namely, in its first stage, $\mathcal{R}^{\mathcal{A}}$ constructs a M-FORS[§] key pair in the same way as the previous reduction adversary. However, in this case, $\mathcal{R}^{\mathcal{A}}$ employs the collection oracle for the construction of Merkle trees and the challenge oracle for the compression of Merkle tree roots. In its second stage, $\mathcal{R}^{\mathcal{A}}$ also proceeds in the same way as the previous reduction adversary, except that it now extracts and returns a collision (and the associated index) based on the Merkle tree root computed from the forgery. Following, we can bound $\Pr[G_{\mathcal{A}}^{\top} \wedge \neg E_M \wedge \neg E_F \wedge \neg E_T]$ by $\text{Adv}_{\text{TRCO,THFC},t_{\text{trco}}}^{\text{SM-DT-TCR-C}}(\mathcal{R}^{\mathcal{A}})$.

Final Result. At this point, Theorem 1 trivially follows from the established bounds and the fact that the sum of the probabilities for the considered cases is precisely equal to $\text{Adv}_{\text{M-FORS}^{\text{§}}}^{\text{EUF-CMA}}(\mathcal{A})$.

4.1 SM-DT-OpenPRE From SM-DT-TCR and SM-DT-DSPR

At this stage, we go over the formal verification of the aforementioned generic relation between the SM-DT-OpenPRE, SM-DT-DSPR and SM-DT-TCR properties of a THF with a finite message space. By instantiating this relation with F^{11} (and combining it with Theorem 1), we complete the modular part of the formal verification rooted at M-FORS[§] (see Figure 12). Formally, the security statement we consider is the following.

Theorem 2 (SM-DT-OpenPRE for a THF). *For any adversary \mathcal{A} , there exist adversaries \mathcal{B}_0 and \mathcal{B}_1 — each with approximately the same running time as \mathcal{A} — such that the following inequality holds.*

$$\text{Adv}_{\text{THF},t}^{\text{SM-DT-OpenPRE}}(\mathcal{A}) \leq \text{Adv}_{\text{THF},t}^{\text{SM-DT-DSPR}}(\mathcal{B}_0) + 3 \cdot \text{Adv}_{\text{THF},t}^{\text{SM-DT-TCR}}(\mathcal{B}_1)$$

Here, THF is an arbitrary THF with a finite message space \mathcal{M} , and $t \geq 0$.

In [8], the authors demonstrate generic relations between similar properties for KHFes. The proofs in [8] make use of non-standard techniques that we also use for the proof of Theorem 2. As these techniques are unprecedented in EasyCrypt, we elaborate on the formal verification and its challenges here.

Typical proofs considered in EasyCrypt compare, at each step, (the simultaneous execution of) two games. This encompasses usual proofs via direct reduction or game hopping. Namely, in these cases, the tool’s probabilistic relational

¹¹ Although F technically has an infinite message space (see Section 2), we can replace it in the context of M-FORS[§]/SPHINCS⁺ by an equivalent function with finite message space $\{0, 1\}^{s \cdot n}$ because F is only evaluated on messages from this space in this context.

logic enables formal reasoning about the desired equivalences (potentially up to some failure event) between each pair of games. Unfortunately, it is not possible to refer to any games beyond the two collated games, which would be needed to formally verify our proof directly. Particularly, the proof for Theorem 2, which closely resembles the proofs for Theorems 25 and 38 in [8], requires simultaneous reasoning about *four games*: $\text{Game}_{\mathcal{A}, \text{THF}, t}^{\text{SM-DT-OpenPRE}}$, $\text{Game}_{\mathcal{R}_D^{\mathcal{A}}, \text{THF}, t}^{\text{SM-DT-DSPR}}$, $\text{Game}_{\mathcal{R}_T^{\mathcal{A}}, \text{THF}, t}^{\text{SM-DT-SPprob}}$, and $\text{Game}_{\mathcal{R}_T^{\mathcal{A}}, \text{THF}, t}^{\text{SM-DT-TCR}}$. Here, the reduction adversaries $\mathcal{R}_D^{\mathcal{A}}$ and $\mathcal{R}_T^{\mathcal{A}}$ are relatively straightforward. Specifically, in their first stage, both reduction adversaries run \mathcal{A} 's first stage to obtain the list of tweaks; then, for each tweak in this list, they query their own oracle on this tweak and a uniformly random message (freshly sampled for each query), constructing a list of digests from the responses. In their second stage, the reduction adversaries run \mathcal{A} 's second stage, providing it with the received public parameter and the previously constructed digest list. Upon receiving (i', x') from \mathcal{A} , $\mathcal{R}_T^{\mathcal{A}}$ returns (i', x') and $\mathcal{R}_D^{\mathcal{A}}$ returns (i', b) , where b guesses that the message contained in $\mathcal{R}_D^{\mathcal{A}}$'s i' -th query only has a single preimage if and only if x' equals this message.

Using the above four games, the proof (and its formal verification) proceeds by performing an extremely granular case analysis across multiple dimensions, expressing the success probability associated with each game as a sum of probabilities of fine-grained events. More precisely, these dimensions of analysis are (1) the index j chosen by \mathcal{A} , (2) the number of preimages for the digest pointed to by j , and (3) the validity of \mathcal{A} 's provided preimage. On a more technical level, we perform this case analysis by defining F_i^j and S_i^j , two auxiliary games parameterized on the number of preimages i and the index j . Intuitively, these games are analogous to the similarly named auxiliary games in [8]: F_i^j and S_i^j respectively capture the failure and success cases for the considered SM-DT-OpenPRE game. Utilizing these auxiliary games, the proof advances by performing the following for each game (of the four primary games): First, decompose the success probability across the above-mentioned dimensions; second, show that, for some cases, the probability equals that of either F_i^j or S_i^j ; third, show that, for some (other) cases, the probability equals 0; fourth, show that, for the remaining cases—corresponding to the adversary finding a preimage different from the original one chosen by the reduction adversary (which is information-theoretically hidden in the preimage set of size i)—the probability can be expressed as $\frac{i-1}{i} \cdot S_i^j$; and, lastly, combining the results into a closed formula. Subsequently, the resulting closed formulas can be combined to derive Theorem 2. In the process performed for each game, the second and third step constitute customary proofs for EasyCrypt, while the first and fourth step are non-standard and technically involved. For the non-standard steps, we develop and apply techniques aimed at the required reasoning. We elaborate on these below.

In the decomposition of the success probabilities, our objective is to express the success probability of a game G ($\Pr[G^\top]$) as follows. Here, $G_{i,j}^\top$ denotes an

event capturing a specific case of winning the game, parameterized by i and j .

$$\Pr[G^\top] = \sum_{j=0}^{t-1} \left(\sum_{i=0}^{|\mathcal{M}|} \Pr[G_{i,j}^\top] \right)$$

We prove this equality by two applications of induction from the outside in — i.e., first on j , then on i . So, we start with proving by induction that, for all z , the following holds, where ti denotes the (adversarially chosen) target index.

$$\Pr[G^\top \wedge 0 \leq \text{ti} < z] = \sum_{j=0}^{z-1} \Pr[G^\top \wedge \text{ti} = j]$$

In this proof, the base case (0) is trivial, and the inductive step directly follows from the fact that the events are disjoint. We obtain the desired summation in the range $[0, t - 1]$ by showing that the adversary loses if it exceeds the number of targets. Then, we continue the deconstruction by introducing the second summation. Specifically, we prove by induction that, for all z , the following holds, where ntp denotes the number of preimages of the (adversarially chosen) target.

$$\Pr[G^\top \wedge 0 \leq \text{ti} < t \wedge 0 \leq \text{ntp} \leq z] = \sum_{j=0}^{t-1} \left(\sum_{i=0}^z \Pr[G^\top \wedge \text{ti} = j \wedge \text{ntp} = i] \right)$$

This proof is similar to the previous one, except that the base case requires us to argue that both probability expressions collapse to the case where the selected target has no preimages. We acquire the intended summation by proving that the number of preimages cannot exceed the (finite) number of messages $|\mathcal{M}|$. The resulting decomposition allows us to continue along the above proof outline.

Lastly, the most technically involved part of the formal verification concerns the reasoning about information-theoretically hidden preimages, which is at the heart of expressing the probability that the adversary finds a second preimage in S_i^j as the probability of sampling an element uniformly at random from a set of cardinality i and it not equaling a fixed element from this set. Proving this in EasyCrypt is technically involved because there is no inherent mechanism for reasoning about complex conditional probabilities (related to the execution of games). In our case, it requires transforming S_i^j into a variant that initially samples a digest y from the distribution induced by THF and, only in case y has exactly i preimages, samples x from the set of y 's preimages after the adversary returned x' . In actuality, this transformation requires several intermediate transformations, each of which needs to guarantee that either the adversary's view is unaltered or the relevant event is not triggered. Loosely speaking, we alter the original S_i^j in the following sequence of game hops. First, we use the sampled message x only when the corresponding digest y has exactly i preimages, and make the adversary's view independent of it otherwise. Second, we invert the order of the sampling by sampling y first and sampling x from the preimage set of y . Third, we move the sampling of x to the end of the game. At this point,

since x is sampled after the adversary returns its guess, the desired probability claim is relatively straightforward to prove using EasyCrypt’s logic. For the technically involved and novel (for EasyCrypt) part of this proof, we developed several reusable results that permit reasoning about distributions over sets of images and preimages in functions with finite domain.

5 FL-SL-XMSS^{MT}[§]

XMSS^{MT} is a stateful post-quantum digital signature scheme that is standardized as a standalone construction, meaning that it is used to sign arbitrary-length messages [12]. In effect, SPHINCS⁺ employs a stateless variant of this scheme that is exclusively used to sign fixed-length messages, i.e., FORS public keys, and therefore omits any initial message compression. We denote this variant by FL-SL-XMSS^{MT}. Within SPHINCS⁺, FL-SL-XMSS^{MT} operates — akin to FORS/M-FORS — using pseudorandomness. As we perform an all-encompassing PRF-related step on the level of SPHINCS⁺, we only consider FL-SL-XMSS^{MT}[§], a variant of FL-SL-XMSS^{MT} which operates using actual randomness.

Intuitively, the (virtual) structure of a FL-SL-XMSS^{MT}[§] instance constitutes a hypertree. Each “node” in this hypertree is an instance of a Merkle signature scheme that uses WOTS-TW[§] as its OTS scheme, essentially constituting a variant of XMSS; we refer to these instances as “inner trees”. The inner trees on the bottom layer of the hypertree are used to sign messages; all other inner trees are used to sign the roots of the inner trees one layer below. An instance of FL-SL-XMSS^{MT}[§] is defined with respect to parameters n , analogous to the identically named parameter for FORS[§]; h' , the height of each inner tree; and d , the number of layers in the hypertree. From h' and d , we compute the number of leaves of each inner tree as $l' = 2^{h'}$, the height of the hypertree as $h = h' \cdot d$, and the number of leaves of the hypertree as $l = 2^h$. Furthermore, FL-SL-XMSS^{MT}[§] employs, in addition to the previously introduced F and TRH, the THF PKCO for the compression of WOTS-TW[§] public keys to inner tree leaves. PKCO has the same domain and range as the other THFs,¹² which largely remain identical to those used in FORS[§]. Here, the only difference concerns the (minimal) indices associated with the addresses. Specifically, in this context, we require addresses to have an associated *layer index* (li), *xtree index* (xtri), *type index* (typei), *key pair index* (kpi), *xtree height index* (xtrhi), and *xtree breadth index* (xtrbi). Respectively, these indices indicate the layer, the inner tree within the layer, the type of operation (chaining, public key compression, or tree hashing), the leaf (within the inner tree), and the height and breadth of the node (within the inner tree). Lastly, as we reuse formal verification artifacts for WOTS-TW[§], we mostly abstract this scheme away, only providing details when needed. For more information about this scheme and its formal verification, see [6].

Loosely speaking, given a public seed and an address, a FL-SL-XMSS^{MT}[§] key pair is constructed as follows. First, a FL-SL-XMSS^{MT}[§] secret key is a uniformly

¹² Recall that we consider the same message space for each THF (i.e., $\{0, 1\}^*$).

random sequence consisting of all WOTS-TW[§] secret keys used throughout the construction. Each of these WOTS-TW[§] secret keys comprises a fixed number (*len*) of bitstrings of length $8 \cdot n$ and is associated with exactly one leaf of a single inner tree. The corresponding FL-SL-XMSS^{MT§} public key is the root of the hypertree, which is computed by (1) transforming the WOTS-TW[§] secret keys associated with the topmost inner tree into the corresponding public keys via *F*, (2) compressing these public keys with PKCO to obtain the corresponding leaves, and (3) computing the root of the topmost inner tree by iteratively constructing its layers (from these leaves) using TRH. For the same reasons as in FORS[§]/M-FORS[§], we include the public seed and address in both the public and secret key.

A FL-SL-XMSS^{MT§} signature on a message $m \in \{0, 1\}^{8 \cdot n}$ is a sequence of *d* pairs, where the *i*-th pair consists of a WOTS-TW[§] signature and an authentication path corresponding to (a particular leaf of) an inner tree on the *i*-th layer. Here, the inner tree and leaf to be used on the bottom layer are provided as input, completely determining the utilized inner trees and leaves from the upper layers. Naturally, the WOTS-TW[§] signatures are produced with appropriate calls to the signing procedure of WOTS-TW[§]; the associated authentication paths are constructed analogously to the construction of such paths in FORS[§], where the considered leaf is obtained by compressing the corresponding WOTS-TW[§] public key via PKCO. Then, verification of a FL-SL-XMSS^{MT§} signature succeeds if and only if the candidate root of the hypertree — constructed from the signature — equals the actual root of the hypertree contained in the public key. More precisely, in a bottom-up manner, the roots of the inner trees corresponding to the pairs in the signature are computed, where the message *m* serves as the initial root: First, the WOTS-TW[§] public key is computed from the WOTS-TW[§] signature (in the pair) and the considered root; second, the leaf corresponding to the obtained public key is produced via PKCO; third, the root of the inner tree is computed using the obtained leaf and the authentication path (in the pair), again in a similar manner to the analogous computation in FORS[§]. Repeating this process for each pair in the signature results in the candidate hypertree root.

In line with the preceding, the specification of FL-SL-XMSS^{MT§} is provided in Listings 4 and 6; respectively, these listings specify the primary and auxiliary algorithms, the latter of which are specified separately for reuse purposes. In these specifications, the *ConsAP*, *LvsToRt*, and *APTToRt* are the same operators as in FORS[§], yet parameterized on *h'* instead of *a*.¹³ Furthermore, *nrtrees*(*i*) denotes the number of inner trees in layer *i* (which equals $2^{h' \cdot (d-i-1)}$), and — preventing clutter due to indexing — *skMX*_{*i,j*} denotes the sequence (of length *l'*) of WOTS-TW[§] secret keys corresponding to the *j*-th inner tree on the *i*-th layer. Finally, although the WOTS-TW[§] procedures are not explicitly specified here, their names are purposely indicative of their functioning; moreover, they are mostly analogous to the similarly named procedures previously specified in this paper.

¹³ The final argument to these operators is omitted as it was only used to determine which Merkle tree in the FORS[§] instance to consider.

Listing 4 FL-SL-XMSS^{MT \S} Primary

```

1: procedure FL-SL-XMSSMT $\S$ .KeyGen(ps, ad)
2:   skMX  $\leftarrow$  [ ]
3:   for  $i = 0, \dots, d - 1$  do
4:     for  $j = 0, \dots, \text{nrtrees}(i) - 1$  do
5:       skX  $\leftarrow$   $\mathcal{U}(\{0, 1\}^{s \cdot n})^{\text{len}}(l')$ 
6:       skMX  $\leftarrow$  skMX || skX
7:   pkMX  $\leftarrow$  FL-SL-XMSSMT $\S$ .SkMXToPkMX(skMX, ps, ad)
8:   return (pkMX, ps, ad), (skMX, ps, ad)
9: procedure FL-SL-XMSSMT $\S$ .Sign(sk := (skMX, ps, ad),  $m$ ,  $i$ )
10:  rt  $\leftarrow$   $m$ 
11:  ad.xtri  $\leftarrow$   $i$ 
12:  sig  $\leftarrow$  [ ]
13:  for  $j = 0, \dots, d - 1$  do
14:    ad.li, ad.xtri, ad.kpi  $\leftarrow$   $j$ , [ad.xtri/ $l'$ ], ad.xtri mod  $l'$ 
15:    skX  $\leftarrow$  skMXad.li, ad.xtri
16:    skW  $\leftarrow$  skX[ad.kpi]
17:    ad.typei  $\leftarrow$  chType
18:    sigW  $\leftarrow$  WOTS-TW $\S$ .Sign((skW, ps, ad), rt)
19:    lvsX  $\leftarrow$  FL-SL-XMSSMT $\S$ .SkXToLvsX(skX, ps, ad)
20:    ad.typei  $\leftarrow$  xtrhType
21:    apX  $\leftarrow$  ConsAP $h'$ (ps, ad, lvsX, ad.kpi)
22:    rt  $\leftarrow$  LvsToRt $h'$ (ps, ad, lvsX)
23:    sig  $\leftarrow$  sig || (sigW, apX)
24:  return sig
25: procedure FL-SL-XMSSMT $\S$ .Verify(pk := (pkMX, ps, ad),  $m$ , sig,  $i$ )
26:  pkMX'  $\leftarrow$  FL-SL-XMSSMT $\S$ .SigToPkMX( $m$ , sig,  $i$ , ps, ad)
27:  return pkMX' = pkMX

```

Security Property. Regarding FL-SL-XMSS^{MT \S} , we consider a non-adaptive, generic variant of the EUF-CMA security property denoted EUF-NAGCMA. Here, “non-adaptive” refers to the fact that the selection of messages for which the adversary receives signatures must happen at once; “generic” refers to the fact that this selection happens without knowledge of the considered public key. Akin to the EUF-CMA property for M-FORS ^{\S} , this property accounts for the fact that FL-XMSS-TW ^{\S} expects a public seed and an address, where the public seed should be sampled uniformly at random. Furthermore, this property uses an indexed version of the conventional freshness definition. Figure 15 provides the game formalizing this property, where ad_z represents an arbitrary address.

Formal Verification. As Figure 12 depicts, we show that the EUF-NAGCMA security of FL-SL-XMSS^{MT \S} can be based on the M-EUF-GCMA property of WOTS-TW ^{\S} , as well as the SM-DT-TCR-C property of PKCO and TRH. Here, we impose some additional constraints on the adversary’s behavior in terms of its collection oracle queries. Intuitively, this limited oracle access can be interpreted as modeling that the scheme remains secure in a greater context where the same collection of THFs is also evaluated on different addresses (e.g., SPHINCS⁺). Formally, the security theorem we consider is stated below.

Theorem 3 (EUF-NAGCMA for FL-SL-XMSS^{MT \S}). *For any adversary \mathcal{A} that does not query its collection oracle on addresses used in FL-SL-XMSS^{MT \S} ,*

Listing 5 FL-SL-XMSS^{MT}^S Auxiliary

```

1: procedure FL-SL-XMSSMTS.SkXToLvsX(skX, ps, ad)
2:   lvsX ← []
3:   for  $i = 0, \dots, l' - 1$  do
4:     ad.typei, ad.kpi ← chType,  $i$ 
5:     pkW ← WOTS-TWS.SkWToPkW(skX[i], ps, ad)
6:     ad.typei ← pkcoType
7:     lf ← PKCO(ps, ad, flatten(pkW))
8:     lvs ← lvs || lf
9:   return lvs
10: procedure FL-SL-XMSSMTS.SkMXToPkMX(skMX, ps, ad)
11:   ad.li, ad.xtri ←  $d - 1, 0$ 
12:   skX ← skMX $d-1,0$ 
13:   lvsX ← FL-SL-XMSSMTS.SkXToLvsX(skX, ps, ad)
14:   ad.typei ← xtrhType
15:   pkMX ← LvsToRt $h'$ (ps, ad, lvsX)
16:   return pkMX
17: procedure FL-SL-XMSSMTS.SigToPkMX( $m$ , sig,  $i$ , ps, ad)
18:   rt ←  $m$ 
19:   ad.xtri ←  $i$ 
20:   for  $j = 0, \dots, d - 1$  do
21:     ad.li, ad.xtri, ad.kpi ←  $j, \lfloor \text{ad.xtri}/l' \rfloor, \text{ad.xtri} \bmod l'$ 
22:     sigW, apX ← sig[ $j$ ]
23:     ad.typei ← chType
24:     pkW ← WOTS-TWS.SigToPkW(rt, sigW, ps, ad)
25:     ad.typei ← pkcoType
26:     lf ← PKCO(ps, ad, flatten(pkW))
27:     ad.typei ← xtrhType
28:     rt ← APToRt $h'$ (ps, ad, apX, lf, ad.kpi)
29:   return rt

```

there exist adversaries \mathcal{B}_0 , \mathcal{B}_1 , and \mathcal{B}_2 — each with approximately the same running time as \mathcal{A} — such that the following inequality holds.

$$\text{Adv}_{\text{FL-SL-XMSS}^{\text{MTS}}, \text{THFC}}^{\text{EUF-NAGCMA}}(\mathcal{A}) \leq \text{Adv}_{\text{WOTS-TW}^{\text{S}}, \text{THFC}, t_{\text{wtw}}}^{\text{M-EUF-GCMA}}(\mathcal{B}_0) + \text{Adv}_{\text{PKCO}, \text{THFC}, t_{\text{pkco}}}^{\text{SM-DT-TCR-C}}(\mathcal{B}_1) \\ + \text{Adv}_{\text{TRH}, \text{THFC}, t_{\text{trh}}}^{\text{SM-DT-TCR-C}}(\mathcal{B}_2)$$

Here, THFC denotes an arbitrary THF collection containing F, PKCO, and TRH. Furthermore, $t_{\text{wtw}} = \sum_{i=0}^{d-1} \text{nrtrees}(i) \cdot l'$, $t_{\text{pkco}} = \sum_{i=0}^{d-1} \text{nrtrees}(i) \cdot l'$, and $t_{\text{trh}} = \sum_{i=0}^{d-1} \text{nrtrees}(i) \cdot (l' - 1)$.

Concerning the M-EUF-GCMA property for WOTS-TW^S, it suffices to know the following for the upcoming discussion. This property considers a two-stage adversary that, only in its first stage, is given access to a WOTS-TW^S signing oracle and $[OC]_{\text{THFC}}$. In this first stage, the adversary can issue queries consisting of a message and an address to the signing oracle, receiving a WOTS-TW^S public key and signature (on the query's message) that were freshly constructed *using the query's address in any THF evaluations*. In its second stage, the adversary is asked to produce a fresh and valid forgery under one of the public keys received in the first stage. For further details about this property, see [6].

The formal verification of Theorem 3 proceeds by an exhaustive case analysis on the scenario where \mathcal{A} wins Game _{$\mathcal{A}, \text{FL-SL-XMSS}^{\text{MTS}}, \text{THFC}$} ^{EUF-NAGCMA}, bounding the proba-

Game $_{\mathcal{A}, \text{FL-SL-XMSS}^{\text{MT}^\$, \text{THFC}}}^{\text{EUF-NAGCMA}}$	
1:	$\text{ad} \leftarrow \text{ad}_z$
2:	$\text{ps} \leftarrow \mathcal{U}(\mathcal{PS})$
3:	$\text{OC}_{\text{THFC}}.\text{Init}(\text{ps})$
4:	$\text{ml} \leftarrow \mathcal{A}^{\text{OC}_{\text{THFC}}.\text{Query}}.\text{Choose}()$
5:	$\text{pk}, \text{sk} \leftarrow \text{FL-SL-XMSS}^{\text{MT}^\$}.\text{KeyGen}(\text{ps}, \text{ad})$
6:	$\text{sigl} \leftarrow []$
7:	for $i = 0 \dots \min(\text{ml} , l) - 1$ do
8:	$\text{sig} \leftarrow \text{FL-SL-XMSS}^{\text{MT}^\$}.\text{Sign}(\text{sk}, \text{ml}[i], i)$
9:	$\text{sigl} \leftarrow \text{sigl} \parallel \text{sig}$
10:	$m', \text{sig}', i' \leftarrow \mathcal{A}.\text{Forge}(\text{pk}, \text{sigl})$
11:	$\text{isValid} \leftarrow \text{FL-SL-XMSS}^{\text{MT}^\$}.\text{Verify}(\text{pk}, m', \text{sig}', i')$
12:	$\text{isFresh} \leftarrow m' \neq \text{ml}[i']$
13:	return $\text{isValid} \wedge \text{isFresh} \wedge 0 \leq i' < \text{ml} $

Fig. 15. EUF-NAGCMA game for $\text{FL-SL-XMSS}^{\text{MT}^\$}$.

bility of each of these cases by the relevant advantage term. Here, much of the reasoning related to the nature of the computations and the behavior of the reduction adversaries is analogous to the reasoning presented in the discussion on the formal verification for M-FORS $^\$$. As such, we do not elaborate as much here. In the ensuing, $G_{\mathcal{A}}^\top$ refers to the event $\text{Game}_{\mathcal{A}, \text{FL-SL-XMSS}^{\text{MT}^\$, \text{THFC}}}^{\text{EUF-NAGCMA}} = 1$.

Case Distinction for $G_{\mathcal{A}}^\top$ and Corresponding Bounds. First, remark that a valid EUF-NAGCMA forgery for $\text{FL-SL-XMSS}^{\text{MT}^\$}$ comprises a message m' , a signature sig' , and an index i' such that m' is fresh — which, in this context, means that m' is different from the message at index i' in the list of selected (and signed) messages — and sig' is a valid signature on m' under the considered public key $\text{pk} = (\text{pkMX}, \text{ps}, \text{ad})$ and i' . Recall that sig' is only valid if the candidate root pkMX' computed from m' , sig' , and i' equals the actual root pkMX . By the verification procedure, validity of the forgery implies that the values considered in the computation of pkMX' must at some point coincide with the corresponding values in the original computation of pkMX . As such, we can distinguish the following three exhaustive cases in the verification of the $\text{FL-SL-XMSS}^{\text{MT}^\$}$ forgery. In the first case, either (1) the initial WOTS-TW $^\$$ signature in the forgery is valid on m' or (2) we encounter an inner tree root that is different from the corresponding original root, but the associated WOTS-TW $^\$$ signature is valid. In the second case, we encounter a WOTS-TW $^\$$ public key (computed from a WOTS-TW $^\$$ signature in the forgery) that does not match the corresponding original public key, but the inner tree leaf resulting from the compression of the public key equals the corresponding original leaf. In the last case, we encounter an inner tree leaf that does not equal the corresponding original leaf, but the root computed with the associated authentication path is equal to the corresponding original root. Hereafter, in order, \mathcal{R}_W^A , \mathcal{R}_P^A , and \mathcal{R}_T^A denote the reduction adver-

saries considered in each case. Furthermore, E_W and E_P refer to the events capturing the first two cases, respectively.

On a high level, the constructed reduction adversaries all follow a similar approach. Specifically, in their first stage, the reduction adversaries run \mathcal{A} 's first stage, answering collection oracle queries via their own collection oracle, and obtain a list of messages to sign (ml). Then, the reduction adversaries construct a hypertree structure in line with FL-SL-XMSS^{MT}.KeyGen using the provided oracles. Here, the primary difference between the reduction adversaries concerns which oracle they employ to compute certain values: \mathcal{R}_W^A uses the signing oracle to obtain WOTS-TW^S public keys and signatures on the messages and roots of inner trees; \mathcal{R}_P^A uses the challenge oracle to compress the WOTS-TW^S public keys to inner tree leaves; and \mathcal{R}_T^A uses the challenge oracle to compute the inner tree nodes from their leaves. All reduction adversaries compute the (for them) remaining values using the collection oracle. In their second stage, the reduction adversaries sign the messages in ml, conforming to FL-SL-XMSS^{MT}.Sign, using the (hypertree) values obtained in their first stage. Subsequently, they run \mathcal{A} 's second stage, giving it their public key (i.e., hypertree root), the received public seed, the initialization address, and the list of signatures. Upon receiving the forgery from \mathcal{A} , the reduction adversaries find their forgery or collision and return this together with the associated index, winning their game. Importantly, as the reduction adversaries simulate the collection oracle for \mathcal{A} through their own collection oracle, the constraint on \mathcal{A} 's queries guarantees that no addresses between the reduction adversaries' oracles overlap, which is required to win their games. Concluding, we can bound $\Pr[G_A^\top \wedge E_W]$ by $\text{Adv}_{\text{WOTS-TW}^S, \text{THFC}, t_{\text{wtw}}}^{\text{M-EUF-GCMA}}(\mathcal{R}_W^A)$, $\Pr[G_A^\top \wedge \neg E_W \wedge E_P]$ by $\text{Adv}_{\text{PKCO}, \text{THFC}, t_{\text{pkco}}}^{\text{SM-DT-TCR-C}}(\mathcal{R}_P^A)$, and $\Pr[G_A^\top \wedge \neg E_W \wedge \neg E_P]$ by $\text{Adv}_{\text{TRH}, \text{THFC}, t_{\text{trh}}}^{\text{SM-DT-TCR-C}}(\mathcal{R}_T^A)$.

Final Result. Combining the acquired bounds and the fact that the sum of the considered probabilities equals $\text{Adv}_{\text{FL-SL-XMSS}^{\text{MT}^S}, \text{THFC}}^{\text{EUF-NAGCMA}}(\mathcal{A})$, Theorem 3 follows.

6 SPHINCS⁺

SPHINCS⁺ is essentially a straightforward extension of the *pseudorandom* versions of the constructions considered hitherto. Namely, a SPHINCS⁺ instance uses the KHF's MKG and SKG to generate pseudorandom values when necessary, rather than sampling and maintaining all of these values throughout. To this end, in addition to a public seed and address, it initializes and maintains a *message seed* and a *secret seed* used to index MKG and SKG, respectively. Furthermore, as alluded to above, it employs the pseudorandom versions of FORS^S (FORS), M-FORS^S (M-FORS), and FL-SL-XMSS^{MT^S} (FL-SL-XMSS^{MT}). Although not explicitly presented here, these pseudorandom versions are trivially obtained from their counterparts by replacing (1) the sampled (sequence of) values in the secret key by the message seed and secret seed, (2) evaluations of MKG^S by MKG (indexed by the message seed), and (3) references to sampled secret key values by the generation of these values through SKG (indexed by the secret

Listing 6 SPHINCS⁺

```

1: procedure SPHINCS+.Keygen()
2:   ad ← adz
3:   ms ← $ U(MS)
4:   ss ← $ U(SS)
5:   ps ← $ U(PS)
6:   pkS ← FL-SL-XMSSMT.SkMXToPkMX(ss, ps, ad)
7:   return (pkS, ps), (ms, ss, ps)
8: procedure SPHINCS+.Sign(sk := (ms, ss, ps), m)
9:   ad ← adz
10:  mk, sigF ← M-FORS.Sign((ms, ss, ps, ad), m)
11:  mc, i ← MCO(mk, m)
12:  ad.xtri, ad.kpi, ad.typei ← [i/l'], i mod l', ftrhtype
13:  pkF ← FORS.SigToPkF(mc, sigF, ps, ad)
14:  sigMX ← FL-SL-XMSSMT.Sign((ss, ps, ad), pkF, i)
15:  return mk, sigF, sigMX
16: procedure SPHINCS+.Verify(pk := (pkS, ps), m, sig := (mk, sigF, sigMX))
17:  ad ← adz
18:  mc, i ← MCO(mk, m)
19:  ad.xtri, ad.kpi, ad.typei ← [i/l'], i mod l', ftrhtype
20:  pkF' ← FORS.SigToPkF(mc, sigF, ps, ad)
21:  pkS' ← FL-SL-XMSSMT.SigToPkMX(pkF', sigMX, i, ps, ad)
22:  return pkS' = pkS

```

seed and an appropriately adjusted address). Lastly, the set of valid addresses for SPHINCS⁺ is the union of those for M-FORS and FL-SL-XMSS^{MT}.

Following the preceding, Listing 6 specifies SPHINCS⁺'s algorithms. Here, ad_z is an initialization address that, as per the official SPHINCS⁺ specification, has every associated index set to 0 (and the type index set to chType).

Security Property. As is customary for standalone signature schemes, we consider the conventional EUF-CMA security property for SPHINCS⁺. For completeness, this property and the corresponding oracle are given in Figures 16 and 17, respectively.

$\text{Game}_{\mathcal{A}, \text{SPHINCS}^+}^{\text{EUF-CMA}}$
<pre> 1 : (pk, sk) ← SPHINCS⁺.Keygen() 2 : O_{SPHINCS⁺}.Init(sk) 3 : m', sig' ← A^{O_{SPHINCS⁺}.Query}.Forge(pk) 4 : isValid ← SPHINCS⁺.Verify(pk, m, sig) 5 : isFresh ← m' ∉ O_{SPHINCS⁺}..M 6 : return isValid ∧ isFresh </pre>

Fig. 16. EUF-CMA game for SPHINCS⁺.

$\text{O}_{\text{SPHINCS}^+}$
<pre> vars sk, M Init(ski) 1 : sk, M ← ski, [] Query(m) 1 : sig ← SPHINCS⁺.Sign(sk, m) 2 : M ← M m 3 : return </pre>

Fig. 17. Oracle employed in EUF-CMA game for SPHINCS⁺.

Formal Verification. As Figure 12 portrays, we demonstrate that the EUF-CMA security of SPHINCS⁺ is implied by the EUF-CMA property of SPHINCS^{+§} and the PRF property of MKG and SKG. Furthermore, we show that the EUF-CMA security of SPHINCS^{+§} can be based on the EUF-CMA property of M-FORS[§] and the EUF-NAGCMA property of FL-SL-XMSS^{MT§}. In the ensuing exposition, we combine these two proof steps (without affecting the reasoning) for the sake of brevity. More formally, we consider the following security theorem.

Theorem 4 (EUF-CMA for SPHINCS⁺). *For any adversary \mathcal{A} , there exist adversaries \mathcal{B}_0 , \mathcal{B}_1 , \mathcal{B}_2 , and \mathcal{B}_3 — each with approximately the same running time as \mathcal{A} — such that the following inequality holds.*

$$\begin{aligned} \text{Adv}_{\text{SPHINCS}^+}^{\text{EUF-CMA}}(\mathcal{A}) &\leq \text{Adv}_{\text{SKG}}^{\text{PRF}}(\mathcal{B}_0) + \text{Adv}_{\text{MKG}}^{\text{PRF}}(\mathcal{B}_1) + \text{Adv}_{\text{M-FORS}^\S}^{\text{EUF-CMA}}(\mathcal{B}_2) \\ &\quad + \text{Adv}_{\text{FL-SL-XMSS}^{\text{MT}\S}, \text{THFC}}^{\text{EUF-NAGCMA}}(\mathcal{B}_3) \end{aligned}$$

Here, THFC denotes an arbitrary THF collection containing F, PKCO, TRCO, and TRH.

Unsurprisingly, the formal verification of this security theorem effectively performs two conceptual steps: The substitution of all pseudorandomness by actual randomness and, subsequently, the extraction of a forgery for one of the considered sub-constructions. In essence, the former step replaces SPHINCS⁺ by SPHINCS^{+§} (inherently replacing M-FORS and FL-SL-XMSS^{MT} by their “randomized” counterparts). The latter step shows that a valid EUF-CMA forgery for SPHINCS^{+§} contains a valid EUF-CMA forgery for M-FORS[§] or a valid EUF-NAGCMA forgery for FL-SL-XMSS^{MT§}, thereafter relating each case to the corresponding advantage. In the ensuing, $G_{\mathcal{A}}^\top$ denotes $\text{Game}_{\mathcal{A}, \text{SPHINCS}^+}^{\text{EUF-CMA}} = 1$.

Bound on $\left| \text{Adv}_{\text{SPHINCS}^+}^{\text{EUF-CMA}}(\mathcal{A}) - \text{Adv}_{\text{SPHINCS}^+}^{\text{EUF-CMA}}(\mathcal{A}) \right|$. Considering the differences between SPHINCS⁺ and SPHINCS^{+§}, the transition from the former to the latter basically comes down to replacing SKG and MKG by actual random functions with the appropriate domain and range, on top of some refactoring to maintain functional correctness (e.g., moving all evaluations of SKG to the key generation and storing the result in the secret key). In fact, for SKG, we can replace each evaluation by a pure random sampling because each provided input is unique. Thus, for both functions, we can straightforwardly construct a reduction adversary playing in the corresponding PRF game that perfectly simulates an execution of the EUF-CMA game (that \mathcal{A} is playing in) by substituting each evaluation of the considered function by an appropriate query to the provided PRF oracle. As such, we can bound $\left| \text{Adv}_{\text{SPHINCS}^+}^{\text{EUF-CMA}}(\mathcal{A}) - \text{Adv}_{\text{SPHINCS}^+}^{\text{EUF-CMA}}(\mathcal{A}) \right|$ by the sum of $\text{Adv}_{\text{SKG}}^{\text{PRF}}(\mathcal{R}_S^{\mathcal{A}})$ and $\text{Adv}_{\text{MKG}}^{\text{PRF}}(\mathcal{R}_M^{\mathcal{A}})$, where $\mathcal{R}_S^{\mathcal{A}}$ and $\mathcal{R}_M^{\mathcal{A}}$ are the relevant reduction adversaries.

Case Distinction for $G_{\mathcal{A}}^\top$ and Corresponding Bounds. First, observe that a valid EUF-CMA forgery for SPHINCS^{+§} consists of a message m' and a signature

$\text{sig}' = (\text{mk}', \text{sigF}', \text{sigMX}')$ such that m' is fresh and sig' is a valid signature for m' under the considered public key $\text{pk} = (\text{pkS}, \text{ps})$. Moreover, by the verification procedure of SPHINCS⁺^s, validity of sig' implies that sigMX' is a valid signature on the FORS^s public key pkF' derived from sigF' . Thus, if pkF' *does not* equal the corresponding original public key, it constitutes a different “message” than the original one signed by FL-SL-XMSS^{MTs} in the considered SPHINCS⁺^s instance. Otherwise, by definition, $(\text{mk}', \text{sigF}')$ is a valid M-FORS^s signature on m' , where m' is fresh. Indeed, the former case allows for the extraction of an EUF-NAGCMA forgery for FL-SL-XMSS^{MTs}; the latter case allows for the extraction of an EUF-CMA forgery for M-FORS^s. In the following, E_X and \mathcal{R}_X^A denote the event and reduction adversary for the former case; \mathcal{R}_M^A denotes the reduction adversary for the latter case ($\neg E_X$ suffices to capture the latter case).

Loosely speaking, the considered reduction adversaries follow a similar approach. Namely, both reduction adversaries construct a key pair for the sub-construction they *are not* an adversary for, using the provided collection oracle (FL-SL-XMSS^{MT})¹⁴ or public seed and address (M-FORS^s). Then, to simulate the signing oracle for \mathcal{A} , the reduction adversaries use the constructed key pair to create signatures for the corresponding sub-construction, use either the provided list of signatures (FL-SL-XMSS^{MT}) or their own signing oracle (M-FORS^s) to obtain signatures for the other sub-construction, and combine the signatures to construct the corresponding SPHINCS⁺ signature. Upon receiving the forgery from \mathcal{A} , they extract and return the relevant forgery, winning their own game. As a result, we can bound $\Pr[G_{\mathcal{A}}^{\top} \wedge E_X]$ by $\text{Adv}_{\text{FL-SL-XMSS}^{\text{MTs}}, \text{THFC}}^{\text{EUF-NAGCMA}}(\mathcal{R}_X^A)$ and $\Pr[G_{\mathcal{A}}^{\top} \wedge \neg E_X]$ by $\text{Adv}_{\text{M-FORS}^{\text{s}}}^{\text{EUF-CMA}}(\mathcal{R}_M^A)$.

Final Result. From the above results, Theorem 4 follows. At last, we can combine the security theorem regarding WOTS-TW^s in [6] with the security theorems considered in this work to acquire a bound on the EUF-CMA security of SPHINCS⁺ that is entirely based on the properties of the employed KHF and THFs. This completes the formal verification of the security of SPHINCS⁺.

Acknowledgments. We gratefully acknowledge discussions and support from the Formosa Crypto consortium. This research is supported by Amazon Web Services, as an Amazon Research Award supporting the Formosa Crypto consortium; by the InnovateUK ATI programme (10065634). Andreas Hülsing and Matthias Meijers are funded by an NWO VIDI grant (Project No. VI.Vidi.193.066).

Disclosure of Interests. All authors are members of the Formosa Crypto consortium. François Dupressoir is a member of the technical evaluation committee for PEPR en Cybersécurité.

¹⁴ Crucially, this means that \mathcal{R}_X^A *does not* query its collection oracle on addresses used in FL-SL-XMSS^{MT}, as required for the application of Theorem 3.

References

1. J. B. Almeida, M. Barbosa, G. Barthe, B. Grégoire, A. Koutsos, V. Laporte, T. Oliveira, and P.-Y. Strub. The last mile: High-assurance and high-speed cryptographic implementations. In *2020 IEEE Symposium on Security and Privacy*, pages 965–982, San Francisco, CA, USA, May 18–21, 2020. IEEE Computer Society Press.
2. J. B. Almeida, M. Barbosa, G. Barthe, B. Grégoire, V. Laporte, J.-C. Léchenet, T. Oliveira, H. Pacheco, M. Quaresma, P. Schwabe, A. Séré, and P.-Y. Strub. Formally verifying kyber episode IV: Implementation correctness. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2023(3):164–193, 2023.
3. J. B. Almeida, C. Baritel-Ruet, M. Barbosa, G. Barthe, F. Dupressoir, B. Grégoire, V. Laporte, T. Oliveira, A. Stoughton, and P.-Y. Strub. Machine-checked proofs for cryptographic standards: Indifferentiability of sponge and secure high-assurance implementations of SHA-3. In L. Cavallaro, J. Kinder, X. Wang, and J. Katz, editors, *ACM CCS 2019: 26th Conference on Computer and Communications Security*, pages 1607–1622, London, UK, Nov. 11–15, 2019. ACM Press.
4. J. Alwen, B. Blanchet, E. Hauck, E. Kiltz, B. Lipp, and D. Riepel. Analysing the HPKE standard. In A. Canteaut and F.-X. Standaert, editors, *Advances in Cryptology – EUROCRYPT 2021, Part I*, volume 12696 of *Lecture Notes in Computer Science*, pages 87–116, Zagreb, Croatia, Oct. 17–21, 2021. Springer, Heidelberg, Germany.
5. M. Barbosa, G. Barthe, K. Bhargavan, B. Blanchet, C. Cremers, K. Liao, and B. Parno. SoK: Computer-aided cryptography. In *2021 IEEE Symposium on Security and Privacy*, pages 777–795, San Francisco, CA, USA, May 24–27, 2021. IEEE Computer Society Press.
6. M. Barbosa, F. Dupressoir, B. Grégoire, A. Hülsing, M. Meijers, and P.-Y. Strub. Machine-checked security for XMSS as in RFC 8391 and SPHINCS⁺. In H. Handschuh and A. Lysyanskaya, editors, *Advances in Cryptology – CRYPTO 2023*, volume 14085, pages 421–454, Cham, Aug. 2023. Springer Nature Switzerland.
7. G. Barthe, B. Grégoire, S. Héraud, and S. Zanella Béguelin. Computer-aided security proofs for the working cryptographer. In P. Rogaway, editor, *Advances in Cryptology – CRYPTO 2011*, volume 6841 of *Lecture Notes in Computer Science*, pages 71–90, Santa Barbara, CA, USA, Aug. 14–18, 2011. Springer, Heidelberg, Germany.
8. D. J. Bernstein and A. Hülsing. Decisional second-preimage resistance: When does SPR imply PRE? In S. D. Galbraith and S. Moriai, editors, *Advances in Cryptology – ASIACRYPT 2019, Part III*, volume 11923 of *Lecture Notes in Computer Science*, pages 33–62, Kobe, Japan, Dec. 8–12, 2019. Springer, Heidelberg, Germany.
9. D. J. Bernstein, A. Hülsing, S. Kölbl, R. Niederhagen, J. Rijneveld, and P. Schwabe. The SPHINCS⁺ signature framework. In L. Cavallaro, J. Kinder, X. Wang, and J. Katz, editors, *ACM CCS 2019: 26th Conference on Computer and Communications Security*, pages 2129–2146, London, UK, Nov. 11–15, 2019. ACM Press.
10. D. J. Bernstein and T. Lange. Post-quantum cryptography. *Nature*, 549(7671):188–194, Sep. 2017.
11. K. Cohn-Gordon, C. Cremers, B. Dowling, L. Garratt, and D. Stebila. A formal security analysis of the Signal messaging protocol. *Journal of Cryptology*, 33(4):1914–1983, Oct. 2020.
12. D. Cooper, D. Apon, Q. Dang, M. Davidson, M. Dworkin, and C. Miller. Recommendation for stateful hash-based signature schemes, 2020-10-29 00:10:00 2020.

13. C. Cremers, M. Horvat, J. Hoyland, S. Scott, and T. van der Merwe. A comprehensive symbolic analysis of TLS 1.3. In B. M. Thuraisingham, D. Evans, T. Malkin, and D. Xu, editors, *ACM CCS 2017: 24th Conference on Computer and Communications Security*, pages 1773–1788, Dallas, TX, USA, Oct. 31 – Nov. 2, 2017. ACM Press.
14. A. Fedorov, E. Kiktenko, and M. Kudinov. [pqc-forum] round 3 official comment: SPHINCS⁺. <https://csrc.nist.gov/CSRC/media/Projects/post-quantum-cryptography/documents/round-3/official-comments/Sphincs-Plus-round3-official-comment.pdf>, 2020. Accessed: 22-05-2024.
15. E. Grumbling and M. Horowitz. *Quantum Computing: Progress and Prospects*. National Academies of Sciences, Engineering, and Medicine. The National Academies Press, 1st edition, Apr. 2019.
16. A. Hülsing, D. Butin, S.-L. Gazdag, J. Rijneveld, and A. Mohaisen. XMSS: eXtended Merkle Signature Scheme. RFC 8391, May 2018.
17. A. Hülsing and M. A. Kudinov. Recovering the tight security proof of SPHINCS⁺. In S. Agrawal and D. Lin, editors, *Advances in Cryptology – ASIACRYPT 2022, Part IV*, volume 13794 of *Lecture Notes in Computer Science*, pages 3–33, Taipei, Taiwan, Dec. 5–9, 2022. Springer, Heidelberg, Germany.
18. A. Hülsing, M. Meijers, and P-Y. Strub. Formal verification of Saber’s public-key encryption scheme in EasyCrypt. In Y. Dodis and T. Shrimpton, editors, *Advances in Cryptology – CRYPTO 2022, Part I*, volume 13507 of *Lecture Notes in Computer Science*, pages 622–653, Santa Barbara, CA, USA, Aug. 15–18, 2022. Springer, Heidelberg, Germany.
19. A. Hülsing, L. Rausch, and J. Buchmann. Optimal parameters for XMSS^{MT}. In A. Cuzzocrea, C. Kittl, D. E. Simos, E. Weippl, and L. Xu, editors, *Security Engineering and Intelligence Informatics*, pages 194–208, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.
20. A. Hülsing, J. Rijneveld, and F. Song. Mitigating multi-target attacks in hash-based signatures. In C.-M. Cheng, K.-M. Chung, G. Persiano, and B.-Y. Yang, editors, *PKC 2016: 19th International Conference on Theory and Practice of Public Key Cryptography, Part I*, volume 9614 of *Lecture Notes in Computer Science*, pages 387–416, Taipei, Taiwan, Mar. 6–9, 2016. Springer, Heidelberg, Germany.
21. M. A. Kudinov, E. O. Kiktenko, and A. K. Fedorov. Security analysis of the w-ots⁺ signature scheme: Updating security bounds. *CoRR*, abs/2002.07419, 2020.
22. D. McGrew, P. Kampanakis, S. Fluhrer, S.-L. Gazdag, D. Butin, and J. Buchmann. State management for hash-based signatures. Cryptology ePrint Archive, Report 2016/357, 2016. <https://eprint.iacr.org/2016/357>.
23. M. Mosca and M. Piani. Quantum threat timeline report 2023. Technical report, Global Risk Insitute, dec 2023.
24. NIST. National Institute for Standards and Technology. announcing request for nominations for public-key post-quantum cryptographic algorithms., Dec. 2016. <https://csrc.nist.gov/News/2016/Public-Key-Post-Quantum-Cryptographic-Algorithms>.
25. NIST. National Institute for Standards and Technology. PQC standardization process: Announcing four candidates to be standardized, plus fourth round candidates., Mar. 2022. <https://csrc.nist.gov/News/2022/pqc-candidates-to-be-standardized-and-round-4>.



MinRank Gabidulin Encryption Scheme on Matrix Codes

Nicolas Aragon¹(✉), Alain Couvreur^{2,3,5}, Victor Dıyserın^{4,5}, Philippe Gaborit¹,
and Adrien Vinçotte¹

¹ XLIM, Université de Limoges, Limoges, France
{nicolas.aragon,philippe.gaborit,adrien.vincotte}@unilim.fr

² Inria, Le Chesnay-Rocquencourt, France
alain.couvreur@inria.fr

³ LIX, École Polytechnique, Palaiseau, France

⁴ LTCI, Télécom Paris, Paris, France
victor.dyserın@telecom-paris.fr

⁵ Institut Polytechnique de Paris, Paris, France

Abstract. The McEliece scheme is a generic frame introduced in [28], which allows to use any error correcting code for which there exists an efficient decoding algorithm to design an encryption scheme by hiding the generator matrix of the code. Similarly, the Niederreiter frame, introduced in [30], is the dual version of the McEliece scheme, and achieves smaller ciphertexts. In the present paper, we propose a generalization of the McEliece and the Niederreiter frame to matrix codes and the MinRank problem, that we apply to Gabidulin matrix codes (Gabidulin rank codes considered as matrix codes). The masking we consider consists in starting from a rank code \mathcal{C} , computing a matrix version of \mathcal{C} and then concatenating a certain number of rows and columns to the matrix code version of the rank code \mathcal{C} before applying an isometry for matrix codes, i.e. right and left multiplications by fixed random matrices. The security of the schemes relies on the MinRank problem to decrypt a ciphertext, and the structural security of the scheme relies on the new EGMC-Indistinguishability problem that we introduce and that we study in detail. The main structural attack that we propose consists in trying to recover the masked linearity over the extension field which is lost during the masking process. Overall, starting from Gabidulin codes, we obtain a very appealing trade off between the size of the ciphertext and the size of the public key. For 128 bits of security we propose parameters ranging from ciphertexts of size 65 B (and public keys of size 98 kB) to ciphertexts of size 138 B (and public keys of size 41 kB). For 256 bits of security, we can obtain ciphertext as low as 119 B, or public key as low as 87 kB. Our new approach permits to achieve a better trade-off between ciphertexts and public key than the classical McEliece scheme instantiated with Goppa codes.

1 Introduction

Matrix Codes and Vector Codes. Introduced in 1951 in [26], before the well-known notion of Hamming metric, matrix codes are \mathbb{F}_q -linear subspaces of

the matrix space $\mathbb{F}_q^{m \times n}$, endowed with the rank metric. The generic decoding problem in matrix codes relies on the difficulty to solving an instance of the MinRank problem, which is known to be NP-complete. A rank metric code of length n over the extension field \mathbb{F}_{q^m} is an \mathbb{F}_{q^m} -linear vector subspace of $\mathbb{F}_{q^m}^n$, endowed with the rank metric. It is possible to turn a vector code into a matrix code by considering all vectors of a vector code as matrices over \mathbb{F}_q by writing of the extension field \mathbb{F}_{q^m} as a vector space of dimension m over \mathbb{F}_q . The main advantage of vector codes compared to matrix codes lies in the fact that the linearity over the extension field permits to achieve smaller description of the code (division by a factor m , the degree of the field extension).

Code-Based Cryptography. There exist two approaches for code-based encryption: in the first one, the McEliece frame, the public key consists in a masking of a structured code, for instance using Goppa codes [28] or MDPC codes [29]. In that case the hidden code is used both for encryption and decryption. The second one is the Alekhovich approach [5] and its variants like [2, 3] in which there is no structural masking and for which two types of codes are considered, a random (or random quasi-cyclic) code for encryption and another code for decryption. The main advantage of the second approach is that it avoids structural attacks, so it is semantically stronger in terms of security. However, the price to pay is a larger ciphertext (typically quadratic in the security parameter), whereas the first approach permits to obtain much smaller ciphertexts (linear in the security parameter) at a cost of a very large public key. For instance for 128 bits of security for McEliece it is possible to get ciphertexts of size of order 100B, whereas for HQC the ciphertext is of order 1500kB. There exist applications for which having a very small ciphertext may be of interest.

Structural Attacks and Distinguishers on McEliece-Like Schemes. The main tool for structural attacks is to consider a distinguisher for the underlying masked code. While it is really interesting to consider Reed-Solomon codes or Gabidulin codes – their rank metric analog – for encryption because of their very good decoding properties, these codes are very highly structured which makes them easy to distinguish from random codes. For Reed-Solomon codes, the main distinguisher is the square code distinguisher of [15] which states that the square code of an $[n, k]$ Reed-Solomon code is a code of dimension $2k - 1$, whereas for a random code its dimension is close to $\min(n, \frac{k(k-1)}{2})$. There exists an analog distinguisher for Gabidulin codes [33], starting from an $[m, k]$ Gabidulin code over \mathbb{F}_{q^m} with generator matrix G , and considering G^q the matrix in which one applies the Frobenius action $x \mapsto x^q$ to all entries of G . The code whose generator matrix is a vertical concatenation of G and G^q has dimension $k + 1$, whereas for a random code it would be $2k$. These two distinguishers were used to break systems in which the masking consisted in adding random columns (and then applying an isometry) to Reed-Solomon codes [15, 35] or to Gabidulin codes [31, 33].

The Case of the Original Goppa-McEliece Scheme. The main masking which is still resistant to structural attacks is the case of the original Goppa-McEliece scheme. This case can be seen as considering generalized Reed-Solomon codes over an extension field \mathbb{F}_{2^m} , then considering its \mathbb{F}_2 subcodes. The effect of considering the \mathbb{F}_2 subcode is that it breaks the structure over the \mathbb{F}_{2^m} extension and hence the direct application of a Reed-Solomon distinguisher, at least in the case where the rate of the code is not close to 1, which is the general case (see also [17]).

Contributions. Inspired by the previous situation we consider the case of Gabidulin codes in which we want to break the structure over the extension on which relies the Gabidulin distinguisher. Indeed, in order to apply the Overbeck distinguisher we need to consider the application of the Frobenius map which only makes sense over an extension \mathbb{F}_{q^m} and not directly on the base field \mathbb{F}_q . We consider the following masking: we start from a matrix code, which can decode up to errors of rank r , we then add random rows and columns and apply an isometry for matrix codes (multiplying on the left and on the right by two fixed random invertible matrices). Starting from a vector code \mathcal{C} , we call *enhanced code* the transformation which consists in turning the vector code \mathcal{C} into a matrix code and then applying the previous masking. Since it is a masking, there always exists the possibility of a structural attack but the idea is that considering matrix version of the vector codes with our masking (in the spirit of the original Goppa-McEliece scheme) permits to avoid an efficient action of a distinguisher.

The main contributions of the paper are the following:

- we describe a general encryption McEliece-like frame for matrix codes over the MinRank problem,
- we propose a general masking for matrix codes that we apply on a matrix code version of Gabidulin codes,
- we study in detail possible distinguishers for solving the Enhanced Gabidulin Matrix Code (EGMC) distinguishing problem that we introduce for our scheme.

In terms of distinguisher the best attack that we obtain for the vector code we consider consists in trying to dismiss the action of the addition of random rows and columns on the matrix code in order to test for its underlying linearity over the extension field. In terms of parameters, our new approach permits to obtain an alternative scheme to the classic McEliece scheme with very small ciphertexts and even smaller public keys than in the classic McEliece scheme. For 128 bits of security, we can achieve a size of 65 B for the ciphertext and 98 kB for the public key, or 138 B for the ciphertext and 41 kB for the public key. These results compare very well to classic McEliece's parameters [4]) yielding ciphertext of 96 B and public keys of 261 kB. For 256 bits of security, we can obtain ciphertext as low as 119 B, or public key as low as 87 kB.

Organization of the Paper. The paper is organized as follows. Section 2 gives an introduction and preliminaries, Sect. 3 defines the masking transformations we introduce and the attached distinguishing problems, Sects. 4 and 5 describe the new encryption schemes we propose, Sect. 6 examines in detail different types of possible distinguishers and Sect. 7 details parameters of our schemes.

2 Preliminaries

2.1 Public Key Encryption

Definition 1 (PKE). A public key encryption scheme PKE consists of three polynomial time algorithms:

- **KeyGen** (1^λ): for a security parameter λ , generates a public key pk and a secret key sk .
- **Encrypt**(pk, \mathbf{m}): outputs a ciphertext \mathbf{c} given the message \mathbf{m} and the public key pk .
- **Decrypt**(sk, \mathbf{c}): outputs the plaintext \mathbf{m} of the encrypted ciphertext \mathbf{c} , or \perp .

We require a PKE scheme to be correct: for every pair (pk, sk) generated by **KeyGen** and message \mathbf{m} , we should have $\Pr(\text{Decrypt}(\text{sk}, \text{Encrypt}(\text{pk}, \mathbf{m})) = \mathbf{m}) = 1 - \text{negl}(\lambda)$, where negl is a negligible function.

There exist several notions of security for PKE schemes. The encryption schemes we propose here achieve OW-CPA security.

Definition 2 (OW-CPA security). Let $\text{PKE} = (\text{KeyGen}, \text{Encrypt}, \text{Decrypt})$ a PKE scheme, \mathcal{A} an adversary against PKE, and λ a level of security. We define the OW-CPA game:

- **Challenge.** The challenger generates $(\text{pk}, \text{sk}) \leftarrow \text{KeyGen}(1^\lambda)$, samples \mathbf{m} from the set of messages and computes $\mathbf{c} \leftarrow \text{Encrypt}(\text{pk}, \mathbf{m})$. He sends (\mathbf{c}, pk) to the adversary \mathcal{A} .
- **Output.** \mathcal{A} outputs the guessing message $\tilde{\mathbf{m}}$. \mathcal{A} wins if $\tilde{\mathbf{m}} = \mathbf{m}$.

PKE is OW-CPA secure if for any PPT adversary \mathcal{A} , the probability that \mathcal{A} wins the game is negligible in λ .

To prove the security of our PKE, we need to formally define the notion of indistinguishable distributions.

Definition 3 (Distinguisher). Let S a set of elements, \mathcal{P}_1 and \mathcal{P}_2 two probability distributions on S . A distinguisher \mathcal{D} for the distributions \mathcal{P}_1 and \mathcal{P}_2 is an algorithm that takes in input an element of S , and outputs a bit. Its advantage is defined by:

$$\text{Adv}(\mathcal{D}) = \left| \Pr(\mathcal{D}(x) = 1 | x \leftarrow \mathcal{P}_1) - \Pr(\mathcal{D}(x) = 1 | x \leftarrow \mathcal{P}_2) \right|.$$

We say that the distributions \mathcal{P}_1 and \mathcal{P}_2 are indistinguishable if there exists no polynomial time distinguisher with non negligible advantage.

2.2 Matrix Codes

Definition 4 (γ -expansion). Let $\gamma = (\gamma_1, \dots, \gamma_m)$ be an \mathbb{F}_q -basis of \mathbb{F}_{q^m} . The γ -expansion of an element in \mathbb{F}_{q^m} to a vector in \mathbb{F}_q^m is defined as the application:

$$\Psi_\gamma : x \in \mathbb{F}_{q^m} \mapsto (x_1, \dots, x_m) \in \mathbb{F}_q^m$$

such that $x = \sum_{i=1}^m x_i \gamma_i$.

Ψ_γ extends naturally to a word $\mathbf{x} \in \mathbb{F}_{q^m}^n$ and turns it into a matrix $\Psi_\gamma(\mathbf{x}) \in \mathbb{F}_q^{m \times n}$, by writing in columns the coordinates of each element in basis γ in column. We denote as $\mathcal{B}(\mathbb{F}_{q^m})$ the set of \mathbb{F}_q -bases of \mathbb{F}_{q^m} .

Definition 5 (Vector codes with rank metric). A vector code \mathcal{C}_{vec} is an \mathbb{F}_{q^m} -subspace of $\mathbb{F}_{q^m}^n$ endowed with the rank metric. The weight of a vector $\mathbf{x} \in \mathbb{F}_{q^m}^n$ is the rank of the matrix $\Psi_\gamma(\mathbf{x})$, for a basis $\gamma \in \mathcal{B}(\mathbb{F}_{q^m})$. The weight of a vector is independent of the choice of the basis γ .

Equivalently, the rank of a vector \mathbf{x} is the dimension of the \mathbb{F}_q -vector space spanned by its coordinates, that is the support of \mathbf{x} . We denote the weight of a vector \mathbf{x} by:

$$\|\mathbf{x}\| \stackrel{\text{def}}{=} \text{rank}(\Psi_\gamma(\mathbf{x})) = \dim(\langle x_1, \dots, x_n \rangle_q)$$

Definition 6 (Matrix codes). A matrix code \mathcal{C}_{mat} is an \mathbb{F}_q -subspace of $\mathbb{F}_q^{m \times n}$ endowed with the rank metric.

An \mathbb{F}_{q^m} -linear vector code \mathcal{C}_{vec} of parameters $[n, k]_{q^m}$ can be turned into a matrix code of size $m \times n$ and dimension mk , defined as:

$$\Psi_\gamma(\mathcal{C}_{vec}) := \{\Psi_\gamma(\mathbf{x}) \mid \mathbf{x} \in \mathcal{C}_{vec}\}.$$

Let $\gamma \in \mathcal{B}(\mathbb{F}_{q^m})$, and $(\mathbf{v}_i)_{i \in \{1, \dots, k\}}$ be an \mathbb{F}_{q^m} -basis of \mathcal{C}_{vec} , then an \mathbb{F}_q -basis of $\Psi_\gamma(\mathcal{C}_{vec})$ is given by:

$$\{\Psi_\gamma(b\mathbf{v}_i) \mid b \in \mathbb{F}_q, i \in \{1, \dots, k\}\}.$$

Definition 7 (Equivalent matrix codes). Two matrix codes \mathcal{C}_{mat} and \mathcal{D}_{mat} are said to be equivalent if there exist two matrices $\mathbf{P} \in \mathbf{GL}_m(\mathbb{F}_q)$ and $\mathbf{Q} \in \mathbf{GL}_n(\mathbb{F}_q)$ such that $\mathcal{D}_{mat} = \mathbf{P}\mathcal{C}_{mat}\mathbf{Q}$. If $\mathbf{P} = \mathbf{I}_m$ (resp. $\mathbf{Q} = \mathbf{I}_n$), \mathcal{C}_{mat} and \mathcal{D}_{mat} are said to be right equivalent (resp. left equivalent).

As seen above, the transformation of a vector code into a matrix code is dependent on the choice of the basis γ . However, two different bases produce left-equivalent codes. For two bases β and γ , if we denote \mathbf{P} the transition matrix between β and γ , we get:

$$\Psi_\gamma(\mathcal{C}_{vec}) = \mathbf{P}\Psi_\beta(\mathcal{C}_{vec})$$

Definition 8 (Folding). *The application Fold turns a vector $\mathbf{v} = (\mathbf{v}_1 \parallel \dots \parallel \mathbf{v}_n) \in \mathbb{F}_q^{mn}$ such that each $\mathbf{v}_i \in \mathbb{F}_q^n$, into the matrix $\text{Fold}(\mathbf{v}) \stackrel{\text{def}}{=} (\mathbf{v}_1^t \parallel \dots \parallel \mathbf{v}_n^t) \in \mathbb{F}_q^{m \times n}$. The inverse map which turns a matrix into a vector is denoted by Unfold.*

The map Unfold allows to define a matrix code \mathcal{C}_{mat} thanks to a generator matrix $\mathbf{G} \in \mathbb{F}_q^{mk \times mn}$, whose rows are an unfolded set of matrices which forms a basis of \mathcal{C}_{mat} . This characterization allows to construct the associated parity check matrix \mathbf{H} , and define the dual of a matrix code: one can simply define \mathcal{C}_{mat}^\perp as the code generated by matrices equal to the folded rows of \mathbf{H} . A more formal definition follows:

Definition 9 (Dual of a matrix code). *Let \mathcal{C}_{mat} be a matrix code of size $m \times n$ and dimension K . Its dual is the matrix code of size $m \times n$ and dimension $mn - K$:*

$$\mathcal{C}_{mat}^\perp = \{ \mathbf{Y} \in \mathbb{F}_q^{m \times n} \mid \forall \mathbf{X} \in \mathcal{C}_{mat} \text{tr}(\mathbf{X}\mathbf{Y}^t) = 0 \}.$$

2.3 MinRank Problem

The MinRank problem can be seen as the Rank Decoding problem adapted to matrix codes:

Definition 10 (MinRank problem). *Given as input matrices $\mathbf{Y}, \mathbf{M}_1, \dots, \mathbf{M}_k \in \mathbb{F}_q^{m \times n}$, the $\text{MinRank}(q, m, n, k, r)$ problem asks to find $x_1, \dots, x_k \in \mathbb{F}_q$ and $\mathbf{E} \in \mathbb{F}_q^{m \times n}$ with $\text{rank } \mathbf{E} \leq r$ such that $\mathbf{Y} = \sum_{i=1}^k x_i \mathbf{M}_i + \mathbf{E}$.*

The decoding problem for a matrix code of size $m \times n$ and dimension K is exactly the $\text{MinRank}(q, m, n, K, r)$ problem. While the MinRank problem is well known, it is more uncommon to present this decoding problem from the syndrome decoding point of view.

Let be $\mathbf{X} = \sum_{i=1}^K x_i \mathbf{M}_i \in \mathcal{C}_{mat}$ with basis the set of (\mathbf{M}_i) and $\mathbf{E} \in \mathbb{F}_q^{m \times n}$ an error of small weight. The code \mathcal{C}_{mat} can be represented as a vector code over \mathbb{F}_q , with generator matrix $\mathbf{G} \in \mathbb{F}_q^{K \times mn}$ has for rows the vectors $\text{Unfold}(\mathbf{M}_i)$. A word to decode would have the form: $\text{Unfold}(\mathbf{X} + \mathbf{E}) = \sum_{i=1}^K x_i \text{Unfold}(\mathbf{M}_i) + \text{Unfold}(\mathbf{E})$, and the weight of the error is defined as the rank of \mathbf{E} .

Let $\mathbf{H} \in \mathbb{F}_q^{(mn-K) \times mn}$ be a parity-check matrix of the dual code. We denote by $(\mathbf{h}_i)_{1 \leq i \leq mn}$ its columns. Let be $\mathbf{x} \in \mathbb{F}_q^{mn}$ a noisy codeword with error $\mathbf{e} \in \mathbb{F}_q^{mn}$. The weight of \mathbf{e} is the rank of the matrix $\text{Fold}(\mathbf{e})$. Since one has reduced itself to a vector code, one can define the syndrome associated to an error in the same way:

$$\mathbf{s} = \mathbf{x}\mathbf{H}^t = \mathbf{e}\mathbf{H}^t = \sum_{i=1}^{mn} e_i \mathbf{h}_i^t.$$

One can deduce the associated problem:

Definition 11 (MinRank-Syndrome problem). *Given as input vectors $\mathbf{s}, \mathbf{v}_1, \dots, \mathbf{v}_k \in \mathbb{F}_q^{nm-k}$, the MinRank – Syndrome(q, m, n, k, r) problem asks to find $(e_1, \dots, e_{nm}) \in \mathbb{F}_q^{nm}$ with $\text{rank Fold}(\mathbf{e}) \leq r$ such that $\mathbf{s} = \sum_{i=1}^k e_i \mathbf{v}_i$.*

The two previous problems are equivalent for the very same reasons that the decoding problem and the syndrome decoding problem are in the vector codes context.

2.4 Gabidulin Codes

Gabidulin codes were introduced by Ernst Gabidulin in 1985 [19]. These vector codes can be seen as the analog in rank metric of the Reed-Solomon codes, but for which a codeword is a set of evaluation points of a q -polynomial rather than a standard polynomial.

Definition 12 (q -polynomial). *A q -polynomial of q -degree r is a polynomial in $\mathbb{F}_{q^m}[X]$ of the form:*

$$P(X) = \sum_{i=0}^r p_i X^{q^i} \quad \text{with } p_r \neq 0.$$

For a q -polynomial P , we denote by $\deg_q P$ its q -degree.

A q -polynomial is also called *linearized polynomial* since it induces an \mathbb{F}_q -linear application due to the linearity of the Frobenius endomorphism $x \mapsto x^q$.

Definition 13 (Gabidulin code). *Let $k, m, n \in \mathbb{N}$, such that $k \leq n \leq m$. Let $\mathbf{g} = (g_1, \dots, g_n) \in \mathbb{F}_{q^m}^n$ a vector of \mathbb{F}_q -linearly independent elements of \mathbb{F}_{q^m} . The Gabidulin code $\mathcal{G}_{\mathbf{g}}(n, k, m)$ is the vector code of parameters $[n, k]_{q^m}$ defined by:*

$$\mathcal{G}_{\mathbf{g}}(n, k, m) = \{P(\mathbf{g}) \mid \deg_q P < k\},$$

where $P(\mathbf{g}) = (P(g_1), \dots, P(g_n))$ and P is a q -polynomial.

The vector \mathbf{g} is said to be an *evaluation vector* of the Gabidulin code $\mathcal{G}_{\mathbf{g}}(n, k, m)$.

Gabidulin codes are popular in cryptography because they benefit from a very efficient decoding algorithm, allowing to correct errors of rank weight up to $\lfloor \frac{n-k}{2} \rfloor$ [19]. However, their strong structure makes them difficult to hide.

2.5 The GPT Cryptosystem

It has been proposed in [20] to concatenate to a generator matrix a random matrix, in order to mask a Gabidulin code more efficiently. Introduced in 1991 by Gabidulin, Paramonov and Tretjakov, the GPT Cryptosystem is an adaptation of the McEliece frame to the rank metric. The first versions having been attacked by Gibson [22, 23] and by Overbeck [33] whose attack is detailed in Sect. 6.1.

We subsequently propose a new version of the scheme to prevent the Overbeck attack. In our scheme, we propose to turn a Gabidulin code into a matrix code before adding some random rows and columns, and multiplying it by a secret invertible matrix.

3 Enhanced Matrix Codes Transformation

We present a general construction which defines a transformation containing a trapdoor and allowing to mask a secret matrix code.

Definition 14 (Random Rows and Columns matrix code transformation). Let $m, n, K, \ell_1, \ell_2 \in \mathbb{N}$. Let \mathcal{C}_{mat} be a matrix code of size $m \times n$ and dimension K on \mathbb{F}_q . Let $\mathcal{B} = (\mathbf{A}_1, \dots, \mathbf{A}_K)$ be a basis of \mathcal{C}_{mat} . The Random Rows and Columns matrix code transformation consists in sampling uniformly at random the following matrices: $\mathbf{P} \xleftarrow{\$} \mathbf{GL}_{m+\ell_1}(\mathbb{F}_q)$, $\mathbf{Q} \xleftarrow{\$} \mathbf{GL}_{n+\ell_2}(\mathbb{F}_q)$ and K random matrices: $\mathbf{R}_i \xleftarrow{\$} \mathbb{F}_q^{m \times \ell_2}$, $\mathbf{R}'_i \xleftarrow{\$} \mathbb{F}_q^{\ell_1 \times m}$ and $\mathbf{R}''_i \xleftarrow{\$} \mathbb{F}_q^{\ell_1 \times \ell_2}$; and defining the matrix code whose basis is:

$$\mathcal{RB} = \left(\mathbf{P} \begin{pmatrix} \mathbf{A}_1 & \mathbf{R}_1 \\ \mathbf{R}'_1 & \mathbf{R}''_1 \end{pmatrix} \mathbf{Q}, \dots, \mathbf{P} \begin{pmatrix} \mathbf{A}_K & \mathbf{R}_K \\ \mathbf{R}'_K & \mathbf{R}''_K \end{pmatrix} \mathbf{Q} \right).$$

In particular, we can apply this construction to Gabidulin codes turned into matrix codes:

Definition 15 (Enhanced Gabidulin matrix code). Let \mathcal{G} be a Gabidulin code $[n, k, r]$ on \mathbb{F}_{q^m} , γ be a \mathbb{F}_q -basis of \mathbb{F}_{q^m} . We recall that Ψ_γ turns a vector $\mathbf{x} \in \mathbb{F}_{q^m}$ into a matrix $\Psi_\gamma(\mathbf{x})$ whose columns are coordinates of coefficients of \mathbf{x} in the basis γ (see Definition 4). An Enhanced Gabidulin matrix code $\mathcal{EG}_g(n, k, m, \ell_1, \ell_2)$ is the matrix code $\Psi_\gamma(\mathcal{G})$ on which we apply the Random Rows and Columns matrix code transformation presented in Definition 14.

Duality. The dual of an enhanced Gabidulin matrix code can be described as follows. Start from an $[n, k]$ Gabidulin code \mathcal{G} over \mathbb{F}_{q^m} . It is well-known that the dual of the vector code \mathcal{G}^\perp for the Euclidean inner product is an $[n, n - k]$ Gabidulin code over \mathbb{F}_{q^m} . Moreover, given an basis γ for \mathbb{F}_{q^m} and denoting by γ' the dual basis with respect to the trace inner product in \mathbb{F}_{q^m} , then from [34, Thm. 21], we get

$$\Psi_\gamma(\mathcal{G})^\perp = \Psi_{\gamma'}(\mathcal{G}^\perp).$$

In short, the dual of the matrix code associated to \mathcal{G} as defined in Definition 9 is a matrix code associated to a Gabidulin code.

Proposition 1. *Following the notation of Definitions 14 and 15, let $\mathbf{B}_1, \dots, \mathbf{B}_{m(n-k)}$ be an \mathbb{F}_q -basis of $\Psi_\gamma(\mathcal{G})^\perp$. Let $\mathcal{V} \subseteq \mathbb{F}_q^{(m+\ell_1)(n+\ell_2)}$ be defined as*

$$\mathcal{V} \stackrel{\text{def}}{=} \left\{ \begin{pmatrix} \mathbf{R} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix} \mid \mathbf{R} \in \mathbb{F}_q^{m \times n} \right\}.$$

Then, there exists a space $\mathcal{W} \subseteq \mathbb{F}_q^{(m+\ell_1) \times (n+\ell_2)}$ such that $\mathcal{E}\mathcal{G}_g(n, k, m, \ell_1, \ell_2)^\perp = (\mathbf{P}^t)^{-1} \mathcal{C}_{mat} (\mathbf{Q}^t)^{-1}$ and

$$\mathcal{C}_{mat} = \text{Span}_{\mathbb{F}_q} \left\{ \begin{pmatrix} \mathbf{B}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix}, \dots, \begin{pmatrix} \mathbf{B}_{m(n-k)} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix} \right\} \oplus \mathcal{W}.$$

Moreover, \mathcal{W} is a complement subspace of \mathcal{V} in $\mathbb{F}_q^{(m+\ell_1) \times (n+\ell_2)}$.

Proof. Denote by \mathcal{C}_{mat}^0 , the space

$$\mathcal{C}_{mat}^0 \stackrel{\text{def}}{=} \text{Span}_{\mathbb{F}_q} \left\{ \begin{pmatrix} \mathbf{B}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix}, \dots, \begin{pmatrix} \mathbf{B}_{m(n-k)} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix} \right\}.$$

Clearly, matrices of $(\mathbf{P}^t)^{-1} \mathcal{C}_{mat}^0 (\mathbf{Q}^t)^{-1}$ lie in $\mathcal{E}\mathcal{G}_g(n, k, m, \ell_1, \ell_2)^\perp$. Moreover, any matrix of the shape $(\mathbf{P}^t)^{-1} \begin{pmatrix} \mathbf{B} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix} (\mathbf{Q}^t)^{-1}$ lying in $\mathcal{E}\mathcal{G}_g(n, k, m, \ell_1, \ell_2)^\perp$ should satisfy $\mathbf{B} \in \Psi_\gamma(\mathcal{G})^\perp$. Consequently, $\mathcal{V} \cap \mathcal{W} = 0$ and, for dimensional reasons, they should be complement subspaces.

More generally, one can prove that any matrix code with the above shape is the dual of an enhanced Gabidulin code. Namely, starting with a matrix Gabidulin code “extended by zero” to which we add a random complement subspace of the aforementioned space \mathcal{V} and which we left and right multiply by invertible matrices, we get the dual of an enhanced Gabidulin code.

Indistinguishability of Enhanced Gabidulin Matrix Codes. The security of the scheme we introduce later is based on the difficulty of distinguishing a random matrix code from a code as defined above.

We formally define in this section the problem of distinguishing an Enhanced Gabidulin matrix code from a random matrix code, on which the security of the EGMC-McEliece encryption scheme that we present below is based, and the associated search problem. We conjecture that both problems are difficult to solve.

Definition 16 (EGMC(k, m, n, ℓ_1, ℓ_2) (distribution)). *Let $k, m, n, \ell_1, \ell_2 \in \mathbb{N}$ be such that $k \leq n \leq m$. The Enhanced Gabidulin Matrix Code distribution EGMC(k, m, n, ℓ_1, ℓ_2) samples a vector $\mathbf{g} \xleftarrow{\$} \mathbb{F}_q^n$, a basis $\gamma \xleftarrow{\$} \mathcal{B}(\mathbb{F}_q^m)$, $\mathbf{P} \xleftarrow{\$}$*

$\mathbf{GL}_{m+\ell_1}(\mathbb{F}_q)$, $\mathbf{Q} \xleftarrow{\$} \mathbf{GL}_{n+\ell_2}(\mathbb{F}_q)$ and km random matrices: $\mathbf{R}_i \xleftarrow{\$} \mathbb{F}_q^{m \times \ell_2}$, $\mathbf{R}'_i \xleftarrow{\$} \mathbb{F}_q^{\ell_1 \times m}$ and $\mathbf{R}''_i \xleftarrow{\$} \mathbb{F}_q^{\ell_1 \times \ell_2}$; computes $\mathcal{B} = (\mathbf{A}_1, \dots, \mathbf{A}_{km})$ a basis of the matrix code $\Psi_\gamma(\mathcal{G})$, and outputs the matrix code with basis:

$$\mathcal{RB} = \left(P \begin{pmatrix} \mathbf{A}_1 & \mathbf{R}_1 \\ \mathbf{R}'_1 & \mathbf{R}''_1 \end{pmatrix} \mathbf{Q}, \dots, P \begin{pmatrix} \mathbf{A}_{km} & \mathbf{R}_{km} \\ \mathbf{R}'_{km} & \mathbf{R}''_{km} \end{pmatrix} \mathbf{Q} \right).$$

Definition 17 (EGMC-Indistinguishability problem). Given a matrix code \mathcal{C}_{mat} of size $(m + \ell_1) \times (n + \ell_2)$ and dimension mk , the Decisional EGMC-Indistinguishability $(k, m, n, \ell_1, \ell_2)$ problem asks to decide with non-negligible advantage whether \mathcal{C}_{mat} sampled from the $\text{EGMC}(k, m, n, \ell_1, \ell_2)$ distribution or the uniform distribution over the set of \mathbb{F}_q -subspaces of $\mathbb{F}_q^{(m+\ell_1) \times (n+\ell_2)}$ of dimension mk .

Definition 18 (EGMC-Search problem). Let $k, m, n, \ell \in \mathbb{N}$ be such that $k \leq n \leq m$, and \mathcal{C}_{mat} sampled from the $\text{EGMC}(k, m, n, \ell)$ distribution. The EGMC-Search problem asks to retrieve the basis $\gamma \in \mathcal{B}(\mathbb{F}_{q^m})$ and the evaluation vector $\mathbf{g} \in \mathbb{F}_{q^m}^n$ used to construct \mathcal{C}_{mat} .

Claim. Given a vector code $\mathcal{C}_{vec} \subseteq \mathbb{F}_{q^m}^n$ and an \mathbb{F}_q -basis γ of \mathbb{F}_{q^m} . Then, the matrix code $\mathcal{C}_{mat} = \Psi_\gamma(\mathcal{C}_{vec})$ is distinguishable from a random matrix code in polynomial time.

Indeed, even without knowing γ , the \mathbb{F}_{q^m} -linear structure can be detected by computing the left stabilizer algebra of \mathcal{C}_{mat} . That is to say

$$\text{Stab}_L(\mathcal{C}_{mat}) \stackrel{\text{def}}{=} \{ \mathbf{P} \in \mathbb{F}_q^{m \times m} \mid \forall \mathbf{C} \in \mathcal{C}_{mat}, \mathbf{PC} \in \mathcal{C}_{mat} \}.$$

For a random matrix code, with high probability this algebra has dimension 1 and only contains the matrices $\lambda \mathbf{I}$ where $\lambda \in \mathbb{F}_q$ and \mathbf{I} denotes the $m \times m$ identity matrix. For a code \mathcal{C}_{mat} which comes from an \mathbb{F}_{q^m} -linear code, the algebra $\text{Stab}_L(\mathcal{C}_{mat})$ contains a sub-algebra isomorphic to \mathbb{F}_{q^m} and hence has dimension at least m . Since the computation of the stabilizer algebra can be done by solving a linear system, this yields a polynomial time distinguisher.

Remark 1. Actually, using tools described in [14] it is even possible to do more than distinguishing and recover a description of \mathcal{C}_{vec} as a vector code.

Alternative Approach: Deleting Rows and Columns. We could have considered another masking procedure which consists in deleting rows and columns of the matrix code rather than adding random rows and columns. Deleting rows or columns is also an option which blurs the (right and left) stabilizer algebras. If some columns are removed, we obtain a punctured Gabidulin code, which remains a Gabidulin code, and hence can be decoded. If some rows are also removed, then the decryption will require to correct both errors and erasures,

which forces to reduce the amount of errors during the encryption process and then reduces the security with respect to generic decoding attacks.

4 Generic McEliece and Niederreiter Frames Based on MinRank

The McEliece frame is a generic process to construct code-based encryption schemes, proposed in [28]. It consists in masking a generator matrix of a vector code for which we know an efficient decoding algorithm with a secret operation. For an opponent who does not know the secret transformation, decrypting the ciphertext is as difficult as decoding a general linear code.

We propose here an adaptation of the McEliece frame to matrix codes: rather than hiding the generator matrix of a vector code, we propose to hide a basis of a secret matrix code, for which there exists an efficient decoding algorithm. The resulting scheme can be found in Fig. 1.

KeyGen (1^λ):

- Select a matrix code \mathcal{C}_{mat} of size $m \times n$ and dimension K on \mathbb{F}_q , with an efficient algorithm capable of decoding up to r errors.
- Let \mathcal{T} a transformation which turns a matrix code into another one with a trapdoor. Define the code $\mathcal{C}'_{mat} = \mathcal{T}(\mathcal{C}_{mat})$.
- Compute $\mathcal{B} = (\mathbf{M}_1, \dots, \mathbf{M}_K)$ a basis of \mathcal{C}'_{mat} .
- Return $\text{pk} = \mathcal{B}$ and $\text{sk} = (\mathcal{C}_{mat}, \mathcal{T}^{-1})$.

Encrypt(pk, μ):

Input: $\text{pk} = (\mathbf{M}_1, \dots, \mathbf{M}_K)$, $\mu \in \mathbb{F}_q^K$.

- Sample uniformly at random a matrix $\mathbf{E} \in \mathbb{F}_q^{m \times n}$ such that $\text{rank } \mathbf{E} \leq r$.
- Return $\mathbf{Y} = \sum_{i=1}^K \mu_i \mathbf{M}_i + \mathbf{E}$.

Decrypt(sk, \mathbf{Y}):

- Compute $\tilde{\mathbf{Y}} = \mathcal{T}^{-1}(\mathbf{Y})$.
- Apply the decoding algorithm of \mathcal{C}_{mat} on the matrix $\tilde{\mathbf{Y}}$ to retrieve the message μ .

Fig. 1. MinRank-McEliece encryption frame

Comments. Not just any transformation can be chosen: it must be compatible with the decoding of the noisy matrix $\tilde{\mathbf{Y}}$.

An opponent which does not know the original code \mathcal{C}_{mat} must solve a general instance of the MinRank problem to retrieve the message μ , that is at least as difficult as the decoding problem in rank metric. Therefore, it guarantees a security at least as high as that of the standard McEliece frame for the same sizes of ciphertext and public key.

The Niederreiter frame is a variant of the McEliece frame for code-based encryption, which consists in hiding a parity check matrix rather than a generator

matrix. It allows to obtain syndromes as ciphertexts, which are shorter than noisy codewords for an equivalent security, since the MinRank problem and the MinRank-Syndrome problem are equivalent. The resulting scheme can be found in Fig. 2.

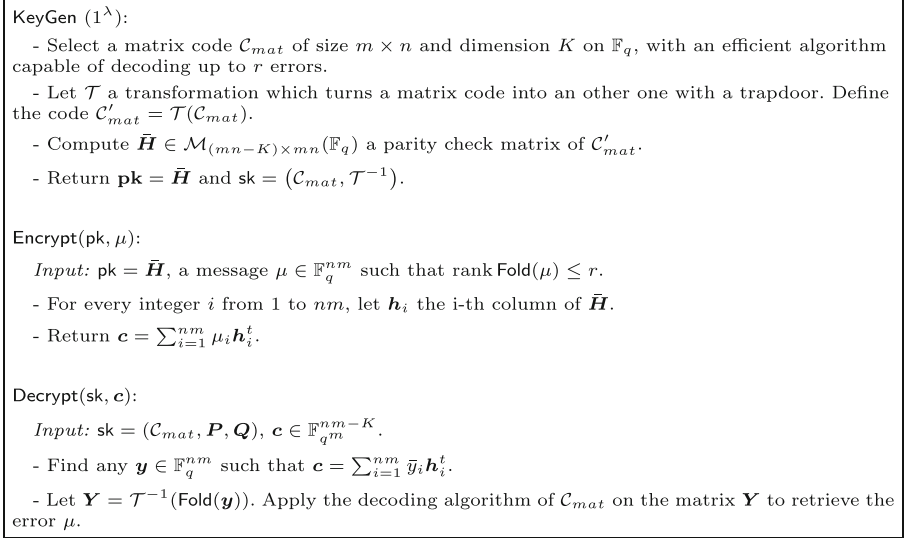


Fig. 2. MinRank-Niederreiter encryption frame

Comments. Since only the original matrix code is equipped with an efficient decoding algorithm, it is necessary to retrieve a noisy word whose syndrome is the ciphertext \mathbf{c} . Unlike the previous scheme, the message μ corresponds to the error of the noisy codeword.

5 New Encryption Schemes

5.1 EGMC-McEliece Encryption Scheme

We apply the encryption frames we defined above in Sect. 4 to matrix Gabidulin codes, by using the trapdoor presented in Definition 14. We obtain an encryption scheme whose public key is an Enhanced Gabidulin Matrix code. The secret key contains the original Gabidulin code \mathcal{G} , the basis γ on which the secret code $\tilde{\mathcal{G}}$ has been extended and the trapdoor. The resulting scheme is presented in Fig. 3.

Comments. The frame is valid since the multiplication by an invertible matrix makes the rank invariant. We can easily see that $\tilde{\mathbf{Y}}$ is a noisy codeword of \mathcal{C}_{mat} , whose error is equal to $\mathbf{P}^{-1} \mathbf{E} \mathbf{Q}^{-1}$. Since the multiplication by \mathbf{P}^{-1} and \mathbf{Q}^{-1} does not change the rank of the matrix, we still have: $\text{rank } \mathbf{P}^{-1} \mathbf{E} \mathbf{Q}^{-1} \leq r$.

KeyGen (1^λ):

- Select a random $[m, k]_q^m$ Gabidulin code \mathcal{G} , with an efficient algorithm capable of decoding up to $\lfloor \frac{m-k}{2} \rfloor$ errors.
- Sample uniformly at random a basis $\gamma \xleftarrow{\$} \mathcal{B}(\mathbb{F}_q^m)$.
- Compute a basis $(\mathbf{A}_1, \dots, \mathbf{A}_{km})$ of the code $\Psi_\gamma(\mathcal{G})$.
- For i in range 1 to km , sample uniformly at random: $\mathbf{R}_i \xleftarrow{\$} \mathbb{F}_q^{m \times \ell_2}$, $\mathbf{R}'_i \xleftarrow{\$} \mathbb{F}_q^{\ell_1 \times m}$ and $\mathbf{R}''_i \xleftarrow{\$} \mathbb{F}_q^{\ell_1 \times \ell_2}$.
- Define the matrix code \mathcal{C}_{mat} as explained in Definition 16.
- Sample uniformly at random matrices $\mathbf{P} \xleftarrow{\$} \mathbf{GL}_{m+\ell_1}(\mathbb{F}_q)$ and $\mathbf{Q} \xleftarrow{\$} \mathbf{GL}_{m+\ell_2}(\mathbb{F}_q)$.
- Define the code $\mathcal{C}'_{mat} = \mathbf{P}\mathcal{C}_{mat}\mathbf{Q}$.
- Let $\mathcal{B} = (\mathbf{M}_1, \dots, \mathbf{M}_{km})$ be a basis of \mathcal{C}'_{mat} .
- Return $\text{pk} = \mathcal{B}$ and $\text{sk} = (\mathcal{G}, \gamma, \mathbf{P}, \mathbf{Q})$.

Encrypt(pk, μ):

- Input:* $\text{pk} = (\mathbf{M}_1, \dots, \mathbf{M}_{km})$, $\mu \in \mathbb{F}_q^{km}$.
- Sample uniformly at random a matrix $\mathbf{E} \in \mathbb{F}_q^{(m+\ell_1) \times (m+\ell_2)}$ such that $\text{rank } \mathbf{E} \leq r$.
 - Return $\mathbf{Y} = \sum_{i=1}^{km} \mu_i \mathbf{M}_i + \mathbf{E}$.

Decrypt(sk, \mathbf{Y}):

- Compute $\mathbf{P}^{-1}\mathbf{Y}\mathbf{Q}^{-1}$. Truncate the ℓ_1 last rows to obtain $\mathbf{M} \in \mathbb{F}_q^{m \times (m+\ell_2)}$.
- Compute $\Psi_\gamma^{-1}(\mathbf{M}) \in \mathbb{F}_q^{m+\ell_2}$. Let \mathbf{y} be the first m entries.
- Apply the decoding algorithm of \mathcal{G} on the word \mathbf{y} , it returns an error vector $\mathbf{e} \in \mathbb{F}_q^m$.
- Compute $\mathbf{E}' = \Psi_\gamma(\mathbf{e})$.
- Solve the linear system:

$$\mathbf{Y} - (\mathbf{E}' | \mathbf{N})\mathbf{Q} = \sum_{i=1}^{km} \mu_i \mathbf{M}_i$$

whose $(k + \ell_2)m$ unknowns in \mathbb{F}_q are the μ_i and the remaining part of the error \mathbf{N} of size $m \times \ell_2$.

- Return $\hat{\mu}$ the solution of the above system.

Fig. 3. EGMC-McEliece encryption scheme

The Enhanced code \mathcal{C}_{mat} is not decodable on all its coordinates: the decoding algorithms allow to decode only the first n coordinates, but the random noise map prevents from decoding the others. However, these ℓ coordinates can be recovered by solving a linear system of equations. On the other hand, using a matrix code, less structured than a vector code, increases the complexity of the attacks.

An opponent who wants to decrypt a ciphertext without retrieving the secret key has to solve a MinRank instance. In practice, we choose the same value for ℓ_1 and ℓ_2 in order to obtain square matrices, the algebraic attacks against the MinRank problem being less efficient in this case.

Proposition 2 (Decryption correctness). *If the weight r of the error is at most the decoding capacity $\lfloor \frac{m-k}{2} \rfloor$ of the code \mathcal{G} , the decryption algorithm outputs the correct message μ .*

Proof. Let $\Phi : \mathbb{F}_q^{(m+\ell_1) \times (m+\ell_2)} \rightarrow \mathbb{F}_q^{m \times m}$ be the map truncating matrices by removing the ℓ_1 last rows and ℓ_2 last columns. Then,

$$\tilde{Y} = \Phi(\mathbf{P}^{-1} \mathbf{Y} \mathbf{Q}^{-1}) = \sum_i \mu_i \Phi(\mathbf{P}^{-1} \mathbf{M}_i \mathbf{Q}^{-1}) + \Phi(\mathbf{P}^{-1} \mathbf{E} \mathbf{Q}^{-1}). \quad (1)$$

Since $\Phi(\mathbf{P}^{-1} \mathbf{M}_1 \mathbf{Q}^{-1}), \dots, \Phi(\mathbf{P}^{-1} \mathbf{M}_{km} \mathbf{Q}^{-1})$ form an \mathbb{F}_q -basis of \mathcal{C}_{mat} and $\text{rank}(\Phi(\mathbf{P}^{-1} \mathbf{E} \mathbf{Q}^{-1})) \leq r$, applying decoding algorithm to (1) yields μ . \square

Proposition 3. *Under the assumption that there exists no PPT algorithm to solve the MinRank problem with non negligible probability and no PPT distinguisher for the EGMC-Indistinguishability problem with non negligible advantage, then the scheme presented Fig. 3 is OW-CPA.*

Proof. Suppose there exists an efficient PPT decoder \mathcal{A}_{EGMC} of the EGMC-McEliece encryption scheme which takes as input the ciphertext and the public key, with a non negligible probability $\varepsilon(\lambda)$. Using this algorithm, we construct the following distinguisher \mathcal{D} for the EGMC-Indistinguishability problem:

```

Input:  $\mathcal{B} = (\mathbf{M}_1, \dots, \mathbf{M}_{km})$ 
 $\mu \xleftarrow{\$} \mathbb{F}_q^{km}$ 
 $\mathbf{E} \xleftarrow{\$} \{\mathbf{X} \in \mathbb{F}_q^{(m+\ell_1) \times (m+\ell_2)} \mid \text{rank } \mathbf{X} \leq r\}$ 
 $\tilde{\mu} \leftarrow \mathcal{A}_{EGMC}(\mathcal{B}, \sum \mu_i \mathbf{M}_i + \mathbf{E})$ 
if  $\mu = \tilde{\mu}$  then
    return 1
else
    return 0
end if
    
```

Let \mathcal{A}_{MR} be an adversary against the MinRank problem. In the case where \mathcal{B} is not a basis of an Enhanced Gabidulin matrix code, retrieving \mathbf{E} is equivalent to solving a random MinRank instance. Then:

$$\begin{aligned} \text{Adv}(\mathcal{D}) &= |\Pr(\mathcal{D}(\mathcal{B}) = 1 | \mathcal{B} \text{ uniformly random}) - \Pr(\mathcal{D}(\mathcal{B}) = 1 | \mathcal{B} \text{ basis of EGMC})| \\ &= |\text{Succ}(\mathcal{A}_{MR}) - \text{Succ}(\mathcal{A}_{EGMC})|, \end{aligned}$$

since decrypting a ciphertext is strictly equivalent to retrieving the error \mathbf{E} in the case where \mathcal{B} is the basis of an EGMC. We deduce that:

$$\text{Succ}(\mathcal{A}_{EGMC}) \leq \text{Succ}(\mathcal{A}_{MR}) + \text{Adv}(\mathcal{D}).$$

Since $\text{Succ}(\mathcal{A}_{EGMC}) \geq \varepsilon(\lambda)$, then $\text{Succ}(\mathcal{A}_{MR}) \geq \frac{\varepsilon(\lambda)}{2}$ or $\text{Adv}(\mathcal{D}) \geq \frac{\varepsilon(\lambda)}{2}$, that are non negligible probabilities. We have proven that there exists either an efficient PPT algorithm to solve the MinRank problem, or an efficient PPT distinguisher to solve the EGMC-Indistinguishability problem, which allows to conclude. \square

5.2 EGMC-Niederreiter Encryption Scheme

It is also possible to adapt the above ideas to the Niederreiter frame, the dual version of the McEliece frame. It is then necessary to adapt the Syndrome Decoding approach to matrix codes, in the same way as what is presented in Subsect. 2.3. The resulting scheme can be found in Fig. 4.

KeyGen (1^λ):

- Select a random $[m, k]_q^m$ Gabidulin code \mathcal{G} , with an efficient algorithm capable of decoding up to $\lfloor \frac{m-k}{2} \rfloor$ errors.
- Sample uniformly at random a basis $\gamma \xleftarrow{\mathbb{S}} \mathcal{B}(\mathbb{F}_q^m)$.
- Compute a basis $(\mathbf{A}_1, \dots, \mathbf{A}_{km})$ of the code $\Psi_\gamma(\mathcal{G})$.
- For i in range 1 to km , sample uniformly at random: $\mathbf{R}_i \xleftarrow{\mathbb{S}} \mathbb{F}_q^{m \times \ell_2}$, $\mathbf{R}'_i \xleftarrow{\mathbb{S}} \mathbb{F}_q^{\ell_1 \times m}$ and $\mathbf{R}''_i \xleftarrow{\mathbb{S}} \mathbb{F}_q^{\ell_1 \times \ell_2}$.
- Define the matrix code \mathcal{C}_{mat} as explained above.
- Sample uniformly at random matrices $\mathbf{P} \xleftarrow{\mathbb{S}} \mathbf{GL}_{m+\ell_1}(\mathbb{F}_q)$ and $\mathbf{Q} \xleftarrow{\mathbb{S}} \mathbf{GL}_{m+\ell_2}(\mathbb{F}_q)$.
- Define the code $\mathcal{C}'_{mat} = \mathbf{P}\mathcal{C}_{mat}\mathbf{Q}$.
- Compute $\tilde{\mathbf{H}} \in \mathbb{F}_q^{((m+\ell_1)(m+\ell_2)-mk) \times (m+\ell_1)(m+\ell_2)}$ a parity check matrix of \mathcal{C}'_{mat} .
- Return $\mathbf{pk} = \tilde{\mathbf{H}}; \mathbf{sk} = (\gamma, \mathcal{G}, \mathbf{Q})$

Encrypt(\mathbf{pk}, μ):

- Input:* $\mathbf{pk} = \tilde{\mathbf{H}}$, a message $\mu \in \mathbb{F}_q^{(m+\ell_1)(m+\ell_2)}$ such that $\text{rank Fold}(\mu) \leq r$.
- For every integer i from 1 to $(m+\ell_1)(m+\ell_2)$, let \mathbf{h}_i be the i -th column of $\tilde{\mathbf{H}}$.
 - Return $\mathbf{c} = \sum_{i=1}^{(m+\ell_1)(m+\ell_2)} \mu_i \mathbf{h}_i^t$.

Decrypt(\mathbf{sk}, \mathbf{c}):

- Input:* $\mathbf{sk} = (\gamma, \mathcal{G}, \mathbf{Q})$, $\mathbf{c} \in \mathbb{F}_q^{(m+\ell_1)(m+\ell_2)-mk}$.
- Find any $\tilde{\mathbf{y}} \in \mathbb{F}_q^{(m+\ell_1)(m+\ell_2)}$ such that $\mathbf{c} = \sum_{i=1}^{(m+\ell_1)(m+\ell_2)} \tilde{y}_i \mathbf{h}_i^t$.
 - Let $\mathbf{Y} = \text{Fold}(\tilde{\mathbf{y}})$. Compute $\mathbf{P}^{-1}\mathbf{Y}\mathbf{Q}^{-1}$. Truncate the ℓ_1 last rows to obtain $\mathbf{M} \in \mathcal{M}_{m \times (m+\ell)}(\mathbb{F}_q)$.
 - Let \mathbf{y} be the first m coordinates of $\Psi_\gamma^{-1}(\mathbf{M}) \in \mathbb{F}_q^{m+\ell_2}$.
 - Let $\mathbf{y} = \mathbf{m}\mathbf{G} + \mathbf{e}$, with $\text{rank}(\mathbf{e}) \leq r$. Apply the decoding algorithm of \mathcal{G} on the word composed of the m first coordinates.
 - Let $\mathbf{E} = \Psi_\gamma(\mathbf{e})$. Then $\mathbf{E} = (\mathbf{E}'|\mathbf{N}) \in \mathcal{M}_{m \times (m+\ell_2)}(\mathbb{F}_q)$ is a matrix whose the nm coefficients of \mathbf{E}' are known, and \mathbf{N} is the remaining part of the error.
 - Solve the linear system $\mathbf{c} = \sum_{i=1}^{(m+\ell_1)(m+\ell_2)} \tilde{e}_i \mathbf{h}_i^t$ to find the ℓm values of \mathbf{N} , with $\tilde{\mathbf{e}} = \text{Unfold}(\mathbf{E})$ with unknown some coefficients.
 - Return $\hat{\mu} = \text{Unfold}(\mathbf{E})$.

Fig. 4. EGMC-Niederreiter encryption scheme

Proposition 4 (Decryption correctness). *If the weight r of the error is under the decoding capacity $\lfloor \frac{m-k}{2} \rfloor$ of the code \mathcal{G} , the decryption algorithm outputs the correct message μ .*

Proof. Let $\bar{\mathbf{G}} \in \mathcal{M}_{mk \times (m+\ell_1)(m+\ell_2)}(\mathbb{F}_q)$ be a generator matrix of the associated unfolded \mathcal{C}'_{mat} vector code, and $\bar{\mathbf{H}} \in \mathcal{M}_{((m+\ell_1)(m+\ell_2)-k) \times (m+\ell_1)(m+\ell_2)}(\mathbb{F}_q)$ a parity check matrix. Let $\bar{\mathbf{y}} \in \mathbb{F}_q^{(m+\ell_1)(m+\ell_2)}$ be such that $\mathbf{c} = \sum_{i=1}^{(m+\ell_1)(m+\ell_2)} \bar{y}_i \mathbf{M}_i$. Then:

$$\text{Unfold}(\mathbf{c}) = \sum_{i=1}^{(m+\ell_1)(m+\ell_2)} \bar{y}_i \text{Unfold}(\mathbf{M}_i) = \bar{\mathbf{y}} \bar{\mathbf{H}}^t = \bar{\mathbf{e}} \bar{\mathbf{H}}^t$$

where $\bar{\mathbf{e}} = \text{Unfold}(\Psi_\gamma(\mathbf{e}))$.

The nm first values of $\bar{\mathbf{e}}$ can be retrieved by decoding a word of the noisy Gabidulin code $\Psi_\gamma^{-1}(\mathcal{C}'_{mat})$. The rank r of the error must not exceed the decoding capacity.

The $\ell_2 m$ last values can be retrieved by solving a linear system of $(m - k + \ell_2)m$ equations, which is possible since $k \leq m$. Multiplying by \mathbf{Q}^{-1} allows to retrieve the original message μ . \square

Proposition 5. *Under the assumption that there exists no PPT algorithm to solve the MinRank-Syndrome problem with non negligible probability and no PPT distinguisher for the EGMC-Indistinguishability problem with non negligible advantage, then the scheme presented Fig. 4 is OW-CPA.*

Proof. This proof is similar to that of Proposition 3. Let \mathcal{A}_{EGMC-N} be an efficient PPT adversary for the EGMC-Niederreiter encryption scheme. By considering the following distinguisher for the EGMC-Indistinguishability problem:

Input: $\bar{\mathbf{H}} \in \mathbb{F}_q^{((m+\ell_1)(m+\ell_2)-mk) \times (m+\ell_1)(m+\ell_2)}$
 $\mu \xleftarrow{\$} \{\mathbf{m} \in \mathbb{F}_q^{(m+\ell_1)(m+\ell_2)} \mid \text{rank Fold}(\mathbf{m}) \leq r\}$
 $\tilde{\mu} \leftarrow \mathcal{A}_{EGMC-N}(\bar{\mathbf{H}}, \sum \mu_i \mathbf{h}_i^t)$
if $\mu = \tilde{\mu}$ **then**
 return 1
else
 return 0
end if

one demonstrates by identical reasoning that there exists either an efficient PPT algorithm to solve the MinRank-Syndrome problem, or an efficient PPT distinguisher to solve the EGMC-Indistinguishability problem. \square

5.3 Algorithms Complexity

The most costly step for the key generation is the multiplication by the matrices \mathbf{P} and \mathbf{Q} . It involves computing km products of the form $\mathbf{P}\mathbf{A}_i\mathbf{Q}$, which requires $km((m+\ell_1)^2(m+\ell_2) + (m+\ell_2)^2(m+\ell_1))$ multiplications in \mathbb{F}_q . For the Niederreiter variant, we must also consider the cost of computing the matrix $\bar{\mathbf{H}}$, which can be done using Gaussian elimination, giving a complexity of $\mathcal{O}(((m+\ell_1)(m+\ell_2))^3)$ operations in \mathbb{F}_q .

The encryption step is the cheapest as we only need to compute the ciphertext \mathbf{c} by adding $(m + \ell_1)(m + \ell_2)$ elements of $\mathbb{F}_q^{(m+\ell_1)(m+\ell_2)-km}$, each requiring $(m + \ell_1)(m + \ell_2) - km$ multiplications in \mathbb{F}_q for the multiplication by μ_i in the Niederreiter variant. For the McEliece variant, computing the $\mu_i \mathbf{M}_i$ requires $km(m + \ell_1)(m + \ell_2)$ multiplications in \mathbb{F}_q .

The decryption process requires solving one (or two in the case of the Niederreiter variant) linear system with $(m + \ell_1)(m + \ell_2)$ unknowns in \mathbb{F}_q . This gives a complexity of $\mathcal{O}(((m + \ell_1)(m + \ell_2))^3)$ operations in \mathbb{F}_q .

6 Security Analysis

This section deals with the security of the schemes proposed in Sect. 5. There exist two types of attacks. The first one are the structural attacks which consist in retrieving the structure of the secret code from the information given in the public key. It amounts to recovering the secret key by solving the EGMC-Search problem (see Definition 18), and efficiently decrypting any ciphertext thanks to the underlying decoding algorithm of the secret code. The second one includes attacks which try to recover the message from the ciphertext and the public key, that is equivalent to solving an instance of the MinRank problem.

6.1 Attacks on the Key

We recall that an opponent who wants to retrieve the secret key has to solve the EGMC-Search problem:

Instance: A matrix code \mathcal{C}_{mat} sampled from the $\text{EGMC}(k, m, n, \ell_1, \ell_2)$ distribution.

Problem: Retrieve the basis $\gamma \in \mathcal{B}(\mathbb{F}_{q^m})$ and the evaluation vector $\mathbf{g} \in \mathbb{F}_{q^m}^n$ of the Gabidulin code \mathcal{G} used to construct \mathcal{C}_{mat} .

In this section, we present two algorithms for solving the EGMC decision problem. Just like other distinguishers for structured codes in rank metric (e.g. LRPC codes), there are two types of attacks. The first one is of combinatorial nature and tries to detect the \mathbb{F}_{q^m} -linear structure, whereas the second one is an algebraic distinguisher that is inspired from the Overbeck attack.

Before that, we present polynomial distinguishers for vector and matrix plain Gabidulin codes, that are not directly usable for EGMC codes considered in our schemes, since they are enhanced with perturbing random rows and columns that increase the cost of a distinguisher.

Distinguisher for Vector Gabidulin Codes. Due to their strong algebraic structure, Gabidulin codes can be easily distinguished from random linear codes. As it has been proven, the security of the schemes relies on the difficulty to distinguish an Enhanced Gabidulin vector code (see Proposition 3).

Let $\mathbf{x} = (x_1, \dots, x_n) \in \mathbb{F}_{q^m}^n$. For any $i \in \{0, \dots, m-1\}$, we define:

$$\mathbf{x}^{[i]} = (x_1^{q^i}, \dots, x_n^{q^i}).$$

This definition naturally extends to codes. For a vector space $E \subset \mathbb{F}_{q^m}^n$, we write $E^{[1]}$ the image of E by the Frobenius application. We can generalize this notation to $E^{[i]}$ for i compositions of the Frobenius.

Definition 19. Let \mathcal{C} be an $[n, k]_{q^m}$ linear code. We define the f -th Frobenius sum of \mathcal{C} as:

$$\Lambda_f(\mathcal{C}) = \mathcal{C} + \mathcal{C}^{[1]} + \dots + \mathcal{C}^{[f]}.$$

If \mathbf{G} is a generator matrix of \mathcal{C} , then a generator matrix for $\Lambda_f(\mathcal{C})$ is:

$$\begin{pmatrix} \mathbf{G} \\ \mathbf{G}^{[1]} \\ \vdots \\ \mathbf{G}^{[f]} \end{pmatrix} \in \mathbb{F}_{q^m}^{(f+1)k \times n}.$$

For convenience, we abusively denote this matrix as $\Lambda_f(\mathbf{G})$.

Proposition 6 ([21]). Let \mathcal{G} be an $[n, k]_{q^m}$ Gabidulin code. For any $f \geq 0$:

$$\dim \Lambda_f(\mathcal{G}) = \min\{n, k + f\}.$$

Let \mathcal{C} be a random $[n, k]_{q^m}$ linear code. For any $f \geq 0$, we have with high probability:

$$\dim \Lambda_f(\mathcal{C}) = \min\{n, k(f+1)\}.$$

Remark 1. Let $f \in \{0, \dots, n-k\}$. If \mathcal{G} is a Gabidulin code $[n, k]_{q^m}$ with evaluation vector \mathbf{g} , then $\Lambda_f(\mathcal{G})$ is a Gabidulin code $[n, n-1]_{q^m}$ with the same evaluation vector \mathbf{g} .

Remark 2. Another way to distinguish a Gabidulin code from a random one is to observe the intersection: $\mathcal{C} \cap \mathcal{C}^{[1]} = \{0\}$ with high probability for a random code, whereas $\dim(\mathcal{G} \cap \mathcal{G}^{[1]}) = k-1$ for any Gabidulin code.

Since the rank of a matrix can be computed in polynomial time, this noteworthy behavior allows to easily distinguish a Gabidulin code from a random code. It has been exploited by Overbeck. See [32, 33] for some structural attacks against cryptographic schemes based on Gabidulin codes.

Distinguisher for Matrix Gabidulin Codes. As already mentioned in the preliminary part, starting from an \mathbb{F}_{q^m} -linear code $\mathcal{C}_{vec} \subseteq \mathbb{F}_{q^m}^n$, the map Ψ_γ sending it on a matrix code is not enough to hide it from a random matrix code since it has a non trivial left stabilizer algebra.

When $n = m$, the following statement suggests the existence of another detectable structure.

Lemma 1. *Suppose that $n = m$ and let $\mathbf{g} \in \mathbb{F}_{q^m}^m$ whose entries are \mathbb{F}_q -linearly independent. let γ be an \mathbb{F}_q -basis of \mathbb{F}_{q^m} . Then, $\Psi_\gamma(\mathcal{G}_g(m, k, m))$ has a right stabilizer algebra of dimension $\geq m$.*

Proof. A q -polynomial P of q -degree $< k$ induces an \mathbb{F}_q -endomorphism of \mathbb{F}_{q^m} and $\Psi_\gamma((P(g_1), \dots, P(g_m)))$ is the matrix representation of this endomorphism from the basis $\mathbf{g} = (g_1, \dots, g_m)$ to the basis γ . Then, observe that the space of q -polynomials of degree $< k$ is stable by right composition by any q -polynomial αX for $\alpha \in \mathbb{F}_{q^m}$. Indeed, for any $P = p_0X + p_1X^q + \dots + p_{k-1}X^{q^{k-1}}$ with q -degree $< k$, we have

$$P \circ \alpha X = p_0\alpha X + p_1\alpha^q X^q + \dots + p_{k-1}\alpha^{q^{k-1}} X^{q^{k-1}}$$

and the resulting q -polynomial has the same degree.

The stability of this space of q -polynomials by right composition by αX entails that $\Psi_\gamma(\mathcal{G}_g(m, k, m))$ is stabilized on the right by the matrix representing the multiplication by α (regarded as an \mathbb{F}_q -endomorphism of \mathbb{F}_{q^m}) in the basis \mathbf{g} .

In summary, the right stabilizer algebra contains a sub-algebra isomorphic to \mathbb{F}_{q^m} and hence has dimension larger than or equal to m . \square

The goal of the masking we propose is to mask both left and right \mathbb{F}_{q^m} -linear structure of a Gabidulin code, leading to trivial left and right stabilizer algebras.

A Combinatorial Distinguisher Detecting the \mathbb{F}_{q^m} -Linear Structure. Suppose from now on that we are given a code \mathcal{C}_{mat} which is an enhanced Gabidulin matrix code. We keep the notation of Definitions 15 and 16. That is to say our code that we denote \mathcal{C}_{pub} is described by a basis:

$$\mathcal{RB}_{pub} = \left(P \begin{pmatrix} \mathbf{A}_1 & \mathbf{R}_1 \\ \mathbf{R}'_1 & \mathbf{R}''_1 \end{pmatrix} \mathbf{Q}, \dots, P \begin{pmatrix} \mathbf{A}_{km} & \mathbf{R}_{km} \\ \mathbf{R}'_{km} & \mathbf{R}''_{km} \end{pmatrix} \mathbf{Q} \right), \tag{2}$$

where the \mathbf{A}_{ij} 's are an \mathbb{F}_q -basis of $\Psi_\gamma(\mathcal{G}_g(m, k, n))$, \mathbf{P}, \mathbf{Q} are random nonsingular matrices and the $\mathbf{R}_i, \mathbf{R}'_i, \mathbf{R}''_i$'s are random matrices with respective sizes $m \times \ell_2$, $\ell_1 \times n$ and $\ell_1 \times \ell_2$. Roughly speaking, this is a Gabidulin matrix code, enhanced with ℓ_1 random rows and ℓ_2 random columns. The matrix \mathbf{P} (resp. \mathbf{Q}) ‘‘mixes’’ rows (resp. columns) from the Gabidulin code with random ones.

We also introduce the secret ‘‘non-scrambled’’ version of the code denoted \mathcal{C}_0 and spanned by the basis:

$$\mathcal{RB}_0 = \left(\begin{pmatrix} \mathbf{A}_1 & \mathbf{R}_1 \\ \mathbf{R}'_1 & \mathbf{R}''_1 \end{pmatrix}, \dots, \begin{pmatrix} \mathbf{A}_{km} & \mathbf{R}_{km} \\ \mathbf{R}'_{km} & \mathbf{R}''_{km} \end{pmatrix} \right). \tag{3}$$

The idea of the combinatorial distinguisher to follow consists in applying a projection map on both the row and columns spaces of \mathcal{C}_{pub} in order to get rid of the contributions of the matrices $\mathbf{R}_i, \mathbf{R}'_i$ and \mathbf{R}''_i while not destroying the underlying \mathbb{F}_{q^m} -linear structure.

To understand the idea, let us first reason on the non scrambled code \mathcal{C}_0 . Choose two matrices

– $\mathbf{U} \in \mathbb{F}_q^{m \times (m+\ell_1)}$ of full rank whose ℓ_1 last columns are 0, *i.e.*,

$$\mathbf{U} = (\mathbf{U}_0 \mid \mathbf{0}), \quad \text{with } \mathbf{U}_0 \in \mathbf{GL}_m(\mathbb{F}_q).$$

– and, for some integer n' such that $k < n' \leq n$, a full rank matrix $\mathbf{V} \in \mathbb{F}_q^{(n+\ell_2) \times n'}$ whose last ℓ_2 rows are 0:

$$\mathbf{V} = \begin{pmatrix} \mathbf{V}_0 \\ \mathbf{0} \end{pmatrix}, \quad \text{with } \mathbf{V}_0 \in \mathbb{F}_q^{n \times n'} \text{ of full rank.}$$

Now, observe that the code $\mathbf{UC}_0\mathbf{V}$ is spanned by

$$\mathbf{U}_0\mathbf{A}_1\mathbf{V}_0, \dots, \mathbf{U}_0\mathbf{A}_{km}\mathbf{V}_0,$$

which is a basis of the matrix Gabidulin code $\Psi_{\gamma'}(\mathcal{G}_{\mathbf{gV}_0}(m, k, n'))$, where γ' is the image of the basis γ by \mathbf{U}_0 . Hence, this code is distinguishable from random by computing its left stabilizer algebra.

Note finally that the number of choices of \mathbf{U}, \mathbf{V} is $\approx q^{m^2+nn'}$. The latter quantity being minimal when $n' = k + 1$ (we should have $n' > k$ since otherwise the resulting code would be the full matrix space $\mathbb{F}_q^{m \times n'}$).

Now, when considering the public code \mathcal{C}_{pub} instead of \mathcal{C}_0 the very same observation can be made by replacing \mathbf{U} by $\mathbf{U}' \stackrel{\text{def}}{=} \mathbf{UP}^{-1}$ and \mathbf{V} by $\mathbf{V}' \stackrel{\text{def}}{=} \mathbf{Q}^{-1}\mathbf{V}$ and the number of good choices killing the contributions of the $\mathbf{R}_i, \mathbf{R}'_i, \mathbf{R}''_i$ matrices remains the same, namely: $q^{m^2+n(k+1)}$.

Thus, the suggested distinguisher consists in repeating the following operations:

- Guess the pair \mathbf{U}', \mathbf{V}' with $\mathbf{U}' \in \mathbb{F}_q^{m \times (m+\ell_1)}$ and $\mathbf{V}' \in \mathbb{F}_q^{(n+\ell_2) \times (k+1)}$,
- Compute the left stabilizer algebra of $\mathbf{U}'\mathcal{C}_{pub}\mathbf{V}'$.

until you get a stabilizer algebra of dimension $\geq m$.

The probability of finding a valid pair \mathbf{U}', \mathbf{V}' is

$$\mathbb{P} \approx \frac{q^{m^2+n(k+1)}}{q^{m(m+\ell_1)+(n+\ell_2)(k+1)}} = q^{-(m\ell_1+(k+1)\ell_2)}$$

which yields a complexity of

$$\tilde{O}(q^{m\ell_1+(k+1)\ell_2}) \quad (4)$$

for the distinguisher.

To conclude on this section let us do some remarks.

- The process is not symmetric on rows and columns since, on one hand \mathbf{U}' must kill the random rows while preserving the \mathbb{F}_{q^m} -linearity. On the other hand, the matrix \mathbf{V}' should only kill the random columns, even if it partially punctures the Gabidulin code.
- The above distinguisher holds when replacing the Gabidulin code by any \mathbb{F}_{q^m} -linear code, it does not use the Gabidulin structure. In the subsequent section, the proposed algebraic attacks will try to take advantage of the Gabidulin structure.

An Overbeck-Like Distinguisher. As already mentioned, the previous distinguisher does not actually take advantage of the Gabidulin structure and could hold for any \mathbb{F}_{q^m} -linear code. Let us discuss how to take advantage of the Gabidulin structure. The key of Overbeck's distinguisher is that, when given a Gabidulin code represented as a vector code \mathcal{C}_{vec} , one can easily compute its image by the Frobenius map $\mathcal{C}_{vec}^{[1]}$ and the sum $\mathcal{C}_{vec} + \mathcal{C}_{vec}^{[1]}$ is small (it has \mathbb{F}_{q^m} -dimension equal to $1 + \dim_{\mathbb{F}_{q^m}} \mathcal{C}_{vec}$) compared to what happens with a random code. More generally, the sum $\mathcal{C}_{vec} + \mathcal{C}_{vec}^{[1]} + \dots + \mathcal{C}_{vec}^{[t]}$ is small compared to the random case.

The difficulty in our setting is that we can access neither the multiplication operation by an element of \mathbb{F}_{q^m} nor the action of the Frobenius map. We suggest here to identify similar behaviours by solving a system of bilinear equations. We do not claim that it is the only manner to distinguish via the resolution of a bilinear system. However, our observation is that all our attempts led to the resolution of a quadratic system with $\Theta(m^2)$ unknowns for $\Theta(m^2)$ equations. A linearization would hence lead to $\Theta(m^4)$ unknowns for only $\Theta(m^2)$ equations. A further analysis of this system or of any other (and possibly smarter) algebraic modeling would be important to better assess the security of the system.

In the sequel, we assume that the dimension of the public code \mathcal{C}_{pub} satisfies

$$2 \dim \mathcal{C}_{pub} = 2mk \geq \dim \mathbb{F}_q^{(m+\ell_1) \times (n+\ell_2)}.$$

If this condition is not satisfied, an algebraic distinguisher of similar flavor can be searched on the dual code (see further for a discussion on a structural attack on the dual).

The idea of this distinguisher relies on this observation.

Lemma 2. *Let b be a non-negative integer. Let $\mathbf{M} \in \Psi_\gamma(\mathcal{G}_\gamma(m, b, m))$ and $\mathbf{C} \in \Psi_\gamma(\mathcal{G}_g(m, k, n))$. Then,*

$$\mathbf{MC} \in \Psi_\gamma(\mathcal{G}_g(m, k + b - 1, n)).$$

Proof. The matrix \mathbf{M} represents a q -polynomial P_M of q -degree $< b$ in the basis γ (regarding P_M as an \mathbb{F}_q -endomorphism of \mathbb{F}_{q^m}). The matrix \mathbf{C} represents a q -polynomial P_C of q -degree $< k$ from the basis g to the basis γ . Thus, \mathbf{MC}

represents the q -polynomial $P_M \circ P_C$ from the basis \mathbf{g} to the basis γ . The q -polynomial $P_M \circ P_C$ has q -degree $< k + b - 1$. This yields the result. \square

Remark 2. The previous lemma is somehow a matrix version of Overbeck's distinguisher.

As for the previous distinguisher, for the sake of clarity, we first reason on the non scrambled code \mathcal{C}_0 whose basis is given in (3).

Proposition 7. *Let $\mathcal{D} \subseteq \mathbb{F}_q^{(m+\ell_1) \times (m+\ell_1)}$ be the matrix code:*

$$\mathcal{D} \stackrel{\text{def}}{=} \left\{ \begin{pmatrix} \mathbf{B} & \mathbf{0} \\ \mathbf{T}_1 & \mathbf{T}_2 \end{pmatrix} \mid \mathbf{B} \in \Psi_\gamma(\mathcal{G}_\gamma(m, n-k, m)), \mathbf{T}_1 \in \mathbb{F}_q^{\ell_1 \times m}, \mathbf{T}_2 \in \mathbb{F}_q^{\ell_1 \times \ell_1} \right\}.$$

Then,

- (i) $\dim_{\mathbb{F}_q} \mathcal{D} = m(n-k) + \ell_1(m + \ell_1)$;
- (ii) The code $\mathcal{U} \stackrel{\text{def}}{=} \text{Span}_{\mathbb{F}_q} \{ \mathbf{DC} \mid \mathbf{D} \in \mathcal{D}, \mathbf{C} \in \mathcal{C}_0 \} \subseteq \mathbb{F}_q^{(m+\ell_1) \times (n+\ell_2)}$ satisfies:

$$\dim_{\mathbb{F}_q} \mathcal{U} \leq (m + \ell_1)(n + \ell_2) - m.$$

Proof. (i) is an immediate consequence of the definition of \mathcal{D} . To prove (ii), let $\mathbf{D} \in \mathcal{D}$ and $\mathbf{C} \in \mathcal{C}_0$. Then, they have the following shapes

$$\mathbf{D} = \begin{pmatrix} \mathbf{B} & \mathbf{0} \\ \mathbf{T}_1 & \mathbf{T}_2 \end{pmatrix} \quad \text{and} \quad \mathbf{C} = \begin{pmatrix} \mathbf{A} & \mathbf{R} \\ \mathbf{R}' & \mathbf{R}'' \end{pmatrix},$$

where $\mathbf{B} \in \Psi_\gamma(\mathcal{G}_\gamma(m, n-k, m))$ and $\mathbf{A} \in \Psi_\gamma(\mathcal{G}_g(m, k, n))$ and the other matrices are arbitrary. According to Lemma 2, we deduce that $\mathbf{BA} \in \Psi_\gamma(\mathcal{G}_g(m, n-1, n))$. Hence, any element of \mathcal{U} has the shape

$$\begin{pmatrix} \mathbf{C} & \mathbf{S} \\ \mathbf{S}' & \mathbf{S}'' \end{pmatrix},$$

where $\mathbf{C} \in \Psi_\gamma(\mathcal{G}_g(m, n-1, n))$ and the matrices \mathbf{S} , \mathbf{S}' and \mathbf{S}'' are arbitrary. This yields the upper bound on the dimension of \mathcal{U} . \square

Corollary 1. *There exists a matrix code $\mathcal{D}' \subseteq \mathbb{F}_q^{(m+\ell_1) \times (m+\ell_1)}$ such that*

$$\dim_{\mathbb{F}_q} \text{Span}_{\mathbb{F}_q} \{ \mathbf{DC} \mid \mathbf{D} \in \mathcal{D}', \mathbf{C} \in \mathcal{C}_{pub} \} \leq (m + \ell_1)(n + \ell_2) - m.$$

Proof. Recall that $\mathcal{C}_{pub} = \mathbf{PC}_0\mathbf{Q}$. Then, setting $\mathcal{D}' \stackrel{\text{def}}{=} \mathbf{PDP}^{-1}$ where \mathcal{D} is the code of Proposition 7, yields the result. \square

Corollary 1 is the key of our forthcoming distinguisher : it shows that some operation (the “left multiplication”) by \mathcal{D}' does not fill in the ambient space while it would fill it in if \mathcal{C}_{pub} was random.

However, the difficulties are that:

- (1) the code \mathcal{D}' is unknown;
- (2) we need to equate the fact that the dimension of the span of $\mathcal{D}'\mathcal{C}_{pub}$ is not $(m + \ell_1)(n + \ell_2)$.

To address (2), we proceed as follows. We will define a formal variable $\mathbf{D} \in \mathbb{F}_q^{(m+\ell_1) \times (m+\ell_1)}$. Note first that the matrix code \mathcal{D} of Proposition 7 contains the identity matrix. Then, Corollary 1 asserts that $\mathcal{C}_{pub} + \mathbf{D}\mathcal{C}_{pub}$ is not equal to the ambient space since it is contained in the code $\mathcal{D}'\mathcal{C}_{pub}$ of codimension at least m . Hence its dual $(\mathcal{C}_{pub} + \mathbf{D}\mathcal{C}_{pub})^\perp$ is nonzero. Since $(\mathcal{C}_{pub} + \mathbf{D}\mathcal{C}_{pub})^\perp \subseteq \mathcal{C}_{pub}^\perp$ there exists $\mathbf{M} \in \mathcal{C}_{pub}^\perp$ satisfying:

$$\forall \mathbf{C} \in \mathcal{C}_{pub}, \text{Tr}(\mathbf{D}\mathbf{C}\mathbf{M}^t) = 0.$$

Any $\mathbf{M} \in \mathcal{C}_{pub}^\perp$ solution of the above system is an element of $(\mathcal{C}_{pub} + \mathbf{D}\mathcal{C}_{pub})^\perp$. Thus, we can set the following bilinear system with

- **Unknowns:** $\mathbf{D} \in \mathbb{F}_q^{(m+\ell_1) \times (m+\ell_1)}$ and $\mathbf{M} \in \mathcal{C}_{pub}^\perp$;
- **Equations:** for any element \mathbf{C} of our \mathbb{F}_q -basis \mathcal{RB} (see (2)),

$$\text{Tr}(\mathbf{D}\mathbf{C}\mathbf{M}^t) = 0.$$

For this bilinear system we can count the equations and unknowns.

- **Number of unknowns:**
 - \mathbf{D} has $(m + \ell_1)^2$ entries but, from Proposition 7, it lies in a space of dimension $m(n - k - 1) + \ell_1(m + \ell_1)$. Hence one can specialize some variables and restrict to $m(m - n + \ell_1 + k + 1) + 1$ variables;
 - \mathbf{M} is in \mathcal{C}_{pub}^\perp which has dimension $(m + \ell_1)(n + \ell_2) - mk$, one can even specialize $m - 1$ variables since $(\mathcal{C}_{pub} + \mathbf{D}\mathcal{C}_{pub})^\perp$ has codimension at least m .
- **Number of equations:** it is nothing but $\dim_{\mathbb{F}_q} \mathcal{C}_{pub} = mk$.

Finally, recall that we assumed $2 \dim \mathcal{C}_{pub} = 2mk \geq (m + \ell_1)(n + \ell_2)$ which leads to the fact that, $\mathcal{C}_{pub} + \mathbf{D}\mathcal{C}_{pub}$ would fill in the ambient space if \mathcal{C}_{pub} was random. Indeed, it is easy to check that for a random matrix code \mathcal{C}_{rand} the dimension of $\mathcal{C}_{rand} + \mathbf{D}\mathcal{C}_{rand}$ is typically $\min(m + \ell_1(n + \ell_2), 2mk) = m + \ell_1(n + \ell_2)$ or equivalently that $\mathcal{C}_{rand} + \mathbf{D}\mathcal{C}_{rand}$ fills in the ambient space.

However, assuming that $k = \Theta(m)$ and $m \approx n$, we have $\Theta(m^2)$ bilinear equations with $\Theta(m^2)$ unknowns from \mathbf{D} and from \mathbf{M} . For the parameters we consider, with $m \approx 40$, this represents thousands of variables to handle. Thus, we claim that our system remains out of reach of such a distinguisher.

Attacking the Dual. The dual of an enhanced Gabidulin code is described in Proposition 1: after applying left and right multiplying by (P^t) and (Q^t) respectively, the dual of the public code has the following shape

$$\mathcal{D}_{mat} = \mathcal{U}_{n-k} \oplus \mathcal{W},$$

where \mathcal{U}_{n-k} is a matrix version of a Gabidulin code “extended by zero” *i.e.* to which ℓ_1 rows and ℓ_2 columns of zero have been added and \mathcal{W} is some complement subspace of dimension $n\ell_1 + m\ell_2 + \ell_1\ell_2$. We assume here that $2 \dim_{\mathbb{F}_q} \mathcal{C}_{mat} \geq (m + \ell_1)(n + \ell_2)$. Equivalently, we suppose the dual of the public code to have rate $> 1/2$

The code \mathcal{U}_{n-k} is derived from a Gabidulin code of \mathbb{F}_{q^m} -dimension $n - k$ and hence has \mathbb{F}_q -dimension $m(n - k)$. Similarly to the previous attack, observe that if $D \in \mathbb{F}_q^{(m+\ell_1) \times (m+\ell_1)}$ is in some matrix version of a Gabidulin code of dimension b extended by zero, then

$$(\mathcal{U}_{n-k} + D\mathcal{U}_{n-k}) \subseteq \mathcal{U}_{n-k+b-1},$$

where $\mathcal{U}_{n-k+b-1}$ is a matrix version of a Gabidulin code of \mathbb{F}_{q^m} -dimension $n - k + b - 1$ extended by zero. Thus,

$$\begin{aligned} \dim_{\mathbb{F}_q}(\mathcal{D}_{mat} + D\mathcal{D}_{mat}) &\leq \dim_{\mathbb{F}_q}(\mathcal{U}_{n-k} + D\mathcal{U}_{n-k}) + \dim_{\mathbb{F}_q} \mathcal{W} + \dim_{\mathbb{F}_q} D\mathcal{W} \\ &\leq m(n - k + b - 1) + 2(n\ell_1 + m\ell_2 + \ell_1\ell_2). \end{aligned}$$

Under the assumption that \mathcal{D}_{mat} has rate $> 1/2$, the code $\mathcal{D}_{mat} + D\mathcal{D}_{mat}$ would equal the ambient space w.h.p. if \mathcal{D}_{mat} was random, while, it actually does not as soon as:

$$m(n - k + b - 1) + 2(n\ell_1 + m\ell_2 + \ell_1\ell_2) < (m + \ell_1)(n + \ell_2),$$

which holds as soon as

$$b < k + 1 - \frac{n}{m}\ell_1 - \ell_2 - \frac{\ell_1\ell_2}{m}. \quad (5)$$

Here again as in the previous case, we can equate this distinguisher by choosing a b satisfying (5), searching $D \in \mathbb{F}_q^{(m+\ell_1) \times (m+\ell_1)}$ and $M \in \mathcal{C}_{mat}$ such that for all C in the dual of the public code, we have $\text{Tr}(DCM^t) = 0$.

As in the previous case, the number of unknowns remains prohibitive compared to the number of equations.

Remark 3. Solving such a system for $b = 0$ corresponds to searching the hidden \mathbb{F}_{q^m} -linearity of the code \mathcal{U}_{n-k} .

Remark 4. We also considered a combinatorial approach on the dual as we proposed on the public code itself. The attack on the public code consists in guessing a relevant puncturing of the row and column spaces in order to get rid of the contribution of the random rows and columns added in the enhancing process.

When reasoning on the dual, we need to get rid of the contribution of the complement subspace \mathcal{W} . This can be done by guessing a relevant shortening of the row and column space. Such a structural attack on the dual turns out to be equivalent to the previously presented combinatorial attack performed on the public code itself.

Finding Codewords Just Below the Singleton Bound and Exploiting Their Structure. In this subsection we consider the approach of finding matrices in the public code of weight the MRD bound minus 1. These matrices could be candidates for unraveling the structure of the hidden Gabidulin code. We first consider how the MRD bound is affected by the enhanced matrix codes transformation (i.e. adding random rows and columns). Then, we show there are many matrices of weight the MRD bound minus 1 in the public code and that their structure does not seem to reveal any information on the secret key.

To simplify the analysis, we only consider the case $n = m$ since all parameters in Sect. 7 satisfy this equality. Without loss of generality, since we can always consider the transposed code (i.e. the code $\mathcal{C}_{mat}^t = \{\mathbf{M}^t \mid \mathbf{M} \in \mathcal{C}_{mat}\}$, which is different from the dual code), we assume in this subsection $\ell_1 \geq \ell_2 > 0$.

Definition 20 (Singleton bound [16]). A matrix code $\mathcal{C}_{mat}[m \times n, K, d]$ satisfies the Singleton (or MRD) bound if:

$$d \leq \min(n, m) - \frac{K}{\max(n, m)} + 1.$$

Codes achieving the equality for this bound are called MRD codes.

Lemma 3. The minimum distance for the code $\mathcal{C}_{pub}[(m + \ell_1) \times (m + \ell_2), km, d]$ satisfies:

$$d \leq m - k + 1 + \ell_2 + \left\lfloor \frac{k\ell_1}{m + \ell_1} \right\rfloor.$$

Proof. The direct application of the Singleton bound yields:

$$d \leq m - k + 1 + \ell_2 + \frac{k\ell_1}{m + \ell_1}.$$

Since d is an integer, the integer part of $\frac{k\ell_1}{m + \ell_1}$ can be taken. \square

This shows that after perturbation, the Singleton bound is increased by at least ℓ_2 .

Let us denote $d_0 := m - k + 1 + \ell_2 + \left\lfloor \frac{k\ell_1}{m + \ell_1} \right\rfloor$. It is legitimate to wonder whether \mathcal{C}_{pub} achieves the MRD bound and in the contrary, how many matrices of rank $d_0 - 1$ are contained in \mathcal{C}_{pub} .

In the following we prove a lower bound on the expected number of matrices in \mathcal{C}_{pub} of rank $d_0 - 1$, which shows that \mathcal{C}_{pub} is highly unlikely to be an MRD code and that, moreover, it contains lots of matrices of rank $d_0 - 1$.

Lemma 4. *Let N be the expected number of matrices in \mathcal{C}_{pub} of rank $d_0 - 1$. It achieves the following inequality*

$$N \geq \left[\begin{matrix} m \\ m - k + 1 \end{matrix} \right]_q (q^m - 1) \times q^{(\ell_1 + 1)(1 - k) - 1} \approx q^{m + (m - k - \ell_1)(k - 1) - 1}.$$

Proof. The matrices in \mathcal{C}_{pub} are of the form

$$M = P \left(\begin{array}{c|c} \mathbf{G} & \\ \mathbf{A} & \mathbf{B} \end{array} \right) Q$$

with $\mathbf{G} \in \mathcal{G}_g(m, k, m)$ of size $m \times m$, \mathbf{A} of size $\ell_1 \times m$ and \mathbf{B} of size $(m + \ell_1) \times \ell_2$.

Sufficient conditions such that M is of rank $d_0 - 1$ are:

1. \mathbf{G} is of rank $d_0 - \ell_2$
2. $RowSpace(\mathbf{A}) \subset RowSpace(\mathbf{G})$
3. \mathbf{B} is of full rank ℓ_2
4. $\exists i \in [1, \ell_2], \mathbf{B}_{*,i} \in ColSpace \left(\begin{array}{c} \mathbf{G} \\ \mathbf{A} \end{array} \right)$
5. $ColSpace(\mathbf{B}_{*,i})_{j \in [1, \ell_2], j \neq i} \cap ColSpace \left(\begin{array}{c} \mathbf{G} \\ \mathbf{A} \end{array} \right) = \{0\}$

The number of matrices \mathbf{G} of rank $m - k + 1$ is $\left[\begin{matrix} m \\ m - k + 1 \end{matrix} \right]_q (q^m - 1)$ [27].

Because $d_0 - \ell_2 \geq m - k + 1$ and since the weight distribution in a Gabidulin code is increasing with the weight [11], the number of matrices \mathbf{G} of rank $d_0 - \ell_2$

is $\geq \left[\begin{matrix} m \\ m - k + 1 \end{matrix} \right]_q (q^m - 1)$. It remains now to evaluate the probabilities of 2.-5.

We assume that \mathbf{A} and \mathbf{B} are uniformly random independent matrices.

The probability that a single row of \mathbf{A} is contained in $RowSpace(\mathbf{G})$, which is a subspace of \mathbb{F}_q^m of dimension $m - k + 1$ is q^{1-k} . Hence the probability of 2. is $q^{\ell_1(1-k)}$.

The probability of 3. is $\prod_{j=0}^{\ell_2-1} (1 - q^{j-\ell_2-m}) \geq (1 - q^m)^{\ell_2} \geq q^{-1}$.

Since \mathbf{B} is of full rank, $\dim ColSpace(\mathbf{B}_{*,i})_{j \in [1, \ell_2], j \neq i} = \ell_2 - 1$. Noting that

$\dim ColSpace \left(\begin{array}{c} \mathbf{G} \\ \mathbf{A} \end{array} \right) = m - k + 1$, the probability of 4. and 5. together is

$$\ell_2 q^{(1-k)} (1 - q^{(\ell_2-1)(1-k)}) \geq q^{1-k}.$$

□

The above lemma shows that for the parameters presented in Sect. 7, the expected number of matrices of rank $d_0 - 1$ is very large (at least q^{128} for all parameters presented in Fig. 5) and does not seem to yield any information to retrieve the secret Gabidulin structure.

The above proof can be easily adapted in the special case of $\ell_2 = 0$ to find a similar upper bound as long as $\frac{k\ell_1}{m+\ell_1} \geq 1$, which is always the case in our parameters.

6.2 Attacks on the Message

In all encryption schemes derived from the MinRank-McEliece and Niederreiter encryption frames, an opponent who wants to retrieve the original message without knowing the secret key has to solve a generic MinRank(q, m, n, k, r) instance. We recall here the main attacks on this problem.

Hybrid Approach. In order to improve the attacks on MinRank, a generic approach has been introduced in [9], which consists in solving smaller instances. The complexity is given by:

$$\min_a q^{ar} \mathcal{A}(q, m, n - a, K - am, r) \quad (6)$$

where \mathcal{A} is the cost of an algorithm to solve a MinRank instance.

The Kernel Attack. This attack was described in [25]. The idea of the attack consists in sampling random vectors, hoping that they are in the kernel of \mathbf{E} , and deducing a linear system of equations. Its complexity is equal to:

$$O(q^{r \lceil \frac{k}{m} \rceil} k^\omega). \quad (7)$$

Minors Attack. This algebraic attack was introduced and studied in [18]. This modeling uses the minors of \mathbf{E} . This method has been improved in [24]. We refer to these papers for the complexity of the attack.

Support Minors Attack. The Support Minors modeling was introduced in [10]. This idea also uses minors of a matrix, giving us another system of equations. With this approach, the complexity is of:

$$O(N_b M_b^{\omega-1}) \quad (8)$$

where

$$N_b = \sum_{i=1}^b (-1)^{i+1} \binom{n}{r+i} \binom{k+b-i-1}{b-i} \binom{k+i-1}{i}$$

$$M_b = \binom{n}{r} \binom{k+b-1}{b}$$

and b is the degree to which we augment the Macaulay matrix of the system.

7 Parameters

7.1 Parameters for EGMC-Niederreiter Encryption Scheme

We apply our idea of Enhanced MinRank-Niederreiter encryption frame by taking a Gabidulin code $\mathcal{G}[m, k]_{q^m}$, for which we know an efficient decoding algorithm, allowing to decode errors of weight up to $\lfloor \frac{m-k}{2} \rfloor$. As we previously said, decrypting the ciphertext \mathbf{Y} without the secret key sk is strictly equivalent to the MinRank problem of parameters $(q, m + \ell_1, m + \ell_2, km, r)$.

Choice of Parameters. For a given value of r , we choose the values of m and k (the optimal parameters verify $r = \lfloor \frac{m-k}{2} \rfloor$) such that the underlying MinRank instance is secure against the known attacks. Then, we choose ℓ_1 and ℓ_2 to obtain parameters resistant to structural attacks (see below).

We choose the parameters based on the best known attacks, see Fig. 5 and Fig. 6. Among these attacks, three of them rely on solving the associated MinRank instance: *Alg.* refers to the algebraic Support Minors attack (see Eq. 8), *Hyb.* refers to the Hybrid approach (see Eq. 6), and *Comb.* refers to the combinatorial Kernel attack (see Eq. 7). We also consider the attack which consists in retrieving the \mathbb{F}_{q^m} -linear structure of the code \mathcal{C}_{mat} (see Eq. 4).

We propose two kinds of parameters. In our main parameters, we add as many rows as columns to the matrix Gabidulin code ($\ell_1 = \ell_2$). The resulting parameters for 128 bits, 192 bits and 256 bits of security can be found in Fig. 5. We also propose some sets of parameters for which only rows or only columns are added to the matrix Gabidulin code ($\ell_1 = 0$ or $\ell_2 = 0$), and which can be found in Fig. 6. We consider NIST-compliant parameters, while maintaining a margin of 15 bits above the security level.

The public key pk consists of a parity check matrix of \mathcal{C}'_{mat} . As a linear matrix code $[(m + \ell_1)(m + \ell_2), km]_q$, it can be represented with:

$$km((m + \ell_1)(m + \ell_2) - km) \log_2 q$$

bits. The number of bits necessary to represent the ciphertext $\mathbf{c} \in \mathbb{F}_{q^m}^{(m+\ell_1)(m+\ell_2)-km}$ is equal to:

$$((m + \ell_1)(m + \ell_2) - km) \log_2 q.$$

7.2 Comparison with Other Schemes

We propose a comparison of our sizes with those of other encryption schemes based on various problems, see Figs. 7, 8 and 9. Note that we achieve better performances than RQC and ROLLO, which are other rank-based encryption schemes, and even the smallest ciphertext sizes compared to the schemes proposed to the NIST based on codes and lattices.

Sec.	q	k	m	ℓ_1	ℓ_2	r	Alg.	Hyb.	Comb.	Struc.	pk	ct
128	2	17	37	3	3	10	193	170	179	165	76 kB	121 B
	2	25	37	3	3	6	168	150	164	189	78 kB	84 B
	2	35	43	2	2	4	158	145	158	158	98 kB	65 B
	2	47	53	2	2	3	158	147	161	202	166 kB	66 B
192	2	51	59	2	2	4	222	209	224	222	268 kB	89 B
256	2	23	47	3	3	12	302	271	285	284	191 kB	177 B
	2	37	53	3	2	8	315	290	310	273	274 kB	139 B
	2	71	79	2	2	4	303	289	305	302	667 kB	119 B

Fig. 5. Reference parameters for the EGMC-Niederreiter encryption scheme

Sec.	q	k	m	ℓ_1	ℓ_2	r	Alg.	Hyb.	Comb.	Struc.	pk	ct
128	2	17	37	4	0	10	181	168	179	148	70 kB	111 B
	16	13	23	1	1	5	236	273	282	148	41 kB	138 B
	16	7	23	0	5	8	172	262	276	160	33 kB	207 B
192	2	23	43	5	0	10	239	220	230	215	133 kB	134 B
	2	33	47	5	0	7	238	221	232	235	173 kB	111 B
	2	41	53	4	0	6	258	240	257	212	230 kB	106 B
256	16	9	29	2	1	10	310	373	382	272	87 kB	334 B
	16	17	29	2	1	8	357	399	408	304	107 kB	218 B

Fig. 6. Alternative parameters in particular case of $\ell_1 = 0$ or $\ell_2 = 0$, or $q > 2$

Scheme	pk	ct
EGMC-Niederreiter, Fig. 4	98 kB	65 B
Classic McEliece [12]	261 kB	96 B
EGMC-Niederreiter, Fig. 4	33 kB	207 B
ROLLO I [6]	696 B	696 B
KYBER [8]	800 B	768 B
RQC-Block-NH-MS-AG [7]	312 B	1118 B
BIKE [1]	1540 B	1572 B
RQC-NH-MS-AG [13]	422 B	2288 B
RQC [2]	1834 B	3652 B
HQC [3]	2249 B	4481 B

Fig. 7. Comparison of different schemes for 128 bits of security

Scheme	pk	ct
EGMC-Niederreiter, Fig. 4	268 kB	89 B
EGMC-Niederreiter, Fig. 4	133 kB	134 B
Classic McEliece [12]	524 kB	156 B
ROLLO I [6]	958 B	958 B
KYBER [8]	1184 B	1088 B
RQC-Block-NH-MS-AG [7]	618 B	2278 B
BIKE [1]	3082 B	3024 B
RQC-NH-MS-AG [13]	979 B	3753 B
RQC [2]	2853 B	5690 B
HQC [3]	4522 B	9026 B

Fig. 8. Comparison of different schemes for 192 bits of security

Scheme	pk	ct
EGMC-Niederreiter , Fig. 4	667 kB	119 B
Classic McEliece [12]	1044 kB	208 B
EGMC-Niederreiter , Fig. 4	87 kB	334 B
ROLLO I [6]	1371 B	1371 B
KYBER [8]	1568 B	1568 B
BIKE [1]	5121 B	5153 B
RQC [2]	4090 B	8164 B
HQC [3]	7245 B	14465 B

Fig. 9. Comparison of different schemes for 256 bits of security

8 Conclusion and Further Work

This work presents a general McEliece-like encryption frame for matrix codes whose security is based on the MinRank problem. We propose a general masking for rank codes that we apply on a matrix code version of Gabidulin vector code. We define a new problem: the Enhanced Gabidulin Matrix Code (EGMC) distinguishing problem, for which we propose a thorough analysis and study possible distinguishers to solve it. It results in a competitive encryption scheme, which achieves a ciphertext size of 65B for 128 bits of security.

For future work, it would be interesting to extend our trapdoor by also considering subcodes of matrix codes that we obtain with our masking. Such modifications would probably make the action of the distinguisher more complex, and thus may permit to reduce the parameters of our scheme.

Acknowledgements. The authors express their deep gratitude to the anonymous referees for their help relevant comments that clearly contributed in improving the article. The authors are supported by the *Plan France 2030* under the project ANR-22-PETQ-0008 and by the French *Agence Nationale de la recherche*, under the grant ANR-21-CE39-0009-BARRACUDA. The second author is supported by Horizon-Europe MSCA-DN project *Encode*.

References

1. Carlos Aguilar Melchor, Nicolas Aragon, Paulo Barreto, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Shay Gueron, Tim Güneysu, Rafael Misoczki, Edoardo Persichetti, Nicolas Sendrier, Jean-Pierre Tillich, and Gilles Zémor. BIKE. First round submission to the NIST post-quantum cryptography call, November 2017.
2. Carlos Aguilar Melchor, Nicolas Aragon, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, and Gilles Zémor. Rank quasi cyclic (RQC). First round submission to the NIST post-quantum cryptography call, November 2017.

3. Carlos Aguilar Melchor, Nicolas Aragon, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Edoardo Persichetti, Gilles Zémor, and Jurjen Bos. HQC. Round 3 Submission to the NIST Post-Quantum Cryptography Call, June 2021. https://pqc-hqc.org/doc/hqc-specification_2021-06-06.pdf.
4. Martin Albrecht, Daniel J. Bernstein, Tung Chou, Carlos Cid, Jan Gilcher, Tanja Lange, Varun Maram, Ingo von Maurich, Rafael Mizoczki, Ruben Niederhagen, Edoardo Persichetti, Kenneth Paterson, Christiane Peters, Peter Schwabe, Nicolas Sendrier, Jakub Szefer, Cen Jung Tjhai, Martin Tomlinson, and Wang Wen. Classic McEliece (merger of Classic McEliece and NTS-KEM). <https://classic.mceliece.org>, November 2022. Fourth round finalist of the NIST post-quantum cryptography call.
5. Alekhnovich, Michael. More on Average Case vs Approximation Complexity. In *44th Symposium on Foundations of Computer Science (FOCS 2003), 11-14 October 2003, Cambridge, MA, USA, Proceedings*, pages 298–307. IEEE Computer Society, 2003.
6. Nicolas Aragon, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Adrien Hauteville, Olivier Ruatta, Jean-Pierre Tillich, Gilles Zémor, Carlos Aguilar Melchor, Slim Bettaieb, Loïc Bidoux, Magali Bardet, and Ayoub Otmani. ROLLO (merger of Rank-Ouroboros, LAKE and LOCKER). Second round submission to the NIST post-quantum cryptography call, March 2019.
7. Nicolas Aragon, Pierre Briaud, Victor Dyseryn, Philippe Gaborit, and Adrien Vinçotte. The blockwise rank syndrome learning problem and its applications to cryptography. Cryptology ePrint Archive, Paper 2023/1875, 2023.
8. Roberto Avanzi, Joppe Bos, Léo Ducas, Eike Kiltz, Tancrede Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, , and Damien Stehlé. Crystals-kyber. Third round submission to the NIST post-quantum cryptography call, August 2021.
9. Magali Bardet, Pierre Briaud, Maxime Bros, Philippe Gaborit, and Jean-Pierre Tillich. Revisiting algebraic attacks on MinRank and on the rank decoding problem, 2022. [ArXiv:2208.05471](https://arxiv.org/abs/2208.05471).
10. Magali Bardet, Maxime Bros, Daniel Cabarcas, Philippe Gaborit, Ray Perlner, Daniel Smith-Tone, Jean-Pierre Tillich, and Javier Verbel. Improvements of algebraic attacks for solving the rank decoding and MinRank problems. In *Advances in Cryptology - ASIACRYPT 2020, International Conference on the Theory and Application of Cryptology and Information Security, 2020. Proceedings*, pages 507–536, 2020.
11. Hannes Bartz, Lukas Holzbaur, Hedongliang Liu, Sven Puchinger, Julian Renner, Antonia Wachter-Zeh, et al. Rank-metric codes and their applications. *Foundations and Trends® in Communications and Information Theory*, 19(3):390–546, 2022.
12. Daniel J. Bernstein, Tung Chou, Tanja Lange, Ingo von Maurich, Rafael Mizoczki, Ruben Niederhagen, Edoardo Persichetti, Christiane Peters, Peter Schwabe, Nicolas Sendrier, Jakub Szefer, and Wang Wen. Classic McEliece: conservative code-based cryptography. <https://classic.mceliece.org>, November 2017. First round submission to the NIST post-quantum cryptography call.
13. Loïc Bidoux, Pierre Briaud, Maxime Bros, and Philippe Gaborit. RQC revisited and more cryptanalysis for rank-based cryptography. [ArXiv:2207.01410](https://arxiv.org/abs/2207.01410), 2022.

14. Alain Couvreur, Thomas Debris-Alazard, and Philippe Gaborit. On the hardness of code equivalence problems in rank metric. working paper or preprint, November 2020.
15. Alain Couvreur, Philippe Gaborit, Valérie Gautier, Ayoub Otmani, and Jean-Pierre Tillich. Distinguisher-Based Attacks on Public-Key Cryptosystems Using Reed-Solomon Codes. In *International Workshop on Coding and Cryptography - WCC 2013*, pages 181–193, Bergen, Norway, April 2013.
16. Philippe Delsarte. Bilinear forms over a finite field, with applications to coding theory. *J. Comb. Theory, Ser. A*, 25(3):226–241, 1978.
17. Jean-Charles Faugère, Valérie Gauthier, Ayoub Otmani, Ludovic Perret, and Jean-Pierre Tillich. A distinguisher for high rate McEliece cryptosystems. IACR Cryptology ePrint Archive, Report2010/331, 2010. <http://eprint.iacr.org/>.
18. Jean-Charles Faugère, Mohab Safey El Din, and Pierre-Jean Spaenlehauer. Computing loci of rank defects of linear matrices using Gröbner bases and applications to cryptology. In *International Symposium on Symbolic and Algebraic Computation, ISSAC 2010, Munich, Germany, July 25-28, 2010*, pages 257–264, 2010.
19. Ernst M. Gabidulin. Theory of codes with maximum rank distance. *Problemy Peredachi Informatsii*, 21(1):3–16, 1985.
20. Ernst M. Gabidulin and Alexei V. Ourivski. Modified GPT PKC with right scrambler. *Electron. Notes Discrete Math.*, 6:168–177, 2001.
21. Philippe Gaborit, Ayoub Otmani, and Hervé Talé-Kalachi. Polynomial-time key recovery attack on the Faure-Loidreau scheme based on Gabidulin codes. *Des. Codes Cryptogr.*, 86(7):1391–1403, 2018.
22. Keith Gibson. Severely denting the Gabidulin version of the McEliece public key cryptosystem. *Des. Codes Cryptogr.*, 6(1):37–45, 1995.
23. Keith Gibson. The security of the Gabidulin public key cryptosystem. In Ueli Maurer, editor, *Advances in Cryptology - EUROCRYPT '96*, volume 1070 of *LNCS*, pages 212–223. Springer, 1996.
24. Sriram Gopalakrishnan, Vincent Neiger, and Mohab Safey El Din. Refined f_5 algorithms for ideals of minors of square matrices, 2023.
25. Louis Goubin and Nicolas Courtois. Cryptanalysis of the TTM cryptosystem. In Tatsuaki Okamoto, editor, *Advances in Cryptology - ASIACRYPT 2000*, volume 1976 of *LNCS*, pages 44–57. Springer, 2000.
26. Loo-Keng Hua. A theorem on matrices over a field and its applications. *J. Chinese Math. Soc.*, 1(2):109–163, 1951.
27. Pierre Loidreau. *Rank metric and cryptography*. Accreditation to supervise research, Université Pierre et Marie Curie - Paris VI, January 2007.
28. Robert J. McEliece. *A Public-Key System Based on Algebraic Coding Theory*, pages 114–116. Jet Propulsion Lab, 1978. DSN Progress Report 44.
29. Rafael Misoczki, Jean-Pierre Tillich, Nicolas Sendrier, and Paulo S. L. M. Barreto. MDPC-McEliece: New McEliece variants from moderate density parity-check codes, 2012.
30. Harald Niederreiter. Knapsack-type cryptosystems and algebraic coding theory. *Problems of Control and Information Theory*, 15(2):159–166, 1986.
31. Alexey Ourivskiy and Ernst Gabidulin. Column scrambler for the GPT cryptosystem. *Discrete Applied Mathematics*, 128:207–221, 05 2003.
32. Raphael Overbeck. A new structural attack for GPT and variants. In *Mycrypt*, volume 3715 of *LNCS*, pages 50–63, 2005.

33. Raphael Overbeck. Structural attacks for public key cryptosystems based on Gabidulin codes. *J. Cryptology*, 21(2):280–301, 2008.
34. Alberto Ravagnani. Rank-metric codes and their duality theory. *Des. Codes Cryptogr.*, 80:197–216, 2016.
35. Christian Wieschebrink. Two NP-complete problems in coding theory with an application in code based cryptography. In *Proc. IEEE Int. Symposium Inf. Theory - ISIT*, pages 1733–1737, 2006.



Tighter Proofs for PKE-to-KEM Transformation in the Quantum Random Oracle Model

Jinrong Chen¹, Yi Wang¹, Rongmao Chen^{1(✉)}, Xinyi Huang², and Wei Peng¹

¹ College of Computer Science and Technology, National University of Defense Technology, Changsha 410073, Hunan, China

{jinrongchen, wangyi14, chromao, wpeng}@nudt.edu.cn

² College of Cyber Security, Jinan University, Guangzhou 510632, Guangdong, China

Abstract. In this work, we provide new, tighter proofs for the T_{RH} -transformation by Jiang et al. (ASIACRYPT 2023), which converts OW-CPA secure PKEs into KEMs with IND-1CCA security, a variant of typical IND-CCA security where only a single decapsulation query is allowed. Such KEMs are efficient and have been shown sufficient for real-world applications by Huguenin-Dumittan and Vaudenay at EUROCRYPT 2022. We reprove Jiang et al.'s T_{RH} -transformation in both the random oracle model (ROM) and the quantum random oracle model (QROM), for the case where the underlying PKE is rigid deterministic. In both ROM and QROM models, our reductions achieve security loss factors of $\mathcal{O}(1)$, significantly improving Jiang et al.'s results which have security loss factors of $\mathcal{O}(q)$ in the ROM and $\mathcal{O}(q^2)$ in the QROM respectively. Notably, central to our tight QROM reduction is a new tool called “reprogram-after-measure”, which overcomes the reduction loss posed by oracle reprogramming in QROM proofs. This technique may be of independent interest and useful for achieving tight QROM proofs for other post-quantum cryptographic schemes. We remark that our results also improve the reduction tightness of the T_H -transformation (which also converts PKEs to KEMs) by Huguenin-Dumittan and Vaudenay (EUROCRYPT 2022), as Jiang et al. provided a tight reduction from T_H -transformation to T_{RH} -transformation (ASIACRYPT 2023).

Keywords: QROM · Security proof · Tight reduction · 1CCA security · KEM

1 Introduction

Indistinguishability against Chosen-Ciphertext Attacks (IND-CCA) has been widely considered as the security standard for post-quantum key encapsulation mechanisms (KEMs) [10, 20, 34–37, 40, 47], which could be achieved by applying the Fujisaki-Okamoto-like (FO-like) transformation [27] to public-key encryption (PKE) with security weaker than IND-CCA. However, in the post-quantum

Table 1. The reduction tightness of transformations from OW-CPA-secure deterministic PKE to IND-1CCA-secure KEM in the ROM/QROM. Here ϵ_R represents the advantage of the reduction algorithm R with respect to the OW-CPA security of the underlying PKE scheme, $\epsilon_{\mathcal{A}}$ represents the advantage of the adversary \mathcal{A} with respect to the IND-1CCA security of the obtained KEM scheme, and q is the total number of random oracle queries made by \mathcal{A} .

Model	Transformation	Reduction tightness
ROM	T_{CH}^D [31]	$\epsilon_R \approx \epsilon_{\mathcal{A}}$ [31]
	T_H^D [31]	$\epsilon_R \approx \mathcal{O}(1/q^2)\epsilon_{\mathcal{A}}$ [31] $\epsilon_R \approx \mathcal{O}(1/q)\epsilon_{\mathcal{A}}$ [33]
	T_{RH}^D [33]	$\epsilon_R \approx \mathcal{O}(1/q)\epsilon_{\mathcal{A}}$ [33] $\epsilon_R \approx \epsilon_{\mathcal{A}}$ (Our work)
QROM	T_{CH}^D [31]	$\epsilon_R \approx \mathcal{O}(1/q^3)\epsilon_{\mathcal{A}}^2$ [31]
	T_H^D [31]	$\epsilon_R \approx \mathcal{O}(1/q^2)\epsilon_{\mathcal{A}}^2$ [33]
	T_{RH}^D [33]	$\epsilon_R \approx \mathcal{O}(1/q^2)\epsilon_{\mathcal{A}}^2$ [33] $\epsilon_R \approx \epsilon_{\mathcal{A}}^2$ (Our work)

cryptography (PQC) migration, it has been shown that IND-1CCA-secure KEM is sufficient to replace the Diffie-Hellman key exchange in TLS 1.3 [21] and Signal [14] to achieve post-quantum security [31]. Compared to IND-CCA security, IND-1CCA security allows the adversary to make only a single decapsulation query. This restriction enables more efficient transformations [31, 33] than the FO-like approach, as it removes the need for the time-consuming re-encryption operation in the decapsulation algorithm. In particular, Huguenin-Dumittan and Vaudenay [31] pointed out that omitting the re-encryption step could speed up the decapsulation algorithm of Kyber [13] and Frodo-AES [2] by 2.17 times and 6.11 times, respectively. Besides, removing the re-encryption operation might enhance the security of the obtained KEM against side-channel attacks [49].

To design IND-1CCA-secure KEMs, Huguenin-Dumittan and Vaudenay [31] proposed two transformations called T_{CH} and T_H , both of which build KEMs from PKE schemes with One-Wayness against Chosen-Plainxt Attacks (OW-CPA). In particular, T_{CH} is a variant of the REACT transformation [43], and T_H is the same as the U^\perp transformation in [27]. Later, Jiang et al. [33] presented an implicit variant of T_H called T_{RH} where the decapsulation algorithm returns a pseudo-random value instead of an explicit abort symbol for an invalid ciphertext. Also, they provided tighter proofs for T_H by reducing its IND-1CCA security to the IND-1CCA security of T_{RH} .

Table 1 lists the reduction tightness of these transformations with deterministic PKE in the random oracle model (ROM) [7] and the quantum random oracle model (QROM) [11]. Hereafter, we will use T_X^D to denote T_X with deterministic

PKE for $X \in \{CH, H, RH\}$. As shown in Table 1, the ROM proof of T_{CH}^D is almost tight, but the QROM proof of T_{CH}^D requires an additional hash function for ciphertext verification which increases the size of ciphertext. In contrast, the QROM proofs of T_H^D and T_{RH}^D in [33] do not need ciphertext expansion.

Jiang et al. [33] not only made improvements on the reduction tightness of T_H , but also proved that the reduction losses $\mathcal{O}(q)$ and $\mathcal{O}(q^2)$ are unavoidable in the ROM and QROM proofs of T_{RH} respectively. However, in this work we found that these reduction losses could be further reduced to $\mathcal{O}(1)$ when the underlying PKEs are rigid deterministic (See Sect. 1.2 for detailed explanation). These tight security reductions could improve the practical efficiency of KEMs built via the T_{RH}^D due to no need to increase the security parameter to compensate for the loss factor.

1.1 Our Contributions

In this work, we provide new, tighter proofs for the T_{RH} -transformation by Jiang et al. [33] when the underlying PKE is rigid deterministic¹, as shown in Table 1, and our contributions are as follows.

First, we present a tight security proof with loss factor $\mathcal{O}(1)$ for T_{RH}^D in the ROM (Theorem 2). In this proof, we propose a new strategy to simulate the decapsulation oracle successfully with probability $1/2$. This strategy takes full advantage of the rigid deterministic property of PKE, and does not have to guess the random oracle query of adversary.

Second, we extend the above strategy to the QROM and obtain a tight security proof with loss factor $\mathcal{O}(1)$ for T_{RH}^D in the QROM (Theorem 4). At the core of our QROM proof for T_{RH}^D is a novel technique called *reprogram-after-measure*, which is used to handle the issue of random oracle reprogramming in the QROM.

Compared with existing techniques including one-way to hiding (O2H) [3, 10, 40, 51] and measure-and-reprogram [18, 19], our technique is tailored for this particular case and introduces a reduction loss of $\mathcal{O}(1)$ only. Note that our results also improve the reduction tightness of T_H^D [31], as Jiang et al. [33] provided a tight reduction from T_H to T_{RH} .

1.2 Results Overview

T_{RH} transformation is shown in Fig. 1, where \mathcal{M} and \mathcal{C} are the message space and the ciphertext space of the underlying PKE scheme $\text{PKE}' = (\text{Gen}', \text{Enc}', \text{Dec}')$, respectively, \mathcal{K} is the key space of KEM_{RH} , \star is a fixed public value, and H is a hash function mapping from $\mathcal{M} \cup \{\star\} \times \mathcal{C}$ to \mathcal{K} . For simplicity, we only consider the case of $\star \in \mathcal{M}$, and the case of $\star \notin \mathcal{M}$ can be proved similarly.

¹ The property of “rigidity” is studied by Bernstein and Persichetti [8]. Roughly speaking, it means that $\text{Enc}(\text{pk}, m) = c$ for every $(\text{pk}, \text{sk}) \leftarrow \text{Gen}(1^\lambda)$, $c \in \mathcal{C}$, and $m := \text{Dec}(\text{sk}, c)$.

Gen(1^λ)	Encaps(pk)	Decaps(sk, c)
1 : (pk, sk) \leftarrow Gen'(1^λ)	1 : $m \leftarrow \mathcal{M}$	1 : $m' := \text{Dec}'(\text{sk}, c)$
2 : return (pk, sk)	2 : $c \leftarrow \text{Enc}'(\text{pk}, m)$	2 : if $m' = \perp$ then
	3 : $k := H(m, c)$	3 : return $k' := H(\star, c)$
	4 : return (k, c)	4 : return $k' := H(m', c)$

Fig. 1. $\text{KEM}_{RH} := T_{RH}[\text{PKE}', H]$.

On the Reduction Tightness of T_{RH} by Jiang et al. [33]. Theorem 5.1 in [33] says that the reduction loss factors $\mathcal{O}(q)$ and $\mathcal{O}(q^2)$ are unavoidable in the ROM and QROM proofs of T_{RH} when the underlying PKE is malleable. The proof of this theorem describes a ROM/QROM adversary \mathcal{B} against the IND-1CCA security of T_{RH} . In specific, given $c^* \leftarrow \text{Enc}'(\text{pk}, m^*)$ and k^* , \mathcal{B} needs to determine whether $k^* = H(m^*, c^*)$ or k^* is a random value over \mathcal{K} . By the malleability of PKE' , \mathcal{B} first derives a new c' from c^* where $c' = \text{Enc}'(\text{pk}, f(m^*))$ and f is the function associated to the malleability of PKE' . Then, \mathcal{B} makes the single decapsulation query on c' and receives $\text{tag} = H(f(m^*), c')$. Now \mathcal{B} makes random oracle queries to find $m^* \in \mathcal{M}$ such that $H(f(m^*), c') = \text{tag}$, and computes $H(m^*, c^*)$ to check whether k^* is random or not. Let q be the total number of random oracle queries made by \mathcal{B} , Jiang et al. [33] pointed out that these q random oracle queries contribute to unavoidable loss factors of $\mathcal{O}(q)$ and $\mathcal{O}(q^2)$ in the ROM and QROM.

Note that if $\text{Enc}'(\text{pk}, \cdot)$ is rigid deterministic, \mathcal{B} could find correct m^* by computing $\text{Enc}'(\text{pk}, \cdot)$ and comparing with c^* instead of querying random oracle, and the loss factors in the ROM and QROM could be avoided. This fact implies that it might be possible to improve the reduction tightness of T_{RH}^D by Jiang et al. [33].

Our Result I: Tight ROM Proof of T_{RH}^D . As pointed out by Jiang et al. [33], the core of the ROM proof is simulating decapsulation oracle without sk. The simulation of hash function H relies on a hash list to record all the random oracle queries and corresponding hash values. The ROM proof of T_{RH}^D in [33] is based on the fact that the decapsulation oracle always makes a random oracle query to generate k' and one could find the corresponding query from the hash list of H , say the i^* -th entry, where $i^* \in \{0, \dots, q\}$ and q is the total number of random oracle queries made by \mathcal{A} . So, the simulator of decapsulation oracle first randomly selects $i^* \in \{0, \dots, q\}$. If the i^* -th entry is not empty when \mathcal{A} queries the decapsulation oracle, it returns the hash value of this entry; otherwise, it returns a random $k^* \in \mathcal{K}$ and when \mathcal{A} makes the i^* -th hash query, k^* is returned. Therefore, the probability of a successful simulation is $1/(q + 1)$.

To achieve tighter proof, we present a new simulation strategy. That is, determining the way to compute k' in decapsulation oracle based on a correct guess on $m' \neq \perp$ with probability $1/2$. In the case of a correct guess on $m' \neq \perp$,

assuming PKE' is perfectly correct, the simulator of decapsulation oracle first checks whether there is a pair (m', c) in the hash list such that $\text{Enc}'(\text{pk}, m') = c$:

- If such a pair exists, then $k' := H(m', c)$. The perfect correctness and the rigidity of the deterministic PKE' guarantee that if $\text{Enc}'(\text{pk}, m') = c$, then $\text{Dec}'(\text{sk}, c) = m'$.
- Otherwise, \mathcal{A} does not have any knowledge of $H(m', c)$. Responding with $k' := k^*$, where $k^* \in \mathcal{K}$ is a random value, implicitly assigns k^* to $H(m', c)$ and would not be noticed by \mathcal{A} . After this, if \mathcal{A} makes a random oracle query on this pair, the random oracle should return k^* .

This completes the simulation of the decapsulation oracle without the knowledge of sk . The probability of a successful simulation is $1/2$, and the loss factor of our proof is 2. Note that if PKE' is δ -correct where $\delta \neq 0$, i.e., PKE' is not perfectly correct, then there will be an error term δ in our reduction result.

Our Result II: Tight QROM Proof of T_{RH}^D . Note that, in the QROM, since \mathcal{A} can make the random oracle queries in superposition, there is no such a hash list that can copy down \mathcal{A} 's queries and their responses, which implies that we cannot implicitly reprogram $H(m', c)$ to the random k^* as above. So, we propose following technique to fix this issue.

A New Tool: Reprogram-after-Measure. We present a simulator that can use a random value to simulate the decapsulation oracle without the knowledge of sk . This simulator simulates the random oracle using Zhandry's compressed oracle technique [55], which can record information about the adversary's quantum queries into a database in superposition without being detected by adversary. Assuming PKE' is perfectly correct and rigid deterministic, we can still use the simulation strategy in the ROM, i.e., guessing whether m' is equal to \perp or not with probability $1/2$. When $m' \neq \perp$ and the guess is correct, we have $\text{Enc}'(\text{pk}, m') = c$, as Enc' is rigid deterministic and perfectly correct. Then we could find the pair (m', c) that satisfies $\text{Enc}'(\text{pk}, m') = c$ in the database, and store the responses in a quantum register in superposition. Now we measure this register in the computational basis to get the classical response to (m', c) . This response may be in two cases: $k^* \in \mathcal{K}$, or $\perp \notin \mathcal{K}$. The latter implies that $H(m', c)$ has not been defined, so we use a random $k^* \leftarrow \mathcal{K}$ to replace it. Now, we let k^* be the response to the decapsulation oracle query on c . To make the random oracle responses consistent, in the subsequent random oracle query, we respond with k^* if the query is (m', c) , or still use the compressed oracle to obtain the responses otherwise. This completes the decapsulation simulation and the proof sketch in the QROM. For generality, we further extend this method into a reprogram-after-measure technique, which can address the oracle reprogramming issue encountered during the single classical query in the QROM, and is proved in Sect. 4.1.

The Proof in the QROM. Similar to the ROM proof of T_{RH}^D (see Theorem 4), the QROM proof also can be divided into following two steps:

1. The first step (games G_0 to G_4): We use a random $k \leftarrow \mathcal{K}$ to replace the $k := H(m, c)$ in `Encaps`, and use the double-sided O2H lemma (Lemma 1) to convert the advantage of \mathcal{A} detecting this change to the probability of a new adversary \mathcal{B} outputting the corresponding m .
2. The second step (games G_5 to G_8): We use the proposed reprogram-after-measure technique to simulate the decapsulation oracle without sk , and then use the ability of \mathcal{B} to attack the OW-CPA security of PKE' .

The tightness of this QROM proof results from our reprogram-after-measure technique that has a tighter upper bound than the measure-and-reprogram technique used in [33].

1.3 Related Work

The quantum random oracle model (QROM) [11] has been a popular model to analyze the security of some post-quantum cryptographic schemes, such as encryption [38, 54], signature [1, 9, 24, 46], authenticated key exchange (AKE) [30, 41, 44], classical verification of quantum computations (CVQC) [6, 15], and other cryptographic primitives [4, 29, 32]. Many works [53, 56] showed that there exist schemes that are secure in the ROM but insecure in the QROM, which implies that the QROM is stronger than the ROM.

T_{CH} , T_H , and T_{RH} can be seen as the simplified versions of the FO-like transformation, where the FO-like transformation is a variant of the Fujisaki-Okamoto transformation [22, 23] under KEM. Targhi et al. [50] and Hofheinz et al. [27] conducted the first analyses of the security of FO transformation and FO-like transformation in the QROM, respectively. However, these works need to introduce an additional hash function to achieve post-quantum security, and the proofs suffer from the quartic reduction loss. For the case where the KEM is implicit reject, Jiang et al. [34] provided a proof for the FO-like transformation without the additional hash, where the degree the reduction loss is decreased from quartic to quadratic, and the factor of the reduction loss is $\mathcal{O}(q^2)$. Jiang et al. [37] further pointed out that quadratic loss is unavoidable in the measurement-based black-box reduction, where the adversary is accessed in a black-box manner and is only run once without rewinding, and the reduction algorithm is performed by measuring the state of the adversary. In the following works, Jiang et al. [36] used the semi-classical O2H lemma proposed by Ambainis [3] to improve the security reduction to $\epsilon_R \approx \mathcal{O}(q)\epsilon_{\mathcal{A}}^2$, while Bindel et al. [10] proposed the double-sided O2H lemma to improve the security reduction to $\epsilon_R \approx \epsilon_{\mathcal{A}}^2$. To investigate a tighter transformation, Saito et al. [47] proposed the SXY transformation based on the FO-like transformation, and got a tight security reduction to the newly defined security called disjoint simulatability. This tight result is extended by Jiang et al. [35] to the KEM with explicit reject. Considering stronger quantum adversaries, Xagawa and Yamakawa [52] further proved the IND-QCCA security² of these PKE-to-KEM transformations [35, 47] in the QROM. To remove the

² The decapsulation oracle can also be accessed in superposition.

quadratic loss, Kuchta et al. [40] provided the measure-rewind-measure lemma and obtained a security reduction with tightness $\epsilon_R \approx \mathcal{O}(1/q)\epsilon_{\mathcal{A}}$. As previous works mainly focused on the cases where the underlying PKE has negligible decryption errors, Cini et al. [16] proposed a new transformation that can work for the PKE with non-negligible decryption errors. In addition, Kitagawa and Nishimaki [39] and Pan and Zeng [45] further considered other security notions of the FO-like transformations, named key dependent message (KDM) security and selective opening security (SO) against chosen-ciphertext attacks, respectively.

The compressed oracle technique is a useful tool provided by Zhandry [55]. Based on it, Don et al. [20] proposed an online extractor and provided a proof for the *textbook* FO transformation with tightness $\epsilon_R \approx \mathcal{O}(1/q^2)\epsilon_{\mathcal{A}}^2$. Using a similar method, Shan et al. [48] and Ge et al. [25] began to analyze the IND-QCCA security of the FO-like transformation. T_{CH} and T_H are proposed by Huguenin-Dumittan and Vaudenay [31], but the QROM proof for T_H is left. Jiang et al. [33] proposed and provided the ROM and QROM proofs for T_{RH} , and related the IND-1CCA security of T_{RH} to that of T_H in the QROM. However, their proofs of T_{RH} can be improved when the underlying PKE is rigid deterministic.

2 Preliminaries

2.1 Notation

We represent the function H with domain \mathcal{X} and codomain \mathcal{Y} as $H : \mathcal{X} \rightarrow \mathcal{Y}$. We denote the set of such functions as Ω_H . For any set \mathcal{S} , we use $|\mathcal{S}|$ to represent its cardinality and use $s \leftarrow \mathcal{S}$ to denote the random choice of an element s from \mathcal{S} with uniform probability. To indicate the output of a probabilistic (or deterministic) algorithm A with input x as y , we use the notation $y \leftarrow A(x)$ (or $y := A(x)$). Additionally, A^H (or A^{H^H}) denotes an oracle algorithm that has classical (or quantum) access to the oracle H . We utilize the notation $[x = y]$ to represent an integer value of 1 when $x = y$ and 0 otherwise. The security parameter is denoted by λ , and PPT stands for *probabilistic polynomial time*. The base of logarithm \log is 2, unless stated otherwise.

2.2 The (Quantum) Random Oracle Model

For the introduction to the fundamentals of quantum computation, we recommend readers refer to [42]. In brief, the *state space* of a quantum system is a complex vector space with an inner product. The Dirac notation “ $|\cdot\rangle$ ” (and “ $\langle\cdot|$ ”) is used to represent unit vectors, known as *state vectors*, in the state space (and their counterparts in the dual space). The state space can be spanned by a set of orthonormal bases called *computational bases*. The *joint state* of $|\psi\rangle$ and $|\phi\rangle$ is $|\psi\rangle \otimes |\phi\rangle$. The *norm* of a state $|\psi\rangle$, denoted as $\| |\psi\rangle \|$, is calculated as $\sqrt{\langle\psi|\psi\rangle}$, where “ $\langle\psi|\phi\rangle$ ” signifies the inner product between $|\psi\rangle$ and $|\phi\rangle$.

The random oracle model (ROM), as introduced in [7], is an idealized model where the hash function is modeled as a publicly accessible random oracle. In

this model, to get the value of $H(x)$ for a given hash function H , an adversary must make a H random oracle query on x . The quantum analog of this model, known as the quantum random oracle model (QROM) [11], permits adversaries to make the random oracle queries in a superposition state. Here, the H random oracle behaves as a unitary transformation, mapping $|x, y\rangle$ to $|x, y \oplus H(x)\rangle$. It is worth noting that traditional, or “classical”, queries are still permissible in the QROM. These can be interpreted as first querying the random oracle on $|x, 0\rangle$ and then measuring the second register to obtain the classical output [20].

2.3 The One-Way to Hiding Lemma

In the ROM, random oracles serve as a crucial tool for learning the adversary’s queries. An adversary cannot learn any knowledge about $H(x)$ without querying the H random oracle for x . Furthermore, without querying the random oracle at x , the adversary cannot discover the reprogramming of the oracle at that point. Under certain conditions in the QROM, the simulator can exploit the adversary’s behavior to identify the point of random oracle reprogramming by employing the “one-way to hiding (O2H)” lemma. In this work, we adopt the version of the O2H lemma introduced by Bindel et al. [10], which has a tight bound except for a quadratic loss that is impossible to avoid [37].

Lemma 1 (Double-Sided One-Way to Hiding [10]). *Let $G, H : \mathcal{X} \rightarrow \mathcal{Y}$ be random functions such that $\forall x \neq x^* \in \mathcal{X}, G(x) = H(x)$, and z be a random value, where (G, H, x^*, z) may have arbitrary joint distribution. Let $A^{|H\rangle}$ be an oracle algorithm that has quantum access to the H random oracle. Then there exists a double-sided oracle algorithm $B^{|G\rangle, |H\rangle}$ that can access both G and H , such that*

$$|\Pr[\text{Ev} : A^{|G\rangle}(z)] - \Pr[\text{Ev} : A^{|H\rangle}(z)]| \leq \sqrt{\Pr[\hat{x} = x^* : \hat{x} \leftarrow B^{|G\rangle, |H\rangle}(z)]}$$

for an arbitrary classical event Ev .

2.4 The Compressed Oracle Technique

The reduction in the ROM is allowed to record the adversaries’ queries, but this feature was once considered impossible in the QROM. This is due to the quantum no-cloning principle, which implies that any direct recording of a quantum state would alter the adversary’s state. Fortunately, Zhandry [55] overcomes this “recording barrier” by introducing the compressed oracle technique. The basic idea is to purify the quantum random oracle and then record adversaries’ queries on the purified quantum random oracle.

Definition 1 (Compressed Standard Oracle). *Let D represent the database composed of q pairs $(x, y) \in (\mathcal{X} \times \mathcal{Y}) \cup (\perp, 0^n)$ where $n := \log |\mathcal{Y}|$ and q signifies the maximum quantum random oracle queries a quantum adversary could make. The structure of D is as follows:*

$$D = ((x_1, y_1), (x_2, y_2), \dots, (x_l, y_l), (\perp, 0^n), \dots, (\perp, 0^n)) ,$$

where $0 \leq l \leq q$, $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$ for $i = 1, \dots, l$, $x_1 < \dots < x_l$, and D ends with $q-l$ pairs of $(\perp, 0^n)$. We denote the set of such databases as \mathcal{D} . For any $x \in \mathcal{X}$, if there exists a y such that $(x, y) \in D$, then we define $D(x) = y$; otherwise, $D(x) = \perp$. Notably, no two pairs in D share the same x . We use $|D|$ to denote the number of (x, y) pairs in D where $x \neq \perp$. When $|D| < q$ and $D(x) = \perp$, we define $D \cup (x, y)$ as the operation of removing one $(\perp, 0^n)$ entry from D and then inserting (x, y) while preserving the ascending order of x values.

Let D be a quantum register with state space $\mathcal{H} = \mathbb{C}[D]$. On the basis state $|D\rangle$ (where $D \in \mathcal{D}$), we define a unitary decompression procedure F_x as follows:

– If $D(x) = \perp$ and $|D| < q$, we have

$$F_x |D\rangle = 2^{-n/2} \sum_y |D \cup (x, y)\rangle ,$$

$$F_x \left(2^{-n/2} \sum_y |D \cup (x, y)\rangle \right) = |D\rangle ,$$

$$F_x \left(2^{-n/2} \sum_y (-1)^{z \cdot y} |D \cup (x, y)\rangle \right) = 2^{-n/2} \sum_y (-1)^{z \cdot y} |D \cup (x, y)\rangle \text{ where } z \neq 0 .$$

– If $D(x) = \perp$ but $|D| = q$, we have $F_x |D\rangle = |D\rangle$.

Let X and Y be the input and output registers of the quantum random oracle, respectively. We define a unitary operator O_x that is applied to YD as

$$O_x : |y, D\rangle \rightarrow |y \oplus D(x), D\rangle .$$

Note that unlike the definition in [55] where $y \oplus \perp = y$, here we define $0^n \oplus \perp = \perp$, $\perp \oplus 0^n = \perp$, $\perp \oplus \perp = 0^n$, and for $y \in \mathcal{Y} \setminus \{0^n\}$, $y \oplus \perp = y$, $\perp \oplus y = \perp$ ³. In the end, the compressed standard oracle applied to XYD can be defined as

$$\text{CStO} := \sum_x |x\rangle \langle x| \otimes F_x O_x F_x .$$

The compressed standard oracle is proved to be perfectly indistinguishable from the quantum random oracle by Zhandry [55].

Lemma 2 (Lemma 4 in [55]). *The compressed oracle as defined in Definition 1 with D set as $\bigotimes_{i=1}^q (\perp, 0^n)$ initially is perfectly indistinguishable from a quantum random oracle $H : \mathcal{X} \rightarrow \mathcal{Y}$ for any quantum adversary making at most q random oracle queries.*

³ With this definition, we can verify that $O_x O_x = I$, indicating that the adjoint of O_x is itself, and thus O_x is unitary.

2.5 Cryptographic Primitives

Definition 2 (Public-Key Encryption). *The public-key encryption (PKE) scheme is composed of three PPT algorithms with the security parameter λ , a message space \mathcal{M} , and a ciphertext space \mathcal{C} : (1) The key generation algorithm Gen is a probabilistic algorithm that takes as input 1^λ and outputs a public/private key pair (pk, sk) . (2) The encryption algorithm Enc is a probabilistic algorithm that takes as input pk and a message $m \in \mathcal{M}$, and outputs a ciphertext $c \in \mathcal{C}$. (3) The decryption algorithm Dec is a deterministic algorithm that takes as input sk and $c \in \mathcal{C}$, and outputs $m \in \mathcal{M}$ or a special $\perp \notin \mathcal{M}$ value.*

The correctness requirement of a PKE is that for all possible outputs (pk, sk) of $\text{Gen}(1^\lambda)$, and all possible outputs c of $\text{Enc}(\text{pk}, m)$, we have $\text{Dec}(\text{sk}, c) = m$. We say a PKE scheme is deterministic if Enc is a deterministic algorithm.

Definition 3 (δ -correctness [20]). *We say a PKE scheme is δ -correct if*

$$\mathbb{E}_{(\text{pk}, \text{sk}) \leftarrow \text{Gen}(1^\lambda)} \left[\max_{m \in \mathcal{M}} \Pr[\text{Dec}(\text{sk}, c) \neq m : c \leftarrow \text{Enc}(\text{pk}, m)] \right] \leq \delta .$$

If $\delta = 0$, then we say the PKE scheme is perfectly correct.

Definition 4 (rigidity [8]). *We say a deterministic PKE scheme is rigid if $\text{Enc}(\text{pk}, m) = c$ for every $(\text{pk}, \text{sk}) \leftarrow \text{Gen}(1^\lambda)$, every $c \in \mathcal{C}$, and $m := \text{Dec}(\text{sk}, c)$, when the PKE is correct.*

Definition 5 (The OW-CPA Security of PKE). *We define the OW-CPA security of a PKE scheme $\text{PKE} = (\text{Gen}, \text{Enc}, \text{Dec})$ in terms of an attack game between a challenger and an adversary \mathcal{A} , as follows. The challenger computes*

$$(\text{pk}, \text{sk}) \leftarrow \text{Gen}(1^\lambda), m^* \leftarrow_{\$} \mathcal{M}, c^* \leftarrow \text{Enc}(\text{pk}, m^*),$$

and sends (pk, c^) to \mathcal{A} . Finally, \mathcal{A} outputs $\hat{m} \in \mathcal{M}$. We define \mathcal{A} 's advantage with respect to PKE as $\text{Adv}_{\text{PKE}}^{\text{OW-CPA}}(\mathcal{A}) := \Pr[m^* = \hat{m}]$, and if this advantage is negligible for all PPT adversaries, we say that PKE is OW-CPA secure. We refer to the m^* and the c^* computed by the challenger as the challenge message and the challenge ciphertext, respectively.*

Definition 6 (Key Encapsulation Mechanism). *Key encapsulation mechanism (KEM) is specified by three PPT algorithms with the security parameter λ , a key space \mathcal{K} , and an encapsulation space \mathcal{C} : (1) The key generation algorithm Gen is a probabilistic algorithm that takes as input 1^λ and outputs a public/private key pair (pk, sk) . (2) The encapsulation algorithm Encaps is a probabilistic algorithm that takes as input pk and outputs a pair (k, c) where the key $k \in \mathcal{K}$ and the encapsulation $c \in \mathcal{C}$. (3) The decapsulation algorithm Decaps is a deterministic algorithm that takes as input sk and $c \in \mathcal{C}$, and outputs $k \in \mathcal{K} \cup \{\perp\}$.*

The correctness requirement of a KEM is that for all possible outputs (pk, sk) of $\text{Gen}(1^\lambda)$, and all possible outputs (k, c) of $\text{Encaps}(\text{pk})$, we have $\text{Decaps}(\text{sk}, c) = k$. We usually say that a KEM is explicit reject if $\perp \notin \mathcal{K}$, while a KEM is implicit reject if $\perp \in \mathcal{K}$ represents a random value.

Definition 7 (The IND-1CCA Security of KEM). We define the IND-1CCA security of a KEM scheme $\text{KEM} = (\text{Gen}, \text{Encaps}, \text{Decaps})$ in terms of an attack game between a challenger and an adversary \mathcal{A} , as follows. The challenger computes

$$(\text{pk}, \text{sk}) \leftarrow \text{Gen}(1^\lambda), (k_0, c^*) \leftarrow \text{Encaps}(\text{pk}), k_1 \leftarrow \mathcal{K}, b \leftarrow \{0, 1\},$$

and sends (pk, c^*, k_b) to \mathcal{A} . In this game, \mathcal{A} can make at most one decapsulation query on any $c \neq c^*$. Finally, \mathcal{A} outputs $\hat{b} \in \{0, 1\}$. We define \mathcal{A} 's advantage with respect to KEM as $\text{Adv}_{\text{KEM}}^{\text{IND-1CCA}}(\mathcal{A}) := |\Pr[b = \hat{b}] - 1/2|$, and if this advantage is negligible for all PPT adversaries, we say that KEM is IND-1CCA secure. We refer to the c^* and the k_b sent to \mathcal{A} as the challenge encapsulation and the challenge key, respectively.

Theorem 1 (Difference Lemma [12]). Let Z, W_1, W_2 be some events defined over some probability space, and \bar{Z} be the complement of Z . Assume that $W_0 \wedge \bar{Z}$ occurs if and only if $W_1 \wedge \bar{Z}$ occurs, then we have $|\Pr[W_0] - \Pr[W_1]| \leq \Pr[Z]$.

3 The Security of T_{RH} in the ROM

Here, we prove that the IND-1CCA security of $\text{KEM}_{RH} := T_{RH}[\text{PKE}', H]$ can be tightly reduced to the OW-CPA security of PKE' in the ROM, if PKE' is rigid deterministic.

Theorem 2 (The security of T_{RH} in the ROM). Assume $H : \mathcal{M} \times \mathcal{C} \rightarrow \mathcal{K}$ is modeled as a random oracle. If PKE' is a rigid deterministic public-key encryption scheme that is δ -correct and OW-CPA secure, then KEM_{RH} is IND-1CCA secure.

In particular, for any PPT adversary \mathcal{A} that attacks the IND-1CCA security of KEM_{RH} , there exists a PPT adversary \mathcal{B} that attacks the OW-CPA security of PKE' , such that

$$\text{Adv}_{\text{KEM}_{RH}}^{\text{IND-1CCA}}(\mathcal{A}) \leq 2 \left(\text{Adv}_{\text{PKE}'}^{\text{OW-CPA}}(\mathcal{B}) + \delta \right).$$

Proof (Theorem 2). Figure 2 shows the simulation of the challenger for the adversary \mathcal{A} in game G_j for $j = 0, \dots, 5$. In each game, b is a random bit chosen by the challenger, while \hat{b} is the bit output by \mathcal{A} at the end of the game. We define W_j to be the event that $\hat{b} = b$ in game G_j .

Game G_0 . In this game, the challenger explicitly initializes an empty associative array $\text{Map} : \mathcal{M} \times \mathcal{C} \rightarrow \mathcal{K}$ to implement the random oracle. In the initialization step, k_0 is chosen uniformly over \mathcal{K} and then is stored in $\text{Map}[(m^*, c^*)]$. This is equivalent to setting the value of the random oracle at (m^*, c^*) to k_0 . We can see that, except for the extra records of responses from the random oracle, the behavior of the challenger is clearly consistent with that in the IND-1CCA game of $\text{KEM}_{RH} := T_{RH}[\text{PKE}', H]$. Therefore,

$$|\Pr[W_0] - 1/2| = \text{Adv}_{\text{KEM}_{RH}}^{\text{IND-1CCA}}(\mathcal{A}). \quad (1)$$

Games G_0 to G_5 :	Decapsulation Oracle $O_{\text{Dec}}(c \neq c^*)$:
1 : $(\text{pk}, \text{sk}) \leftarrow \text{Gen}'(1^\lambda)$, $m^* \leftarrow_{\$} \mathcal{M}$	1 : if cnt = 0 then
2 : $c^* := \text{Enc}'(\text{pk}, m^*)$, $k_0 \leftarrow_{\$} \mathcal{K}$, $k_1 \leftarrow_{\$} \mathcal{K}$	2 : cnt := cnt + 1
3 : initialize an empty associative array	3 : if COLL ₂ then // $G_1 - G_5$
$\text{Map} : \mathcal{M} \times \mathcal{C} \rightarrow \mathcal{K}$	4 : return \perp // $G_1 - G_5$
4 : $\text{Map}[(m^*, c^*)] := k_0$ // $G_0 - G_1$	5 : $\text{List}^{\text{O}_{\text{Dec}}}.append(c)$ // $G_4 - G_5$
5 : $b \leftarrow_{\$} \{0, 1\}$, cnt := 0	6 : $m' := \text{Dec}'(\text{sk}, c)$ // $G_0 - G_4$
6 : flag $\leftarrow_{\$} \{0, 1\}$ // $G_3 - G_5$	7 : if $m' = \perp$ then // $G_0 - G_2$
7 : $\text{List}^{\text{O}_{\text{Dec}}} := \perp$, $k^* \leftarrow_{\$} \mathcal{K}$ // $G_4 - G_5$	8 : if flag = 0 then // $G_3 - G_5$
8 : if COLL ₁ then // $G_1 - G_5$	9 : return $k' := H(\star, c)$ // $G_0 - G_3$
9 : return \perp // $G_1 - G_5$	10 : if $(\star, c) \in \text{Domain}(\text{Map})$
10 : $\hat{b} \leftarrow_{\$} \mathcal{A}^{\text{O}_{\text{Dec}}, H}(\text{pk}, c^*, k_b)$	then // $G_4 - G_5$
11 : return $[b = \hat{b}]$	11 : return $k' := \text{Map}[(\star, c)]$
Random Oracle $H(m, c)$:	// $G_4 - G_5$
1 : if $(m, c) \notin \text{Domain}(\text{Map})$ then	12 : return $k' := H(m', c)$ // $G_0 - G_3$
2 : $\text{Map}[(m, c)] \leftarrow_{\$} \mathcal{K}$	13 : if $\exists (m', c) \in \text{Domain}(\text{Map})$
3 : if $c \in \text{List}^{\text{O}_{\text{Dec}}}$ then // $G_4 - G_5$	<i>s.t.</i> $\text{Enc}'(\text{pk}, m') = c$ then
4 : if (flag = 0 and $m = \star$)	// $G_4 - G_5$
or (flag = 1 and $\text{Enc}'(\text{pk}, m) = c$)	14 : return $k' := \text{Map}[(m', c)]$
then // $G_4 - G_5$	// $G_4 - G_5$
5 : $\text{Map}[(m, c)] := k^*$ // $G_4 - G_5$	15 : return $k' := k^*$ // $G_4 - G_5$
6 : return $\text{Map}[(m, c)]$	16 : return $k' := \perp$

Fig. 2. Games G_0 to G_5 for the proof of Theorem 2.

Game G_1 . This game is the same as game G_0 except that events COLL₁ and COLL₂ do not occur, where COLL₁ (or COLL₂) denotes that decrypting the encapsulation $c^* = \text{Enc}'(\text{pk}, m^*)$ received by \mathcal{A} (or the decapsulation oracle query $c = \text{Enc}'(\text{pk}, m')$ issued by \mathcal{A}) with Dec using sk would obtain m such that $m \neq m^*$ (or $m \neq m'$). By the δ -correctness of PKE', the probability of either COLL₁ or COLL₂ occurring is no greater than δ . Therefore,

$$|\Pr[W_1] - \Pr[W_0]| \leq 2\delta. \quad (2)$$

Game G_2 . This game is the same as game G_1 except that assigning k_0 to $\text{Map}[(m^*, c^*)]$ is removed from the initialization step. Let Z_j be the event that \mathcal{A} makes an H random oracle query on (m^*, c^*) in game G_j , then this game and game G_1 proceed identically until Z_1 or Z_2 occurs. By the Difference Lemma (Theorem 1), we have

$$|\Pr[W_2] - \Pr[W_1]| \leq \Pr[Z_2]. \quad (3)$$

Here, since k_0 and k_1 are both randomly chosen from \mathcal{K} , and are both irrelevant to the two oracles, b is independent of \mathcal{A} 's view. Therefore,

$$\Pr[W_2] = 1/2 . \quad (4)$$

Game G_3 . This game modifies the initialization step and the decapsulation oracle in game G_2 . In the initialization step, the challenger picks an extra random bit **flag**. In the decapsulation oracle, the condition $m' = \perp$ is replaced by **flag** = 0. One can note that if $m' = \perp$ when **flag** = 0, or if $m' \neq \perp$ when **flag** = 1, game G_3 is entirely identical to game G_2 ⁴, thereby

$$\Pr[Z_2] = \Pr[Z_3 \wedge m' = \perp | \text{flag} = 0] + \Pr[Z_3 \wedge m' \neq \perp | \text{flag} = 1] . \quad (5)$$

Game G_4 . Compared with game G_3 , we make the following modifications to answer the decapsulation query without using sk . Firstly, in the initialization step, the challenger initializes an extra empty list $\text{List}^{O_{\text{Dec}}}$ to store the c queried to the decapsulation oracle, and chooses a random $k^* \leftarrow \mathcal{K}$ for the decapsulation oracle query. The decapsulation oracle works as follows.

- CASE **flag** = 0: If (\star, c) has been queried in the H random oracle, then return $\text{Map}[\star, c]$; otherwise, return k^* .
- CASE **flag** = 1: If there exists $(m', c) \in \text{Domain}(\text{Map})$ where $\text{Enc}'(\text{pk}, m') = c$, then return $\text{Map}[m', c]$; otherwise, return k^* .

In the H random oracle, for the new query (m, c) , we introduce the following operations: if c has been queried to the decapsulation oracle, i.e., $c \in \text{List}^{O_{\text{Dec}}}$, then if $m = \star$ when **flag** = 0, or if $\text{Enc}'(\text{pk}, m) = c$ when **flag** = 1, we reprogram $\text{Map}[(m, c)]$ to k^* .

Recall that we have assumed COLL_2 would not occur since game G_1 , and that PKE' is rigid deterministic. This means that for any c where $\text{Dec}(\text{sk}, c) = m' \neq \perp$, m' is the only value in \mathcal{M} such that $\text{Enc}'(\text{pk}, m') = c$. Therefore, if \mathcal{A} has performed an H random oracle query on (\star, c) when **flag** = 0, or on (m', c) when **flag** = 1, before the decapsulation query of c , then the decapsulation oracle will return the corresponding random oracle value, which is consistent with the behavior in game G_3 in the same case. If \mathcal{A} does not make an H random oracle query on (\star, c) or (m', c) , then the decapsulation oracle will return k^* , but in the subsequent H random oracle query on the corresponding (\star, c) or (m', c) , it will also respond with the same k^* . Since k^* is chosen randomly, the behavior at this time is consistent with that in game G_3 . Therefore,

$$\begin{aligned} \Pr[Z_4 \wedge m' = \perp | \text{flag} = 0] &= \Pr[Z_3 \wedge m' = \perp | \text{flag} = 0] \\ \Pr[Z_4 \wedge m' \neq \perp | \text{flag} = 1] &= \Pr[Z_3 \wedge m' \neq \perp | \text{flag} = 1] . \end{aligned} \quad (6)$$

⁴ One may note that there are two additional cases in game G_3 , i.e., when $m' = \perp$ but **flag** = 1, and when $m' \neq \perp$ but **flag** = 0, requiring an extension of the domain of H to $\mathcal{M} \times \{\perp\}$ for $H(\perp, c)$ to be defined. Nevertheless, in our analysis of the relationship between G_3 and G_2 , these cases are not pertinent and, thus, are omitted for brevity.

Combining (5) and (6), we obtain

$$\begin{aligned}
\Pr[Z_4] &\geq \Pr[Z_4 \wedge m' = \perp \wedge \mathbf{flag} = 0] + \Pr[Z_4 \wedge m' \neq \perp \wedge \mathbf{flag} = 1] \\
&= \Pr[Z_4 \wedge m' = \perp | \mathbf{flag} = 0] \Pr[\mathbf{flag} = 0] \\
&\quad + \Pr[Z_4 \wedge m' \neq \perp | \mathbf{flag} = 1] \Pr[\mathbf{flag} = 1] \\
&= \frac{1}{2} (\Pr[Z_4 \wedge m' = \perp | \mathbf{flag} = 0] + \Pr[Z_4 \wedge m' \neq \perp | \mathbf{flag} = 1]) \quad (7) \\
&= \frac{1}{2} (\Pr[Z_3 \wedge m' = \perp | \mathbf{flag} = 0] + \Pr[Z_3 \wedge m' \neq \perp | \mathbf{flag} = 1]) \\
&= \frac{1}{2} \Pr[Z_2] .
\end{aligned}$$

At this point, it can be observed that the response of the decapsulation oracle in game G_4 no longer depends on $m' := \text{Dec}'(\text{sk}, c)$. Therefore, removing this step has no impact on $\Pr[Z_4]$.

Game 5. This game is the same as game G_4 , except for removing the step $m' := \text{Dec}'(\text{sk}, c)$ in the decapsulation oracle. From the above discussion, we have

$$\Pr[Z_5] = \Pr[Z_4] . \quad (8)$$

At this point, we can find that all the oracles do not depend on sk and m^* . Therefore, when the event Z_5 occurs, we can construct an adversary \mathcal{B} to attack the OW-CPA security of PKE' as follows: When \mathcal{B} received the public key pk and the challenge ciphertext c^* from the OW-CPA game of PKE' , he chooses a random $k \leftarrow \mathcal{K}$, and then sends (pk, c^*, k) to \mathcal{A} . After that, he uses the decapsulation oracle and H random oracle described in game G_5 to respond to \mathcal{A} 's queries. At the end of the game, \mathcal{B} can search the pair (m^*, c^*) in Map that satisfies $\text{Enc}'(\text{pk}, m^*) = c^*$ and output m^* . Therefore,

$$\Pr[Z_5] \leq \text{Adv}_{\text{PKE}'}^{\text{OW-CPA}}(\mathcal{B}) . \quad (9)$$

Combining (1)–(4) and (7)–(9), we obtain

$$\text{Adv}_{\text{KEM}_{RH}}^{\text{IND-1CCA}}(\mathcal{A}) \leq 2 \left(\text{Adv}_{\text{PKE}'}^{\text{OW-CPA}}(\mathcal{B}) + \delta \right) .$$

That completes the proof of the theorem. \square

Remark 1. The bound given by Jiang et al. [33] in the case of deterministic PKE is

$$\text{Adv}_{\text{KEM}_{RH}}^{\text{IND-1CCA}}(\mathcal{A}) \leq (q_H + 1) \text{Adv}_{\text{PKE}'}^{\text{OW-CPA}}(\mathcal{B}) + \delta ,$$

where q_H is the number of H random oracle queries. We prove that the reduction from IND-1CCA security of KEM_{RH} to the OW-CPA security of rigid deterministic PKE' is *tight*, with a loss factor of $\mathcal{O}(1)$.

4 The Security Analysis in the QROM

4.1 The Reprogram-After-Measure Technique

During the IND-1CCA game of KEM_{RH} in the ROM, the challenger needs to access the random oracle to calculate the response for the adversary \mathcal{A} in the decapsulation oracle. But in some cases where the challenger does not know the point at which it should query the random oracle, the challenger can directly return a random response instead of accessing the random oracle, and the only requirement is that the random response should be consistent with the response to the corresponding random oracle query made by \mathcal{A} in the subsequent process, e.g., Game 5 in the proof of Theorem 2. This process involves two techniques of ROM called *lazy sampling* and *reprogramming*, which are hard to carry over to the quantum setting as Boneh et al. [11] claim.

With the help of the compressed oracle technique introduced by Zhandry [55], we provide a new technique that can simulate the decapsulation oracle in a similar way. We will reprogram the compressed oracle after performing a measurement. Therefore, we refer to the proposed technique as *reprogram-after-measure*. Note that the decapsulation oracle query and the *implicit* random oracle query in it are both classical, and the classical decapsulation oracle is queried at most once. In Sect. 4.2, we can see that we can obtain a *tighter* security proof with this new technique.

Theorem 3 (Reprogram-after-Measure). *Let $A^{O,|H\rangle}$ be a quantum oracle algorithm that can make q_H times (quantum) H random oracle queries, but at most one (classical) O oracle query, where $O : \mathcal{C} \rightarrow \mathcal{Z}, H : \mathcal{X} \rightarrow \mathcal{Y}$. Let $\mathcal{C}^\perp \subseteq \mathcal{C}$ be a set on which A is not allowed to make the O oracle query, and for any $c \in \mathcal{C}^\perp$ the O oracle always returns \perp . For $c \in \mathcal{C} \setminus \mathcal{C}^\perp$, the O oracle computes $x := f^{-1}(c)$, (classically) accesses the H random oracle to obtain $y := H(x)$, and returns $g(y)$, where the functions $f : \mathcal{X} \rightarrow \mathcal{C}, g : \mathcal{Y} \rightarrow \mathcal{Z}$, and there is a unique preimage x for $c \in \mathcal{C} \setminus \mathcal{C}^\perp$ under f . Then there exists an algorithm B that does not need to access the O oracle and the H random oracle, and needs to know how to calculate the functions f and g (but does not need to know how to calculate f^{-1}), such that*

$$\Pr[\text{Ev} : A^{O,|H\rangle}] \leq 2 \Pr[\text{Ev} : B] \quad (10)$$

for any classical event Ev .

In particular, we can construct B from $A^{O,|H\rangle}$ as follows. Firstly, we use the compressed oracle CStO to replace the H random oracle in $A^{O,|H\rangle}$. Let XY be the input/output registers of CStO , \mathcal{D} be the database register used by CStO that is initialized to $\bigotimes_{i=1}^{q_H+1} (\perp, 0^n)$ (note that $A^{O,|H\rangle}$ queries the H random oracle $q_H + 1$ times in total) where $n := \log(|\mathcal{Y}|)$, and CZ be the input/output registers of O oracle. We define a function $e : \mathcal{C} \times \mathcal{D} \rightarrow \mathcal{Y} \cup \perp$ as follows, where \mathcal{D} is the database set as defined in Definition 1:

$$e(c, D) = \begin{cases} D(x) & \text{if there exists } (x, y) \in D \text{ such that } f(x) = c \text{ and } y \neq \perp, \\ \perp & \text{otherwise.} \end{cases}$$

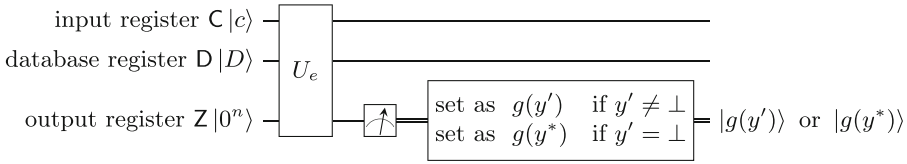


Fig. 3. The quantum circuit diagram for O^B .

Since e can be computed efficiently, the unitary operator $U_e : |c, D, y\rangle \rightarrow |c, D, y \oplus e(c, D)\rangle$ can also be implemented efficiently based on the quantum computation theory. B first chooses a random $y^* \leftarrow \mathcal{Y}$, and then runs $A^{O, |H\rangle}$ until it makes an O oracle query (with classical input c on register C). If $c \in \mathcal{C}^\perp$, B directly sets register Z to \perp and continues running $A^{O, |H\rangle}$ until the end; otherwise, instead of accessing the O oracle, B uses the following O^B oracle as a substitute, as shown in Fig. 3:

1. Initialize the register Z to 0^n .
2. Apply U_e to registers CDZ , where Z is the output register.
3. Perform the measurement M_Z on the register Z in the computational basis $\{|y\rangle\}_{y \in \mathcal{Y} \cup \perp}$, denoting the result as $|y'\rangle$.
4. If $y' = \perp$, let $y' := y^*$. Set Z to $g(y')$.

After that, define a function $u_c(x) : \mathcal{X} \rightarrow \{0, 1\}$ as follows:

$$u_c(x) = \begin{cases} 1 & \text{if } f(x) = c, \\ 0 & \text{otherwise,} \end{cases}$$

where $c \in \mathcal{C}$ is the classical input on the register C when $A^{O, |H\rangle}$ queries the O oracle. Construct a unitary operator $U_{u_c} : |x, b\rangle \rightarrow |x, b \oplus u_c(x)\rangle$. In subsequent H random oracle queries, B uses the CStO^B oracle defined as follows (as shown in Fig. 4) instead of CStO to simulate the H random oracle:

1. Initialize a register R to 0 , where R is a one qubit register.
2. Apply U_{u_c} to registers XR , where R is the output register.
3. Apply the following two conditional operations:
 - (a) The control bit is R , and apply the unitary operator $U_{y'}$ to Y if $b = 1$, where $U_{y'}|y\rangle = |y \oplus y'\rangle$ and y' is the (classical) value obtained in the O^B oracle.
 - (b) The control bit is R , and apply the unitary operator CStO to XYD if $b = 0$.
4. Apply U_{u_c} on XR , where R is the output register. Note that R is restored to $|0\rangle$, so it can be discarded.

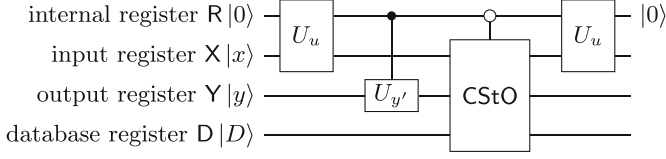


Fig. 4. The quantum circuit diagram for CStO^B , where R is an internal register used by CStO^B .

This completes the description of the construction of B .

Proof (Theorem 3). Here we use the same notation used in Theorem 3. Let $A^{O,|H\rangle}$ be the oracle algorithm defined in Theorem 3, where XY are the input/output registers of the H random oracle and CZ are the input/output registers of O oracle. We introduce a database register D that is initialized to $\bigotimes_{i=1}^{q_H+1}(\perp, 0^n)$ and use the compressed oracle CStO to implement the H random oracle in $A^{O,|H\rangle}$ to get a new oracle algorithm $\hat{A}^{O,|H\rangle}$. According to Lemma 2, we have

$$\Pr[\text{Ev} : A^{O,|H\rangle}] = \Pr[\text{Ev} : \hat{A}^{O,|H\rangle}] \quad (11)$$

for any classical event Ev .

Next, we analyze the relationship between $\hat{A}^{O,|H\rangle}$ and B , where B is defined in Theorem 3. Observe that the behavior of B is the same as that of $\hat{A}^{O,|H\rangle}$ until $\hat{A}^{O,|H\rangle}$ makes an O oracle query on $c \in \mathcal{C} \setminus \mathcal{C}^\perp$. In other words, if $\hat{A}^{O,|H\rangle}$ does not make an O oracle query, or if $\hat{A}^{O,|H\rangle}$ queries the O oracle on $c \in \mathcal{C}^\perp$, the behavior of B is exactly the same as that of $\hat{A}^{O,|H\rangle}$. In these two cases, Eq. (10) obviously holds. Therefore, in what follows, we only consider the case where $\hat{A}^{O,|H\rangle}$ makes only one (classical) O oracle query on $c \in \mathcal{C} \setminus \mathcal{C}^\perp$.

Consider that the H random oracle is invoked $q_H + 1$ times, where q_H times are direct quantum queries made by $\hat{A}^{O,|H\rangle}$, and 1 time is a classical query made through the O oracle. Without loss of generality, let the classical query be the i^* -th H random oracle query ($1 \leq i^* \leq q_H + 1$), and the execution of $\hat{A}^{O,|H\rangle}$ can be described as

$$U_{q_H+2} \left(\prod_{i=i^*+1}^{q_H+1} \text{CStO} \circ U_i \right) O \circ U_{i^*} \left(\prod_{i=1}^{i^*-1} \text{CStO} \circ U_i \right) |\psi_0\rangle,$$

where $|\psi_0\rangle$ is the initial state of $\hat{A}^{O,|H\rangle}$, and for $i = 1, \dots, q_H + 2$, U_i is a unitary operator⁵. Recall that the i^* -th (classical) H random oracle query is made through the O random oracle. The (non-unitary) O can be described by the following steps, where $\mathcal{C}^\perp \subseteq \mathcal{C}$ represents the set of c on which $\hat{A}^{O,|H\rangle}$ is not allowed to make the O oracle query:

⁵ This follows from the fact that any quantum oracle algorithm can be transformed to a *unitary* quantum oracle with constant factor computational overhead and the same number of oracle queries [3, 25].

1. If $c \in \mathcal{C}^\perp$, set Z to \perp ; otherwise
2. Initialize a register X' to $x := f^{-1}(c)$.
3. Initialize the register Z to 0^n , and apply CStO to registers $X'ZD$.
4. Perform the measurement M_Z on the register Z in the computational basis $\{|y\rangle\}_{y \in \mathcal{Y} \cup \perp}$, denoting the result as $|y'\rangle$.
5. Compute $g(y')$ on the register Z .

The quantum circuit diagram for steps 2–5 is shown in Fig. 5.

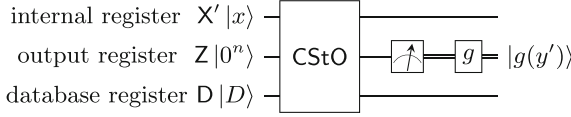


Fig. 5. The quantum circuit diagram for steps 2–5 for O , where X' is an internal register used by O .

Correspondingly, the execution of $B^{|H\rangle}$ can be described as

$$U_{q_H+2} \left(\prod_{i=i^*+1}^{q_H+1} \text{CStO}^B \circ U_i \right) O^B \circ U_{i^*} \left(\prod_{i=1}^{i^*-1} \text{CStO} \circ U_i \right) |\psi_0\rangle,$$

where CStO^B and O^B are defined in Theorem 3.

Since before querying O oracle or O^B oracle, the execution of $\hat{A}^{O,|H\rangle}$ and B are the same, they are in the same state at this time, denoted as $|\Psi\rangle$. Next, we consider the state $|\Psi\rangle$ on the register CZDP, where CZ are the input/output registers of O (or O^B) oracle, D is the database register used by CStO , and P contains all remaining registers of $\hat{A}^{O,|H\rangle}$ (or B).

Next, we divide $|\Psi\rangle$ into three mutually orthogonal parts (note that $c \notin \mathcal{C}^\perp$ and it is a certain classical value):

$$|\Psi\rangle = |\Psi_1\rangle + |\Psi_2\rangle + |\Psi_3\rangle,$$

where

$$\begin{aligned} |\Psi_1\rangle &= \sum_{\substack{z=0^n, D, p, |D| < q_H+1, \\ x=f^{-1}(c), D(x)=\perp}} \beta_{z, D, p} |c, z, D, p\rangle \\ |\Psi_2\rangle &= \sum_{\substack{z=0^n, D, p, |D| < q_H, \\ x=f^{-1}(c), D(x)=\perp, r \in \mathcal{Y}, r \neq 0}} \frac{\beta_{z, D, p, r}}{\sqrt{2^n}} \sum_{y_1 \in \mathcal{Y}} (-1)^{r \cdot y_1} |c, z, D \cup (x, y_1), p\rangle \\ |\Psi_3\rangle &= \sum_{\substack{z=0^n, D, p, |D| < q_H, \\ x=f^{-1}(c), D(x)=\perp, r \in \mathcal{Y}}} \frac{\beta_{z, D, p, 0}}{\sqrt{2^n}} \sum_{y_1 \in \mathcal{Y}} |c, z, D \cup (x, y_1), p\rangle. \end{aligned}$$

Recall that the database register D is an internal register of CStO . Thus before querying the O (or O^B) oracle, except for CStO , $\hat{A}^{O,|H\rangle}$ and B did not perform any operation on D . According to [55], $|\Psi\rangle$ does not have the component $|\Psi_3\rangle$. Hence, $|\Psi\rangle$ can be rewritten as

$$|\Psi\rangle = |\Psi_1\rangle + |\Psi_2\rangle .$$

Denote the operation of O before performing the measurement M_Z as O_1 , and the operation of O^B before performing the measurement M_Z as O_2 , then

$$\begin{aligned} O_1|\Psi_1\rangle &= \sum_{\substack{z=0,D,p,|D|<q_H+1, \\ x=f^{-1}(c),D(x)=\perp}} \frac{\beta_{z,D,p}}{\sqrt{2^n}} F_x \left(\sum_{y_1 \in \mathcal{Y}} |c, y_1, D \cup (x, y_1), p\rangle \right) \\ O_2|\Psi_1\rangle &= \sum_{\substack{z=0,D,p,|D|<q_H+1, \\ x=f^{-1}(c),D(x)=\perp}} \beta_{z,D,p} |c, \perp, D, p\rangle \\ O_1|\Psi_2\rangle &= \sum_{\substack{z=0,D,p,|D|<q_H, \\ x=f^{-1}(c),D(x)=\perp, r \in \mathcal{Y}, r \neq 0}} \frac{\beta_{z,D,p,r}}{\sqrt{2^n}} F_x \left(\sum_{y_1 \in \mathcal{Y}} (-1)^{r \cdot y_1} |c, y_1, D \cup (x, y_1), p\rangle \right) \\ O_2|\Psi_2\rangle &= \sum_{\substack{z=0,D,p,|D|<q_H, \\ x=f^{-1}(c),D(x)=\perp, r \in \mathcal{Y}, r \neq 0}} \frac{\beta_{z,D,p,r}}{\sqrt{2^n}} \sum_{y_1 \in \mathcal{Y}} (-1)^{r \cdot y_1} |c, y_1, D \cup (x, y_1), p\rangle , \end{aligned}$$

where F_x is the decompression procedure in CStO applying on register D .

Therefore, after $\hat{A}^{O,|H\rangle}$ executes O_1 and performs the measurement M_Z , for any $y' \in \mathcal{Y}$, $|\Psi\rangle$ will collapse into the (un-normalized) state

$$\begin{aligned} |\Psi_{y'}^{\hat{A}}\rangle &= \sum_{\substack{z=0,D,p,|D|<q_H+1, \\ x=f^{-1}(c),D(x)=\perp}} \frac{\beta_{z,D,p}}{\sqrt{2^n}} F_x (|c, y', D \cup (x, y'), p\rangle) \\ &+ \sum_{\substack{z=0,D,p,|D|<q_H, \\ x=f^{-1}(c),D(x)=\perp, r \in \mathcal{Y}, r \neq 0}} \frac{\beta_{z,D,p,r}}{\sqrt{2^n}} F_x \left((-1)^{r \cdot y'} |c, y', D \cup (x, y'), p\rangle \right) \end{aligned}$$

with probability $p_{y'}^{\hat{A}} = \|\Psi_{y'}^{\hat{A}}\|^2$. It implies that for any $y' \in \mathcal{Y}$, the O oracle will respond with $g(y')$ with probability $p_{y'}^{\hat{A}}$.

For B , after executing O_2 and measuring M_Z , for any $y' \in \mathcal{Y}$, $|\Psi\rangle$ will collapse into the (un-normalized) state

$$|\Psi_{y'}^B\rangle = \sum_{\substack{z=0,D,p,|D|<q_H, \\ x=f^{-1}(c),D(x)=\perp, r \in \mathcal{Y}, r \neq 0}} \frac{\beta_{z,D,p,r}}{\sqrt{2^n}} (-1)^{r \cdot y'} |c, y', D \cup (x, y'), p\rangle$$

with the *same* probability⁶

$$p_1^B = \|\Psi_{y'}^B\|^2 ,$$

⁶ Since the probability $\|\Psi_{y'}^B\|^2$ has *same* value for any $y' \in \mathcal{Y}$, we denote this common value as p_1^B .

and will collapse into the (un-normalized) state

$$|\Psi_{\perp}^B\rangle = \sum_{\substack{z=0,D,p,|D|<q_H+1, \\ x=f^{-1}(c),D(x)=\perp}} \beta_{z,D,p}|c, \perp, D, p\rangle$$

with the probability

$$p_2^B = \|\Psi_{\perp}^B\|^2 .$$

Note that when Z is \perp , the result is set to $g(y^*)$, where $y^* \in \mathcal{Y}$ is uniformly and randomly chosen by B in the beginning, so for any $y' \in \mathcal{Y}$, $\Pr[y^* = y'] = 2^{-n}$. It implies that for any y' , the probability of O^B returning $g(y')$ is

$$p^B = p_1^B + p_2^B/2^n .$$

Note that after the O oracle query, all the responses of H random oracle query on $x = f^{-1}(c)$ made by $\hat{A}^{O,|H\rangle}$ are

$$\begin{aligned} \text{CStOF}_x|x, y, D \cup (x, y')\rangle &= F_x O_x F_x F_x |x, y, D \cup (x, y')\rangle \\ &= F_x |x, y \oplus y', D \cup (x, y')\rangle , \end{aligned}$$

which is equivalent to applying a unitary operator $U_{y'}$ to $|y\rangle$ such that $U_{y'}|y\rangle = |y \oplus y'\rangle$. Therefore, the H random oracle used by $\hat{A}^{O,|H\rangle}$ after the O oracle query is equivalent to being implemented by CStO^B defined in Theorem 3. Therefore, the execution of $\hat{A}^{O,|H\rangle}$ can be rewritten as

$$U_{q_H+2} \left(\prod_{i=i^*+1}^{q_H+1} \text{CStO}^B \circ U_i \right) O \circ U_{i^*} \left(\prod_{i=1}^{i^*-1} \text{CStO} \circ U_i \right) |\psi_0\rangle .$$

According to [10], when the event Ev is classical and well-defined, the probability of occurrence of the event is equivalent to the measurement of the density operator of the final state of $\hat{A}^{O,|H\rangle}$ or B with M_{Ev} . Recall that the state of $\hat{A}^{O,|H\rangle}$ after the O oracle query is

$$\begin{aligned} |\Psi_{g(y')}^{\hat{A}}\rangle &= \frac{1}{\sqrt{p_{y'}^{\hat{A}}}} \left(\sum_{\substack{z=0,D,p,|D|<q_H+1, \\ x=f^{-1}(c),D(x)=\perp}} \frac{\beta_{z,D,p}}{\sqrt{2^n}} F_x (|c, g(y'), D \cup (x, y'), p\rangle) \right. \\ &\quad \left. + \sum_{\substack{z=0,D,p,|D|<q_H, \\ x=f^{-1}(c),D(x)=\perp, r \in \mathcal{Y}, r \neq 0}} \frac{\beta_{z,D,p,r}}{\sqrt{2^n}} F_x \left((-1)^{r \cdot y'} |c, g(y'), D \cup (x, y'), p\rangle \right) \right) \end{aligned}$$

with probability $p_{y'}^{\hat{A}}$, and the state of B after the O^B oracle query is

$$|\Psi_{g(y'),1}^B\rangle = \frac{1}{\sqrt{p_1^B}} \sum_{\substack{z=0,D,p,|D|<q_H, \\ x=f^{-1}(c),D(x)=\perp, r \in \mathcal{Y}, r \neq 0}} \frac{\beta_{z,D,p,r}}{\sqrt{2^n}} (-1)^{r \cdot y'} |c, g(y'), D \cup (x, y'), p\rangle$$

with probability p_1^β , or is

$$|\Psi_{g(y'),2}^B\rangle = \frac{1}{\sqrt{p_2^B}} \sum_{\substack{z=0,D,p,|D|<q_H+1, \\ x=f^{-1}(c),D(x)=\perp}} \beta_{z,D,p}|c, g(y'), D, p\rangle$$

with probability $p_2^B/2^n$. Thus, let Q denote $M_{\text{Ev}}U_{q_H+2} \left(\prod_{i=i^*+1}^{q_H+1} \text{CStO}^B \circ U_i \right)$, then we have

$$\begin{aligned} \Pr[\text{Ev} : \hat{A}^{O,|H}\rangle] &= \sum_{y'} p_{y'}^{\hat{A}} \|Q|\Psi_{g(y')}^{\hat{A}}\rangle\|^2 \\ \Pr[\text{Ev} : B] &= \sum_{y'} \left(\frac{p_2^B}{2^n} \|Q|\Psi_{g(y'),2}^B\rangle\|^2 + p_1^B \|Q|\Psi_{g(y'),1}^B\rangle\|^2 \right). \end{aligned}$$

Let

$$\begin{aligned} |\Phi_{y'}^1\rangle &= \sum_{\substack{z=0,D,p,|D|<q_H+1, \\ x=f^{-1}(c),D(x)=\perp}} \beta_{z,D,p} F_x(|c, g(y'), D \cup (x, y'), p\rangle) \\ |\Phi_{y'}^2\rangle &= \sum_{\substack{z=0,D,p,|D|<q_H, \\ x=f^{-1}(c),D(x)=\perp, r \in \mathcal{Y}, r \neq 0}} \frac{\beta_{z,D,p,r}}{\sqrt{2^n}} F_x\left((-1)^{r \cdot y'} |c, g(y'), D \cup (x, y'), p\rangle\right) \\ |\Phi_{y'}^3\rangle &= \sum_{\substack{z=0,D,p,|D|<q_H+1, \\ x=f^{-1}(c),D(x)=\perp}} \beta_{z,D,p}(|c, g(y'), D \cup (x, y'), p\rangle) \\ |\Phi_{y'}^4\rangle &= \sum_{\substack{z=0,D,p,|D|<q_H, \\ x=f^{-1}(c),D(x)=\perp, r \in \mathcal{Y}, r \neq 0}} \frac{\beta_{z,D,p,r}}{\sqrt{2^n}} \left((-1)^{r \cdot y'} |c, g(y'), D \cup (x, y'), p\rangle\right), \end{aligned}$$

then for any $y' \in \mathcal{Y}$, we have

$$\begin{aligned} \sqrt{p_{y'}^{\hat{A}}} \|Q|\Psi_{g(y')}^{\hat{A}}\rangle\| &= \sqrt{p_{y'}^{\hat{A}}} \|Q \frac{1}{\sqrt{p_{y'}^{\hat{A}}}} \left(2^{-n/2} |\Phi_{y'}^1\rangle + |\Phi_{y'}^2\rangle\right)\| \\ &= \|Q \left(2^{-n/2} |\Phi_{y'}^1\rangle + |\Phi_{y'}^2\rangle\right)\| \\ &\leq \|Q 2^{-n/2} |\Phi_{y'}^1\rangle\| + \|Q |\Phi_{y'}^2\rangle\| \\ &\stackrel{(*)}{=} 2^{-n/2} \|Q |\Phi_{y'}^3\rangle\| + \|Q |\Phi_{y'}^4\rangle\| \\ &= \sqrt{\frac{p_2^B}{2^n}} \|Q \frac{1}{\sqrt{p_2^B}} |\Phi_{y'}^3\rangle\| + \sqrt{p_1^B} \|Q \frac{1}{\sqrt{p_1^B}} |\Phi_{y'}^4\rangle\| \\ &= \sqrt{\frac{p_2^B}{2^n}} \|Q |\Psi_{g(y'),2}^B\rangle\| + \sqrt{p_1^B} \|Q |\Psi_{g(y'),1}^B\rangle\|, \end{aligned}$$

where equation (*) utilizes the fact that unitary operators preserve the norm, and that the compression procedure F_x is unitary. Thus,

$$\begin{aligned}
\Pr[\text{Ev} : \hat{A}^{O, |H\rangle}] &= \sum_{y'} p_{y'}^{\hat{A}} \|Q|\Psi_{g(y')}^{\hat{A}}\rangle\|^2 \\
&\leq \sum_{y'} \left(\sqrt{\frac{p_2^B}{2^n}} \|Q|\Psi_{g(y'),2}^B\rangle\| + \sqrt{p_1^B} \|Q|\Psi_{g(y'),1}^B\rangle\| \right)^2 \\
&\stackrel{(*)}{\leq} 2 \sum_{y'} \left(\frac{p_2^B}{2^n} \|Q|\Psi_{g(y'),2}^B\rangle\|^2 + p_1^B \|Q|\Psi_{g(y'),1}^B\rangle\|^2 \right) \\
&= 2 \Pr[\text{Ev} : B] ,
\end{aligned} \tag{12}$$

where (*) uses the Jensen's inequality. Combining Eqs. (11) and (12), yields (10). This completes the proof of Theorem 3. \square

4.2 The Security of T_{RH} in the QROM

The security of T_{RH} in the QROM is captured in the following theorem.

Theorem 4 (The security of T_{RH} in the QROM). *Assume $H : \mathcal{M} \times \mathcal{C} \rightarrow \mathcal{K}$ is modeled as a quantum-accessible random oracle. If PKE' is a rigid deterministic public-key encryption scheme that is δ -correct and OW-CPA secure, then KEM_{RH} is IND-1CCA secure.*

In particular, for any PPT adversary \mathcal{A} that attacks the IND-1CCA security of KEM_{RH} and has quantum access to the H random oracle, there exists a PPT adversary \mathcal{B} that attacks the OW-CPA security of PKE' , such that

$$\text{Adv}_{\text{KEM}_{RH}}^{\text{IND-1CCA}}(\mathcal{A}) \leq 4\sqrt{\text{Adv}_{\text{PKE}'}^{\text{OW-CPA}}(\mathcal{B})} + 2\delta .$$

Proof (Theorem 4). For $j = 0, \dots, 3$, we define G_j to be the game played between the adversary \mathcal{A} and the challenger as shown in Fig. 6, where \mathcal{A} can make any number of quantum H random oracle queries, but at most one classical decapsulation oracle query. In each game, b is a random bit chosen by the challenger, while \hat{b} is the bit output by \mathcal{A} at the end of the game. We define W_j to be the event that $\hat{b} = b$ in game G_j .

Game G_0 . In this game, the challenger randomly chooses a function H from the set Ω_H of functions $H : \mathcal{M} \times \mathcal{C} \rightarrow \mathcal{K}$ to respond to the H random oracle queries made by \mathcal{A} . Note that although the challenger chooses a random $k^* \leftarrow \mathcal{K}$ in the initialization step, it is not used in subsequent processes. Therefore, the behavior of the challenger is exactly consistent with that in the IND-1CCA game of $\text{KEM}_{RH} := T_{RH}[\text{PKE}', H]$. Thus, we have

$$|\Pr[W_0] - 1/2| = \text{Adv}_{\text{KEM}_{RH}}^{\text{IND-1CCA}}(\mathcal{A}) . \tag{13}$$

Games G_0 to G_3 :	Decapsulation Oracle $O_{\text{Dec}}(c \neq c^*)$:
<pre> 1 : (pk, sk) ← Gen'(1^λ), m* ←$\\$ \mathcal{M} 2 : c* := Enc'(pk, m*), H ←$\\$ Ω_H 3 : k*, k₁ ←$\\$ \mathcal{K} 4 : k₀ := H(m*, c*) // G₀ - G₁ 5 : k₀ := k* // G₂ - G₃ 6 : b ←$\\$ {0, 1}, cnt := 0 7 : if COLL₁ then // G₁ - G₃ 8 : return ⊥ // G₁ - G₃ 9 : b̂ ←$\\$ $\mathcal{A}^{O_{\text{Dec}} \circ H}(\text{pk}, c^*, k_b)$ 10: return [b = b̂] </pre>	<pre> 1 : if cnt = 0 then 2 : cnt := cnt + 1 3 : if COLL₂ then // G₁ - G₃ 4 : return ⊥ // G₁ - G₃ 5 : m' := Dec'(sk, c) 6 : if m' = ⊥ then 7 : return k' := H(★, c) 8 : return k' := H(m', c) 9 : return k' := ⊥ </pre>
<hr/> Random Oracle $H(m, c)$:	
<pre> 1 : if (m, c) = (m*, c*) then // G₃ 2 : return k* // G₃ 3 : return H(m, c) </pre>	

Fig. 6. Games G_0 to G_3 for the proof of Theorem 4.

Game G_1 . In this game, similar to game G_1 described in the proof of Theorem 2, let COLL_1 (or COLL_2) represent the event of a *collision* occurring in the challenge encapsulation c^* received by \mathcal{A} (or the decapsulation oracle query c issued by \mathcal{A}). We assume that neither COLL_1 nor COLL_2 occurs in this game. The probability of either occurring is no greater than δ since PKE' is δ -correct. Thus, we have

$$|\Pr[W_1] - \Pr[W_0]| \leq 2\delta . \quad (14)$$

Game G_2 . This game is the same as game G_1 , except that $k_0 := H(m^*, c^*)$ in the initialization step is replaced by $k_0 := k^*$. Since k_0 and k_1 are both randomly chosen from \mathcal{K}^* , and are not used in any oracles, b is independent of \mathcal{A} 's view. Therefore,

$$\Pr[W_2] = 1/2 . \quad (15)$$

Game G_3 . This game modifies the H random oracle as follows: upon receiving a query where $(m, c) = (m^*, c^*)$, it returns k^* . At this point, the H random oracle is simulated by a new function $\mathbf{G} : \mathcal{M} \times \mathcal{C} \rightarrow \mathcal{K}$: for all $(m, c) \neq (m^*, c^*)$, $\mathbf{G}(m, c) = H(m, c)$; but when $(m, c) = (m^*, c^*)$, $\mathbf{G}(m^*, c^*) = k^*$ is random and independent of $H(m^*, c^*)$. Since $k_0 = k^* = \mathbf{G}(m^*, c^*)$, the behavior of the challenger is equivalent to that in game G_1 . Thus, we have

$$\Pr[W_3] = \Pr[W_1] . \quad (16)$$

Recall that $G(m, c) = H(m, c)$ for all $(m, c) \neq (m^*, c^*)$ and \mathcal{A} is not allowed to make decapsulation oracle queries on c^* . Therefore, the decapsulation oracle O_{Dec} in game G_2 is identical to that in game G_3 . Next, we can construct two oracle algorithms $A^{O_{\text{Dec}}, |H|}(z)$ and $A^{O_{\text{Dec}}, |G|}(z)$ to execute games G_2 and G_3 respectively, where $z = (\text{pk}, c^*, k_0)$, $(\text{pk}, \text{sk}) \leftarrow \text{Gen}'(\lambda)$, $m^* \leftarrow \mathcal{M}$, $c^* := \text{Enc}'(\text{pk}, m^*)$, $k^* \leftarrow \mathcal{K}$, $k_0 := k^*$. $A^{O_{\text{Dec}}, |H|}(z)$ (or $A^{O_{\text{Dec}}, |G|}(z)$) first computes $k_1 \leftarrow \mathcal{K}$, $b \leftarrow \{0, 1\}$, then runs the adversary $\mathcal{A}^{O_{\text{Dec}}, |H|}(\text{pk}, c^*, k_b)$ in game G_2 (or game G_3) to obtain \hat{b} , and finally outputs $[b = \hat{b}]$, where H (or G) is used to simulate the H random oracle in game G_2 (or game G_3). Note that we still assume that the events COLL_1 and COLL_2 defined in game G_1 do not occur. Therefore,

$$\begin{aligned} \Pr[1 \leftarrow A^{O_{\text{Dec}}, |H|}(z)] &= \Pr[W_2] \\ \Pr[1 \leftarrow A^{O_{\text{Dec}}, |G|}(z)] &= \Pr[W_3], \end{aligned} \quad (17)$$

where $z = (\text{pk}, c^*, k_0)$, $(\text{pk}, \text{sk}) \leftarrow \text{Gen}'(\lambda)$, $m^* \leftarrow \mathcal{M}$, $c^* := \text{Enc}'(\text{pk}, m^*)$, $k^* \leftarrow \mathcal{K}$, $k_0 := k^*$. Since H and G only differ at the point (m^*, c^*) , according to Lemma 1, there exists an oracle algorithm $B^{O_{\text{Dec}}, |H|, |G|}(z)$ such that⁷

$$\begin{aligned} |\Pr[1 \leftarrow A^{O_{\text{Dec}}, |G|}(z)] - \Pr[1 \leftarrow A^{O_{\text{Dec}}, |H|}(z)]| \\ \leq 2\sqrt{\Pr[(m^*, c^*) \leftarrow B^{O_{\text{Dec}}, |G|, |H|}(z)]}. \end{aligned} \quad (18)$$

Then for $j = 4, \dots, 8$, we define G_j played between the oracle algorithm and the challenger as shown in Fig. 7.

In each game, m^* is randomly chosen from \mathcal{M} , while \hat{m} is output by the oracle algorithm at the end of the game. We define the event that $\hat{m} = m^*$ as Z_j in game G_j .

Game G_4 . This game is defined in Fig. 7. It is obvious that

$$\Pr[(m^*, c^*) \leftarrow B^{O_{\text{Dec}}, |G|, |H|}(z)] \leq \Pr[Z_4], \quad (19)$$

where $z = (\text{pk}, c^*, k_0)$, $(\text{pk}, \text{sk}) \leftarrow \text{Gen}'(\lambda)$, $m^* \leftarrow \mathcal{M}$, $c^* := \text{Enc}'(\text{pk}, m^*)$, $k^* \leftarrow \mathcal{K}$, $k_0 := k^*$.

Game G_5 . This game is the same as game G_4 , except that the challenger chooses an extra random bit $\text{flag} \leftarrow \{0, 1\}$ in the initialization step, and replaces the condition $m' = \perp$ by $\text{flag} = 0$ in the decapsulation oracle. Similar to the analysis of game G_3 in the proof of Theorem 2, we have

$$\Pr[Z_4] = \Pr[Z_5 \wedge m' = \perp | \text{flag} = 0] + \Pr[Z_5 \wedge m' \neq \perp | \text{flag} = 1]. \quad (20)$$

Game G_6 . This game replaces the condition $(m, c) = (m^*, c^*)$ by $c = c^* \wedge \text{Enc}'(\text{pk}, m) = c^*$ in the G random oracle of game G_5 . Recall that since game

⁷ Since the decapsulation oracle O_{Dec} in game G_2 is identical to that in game G_3 , therefore it can be seen as an internal oracle of the oracle algorithm A .

Games G_4 to G_8 :	Decapsulation Oracle $O_{\text{Dec}}(c \neq c^*)$:
1 : $(\text{pk}, \text{sk}) \leftarrow \text{Gen}'(1^\lambda), m^* \leftarrow_{\$} \mathcal{M}$	1 : if cnt = 0 then
2 : $c^* := \text{Enc}'(\text{pk}, m^*), H \leftarrow_{\$} \Omega_H$	2 : cnt := cnt + 1
3 : $k^*, k_1 \leftarrow_{\$} \mathcal{K}, k_0 := k^*$	3 : if COLL ₂ then
4 : $b \leftarrow_{\$} \{0, 1\}, \text{cnt} := 0$	4 : return \perp
5 : if COLL ₁ then	5 : $m' := \text{Dec}'(\text{sk}, c)$
6 : return \perp	6 : if $m' = \perp$ then // G_4
7 : flag $\leftarrow_{\$} \{0, 1\}$ // $G_5 - G_8$	7 : if flag = 0 then // $G_5 - G_8$
8 : $(\hat{m}, \hat{c}) \leftarrow B^{O_{\text{Dec}}, H , G }(\text{pk}, c^*, k_0)$	8 : return $k' := H(\star, c)$
// $G_4 - G_6$	9 : return $k' := H(m', c)$
9 : $(\hat{m}, \hat{c}) \leftarrow \bar{B}^{O_{\text{Dec}}, H }(\text{pk}, c^*, k_0)$ // G_7	10 : return $k' := \perp$
10 : $(\hat{m}, \hat{c}) \leftarrow \hat{B}(\text{pk}, c^*, k_0, \text{flag})$ // G_8	Random Oracle $G(m, c)$: // $G_4 - G_6$
11 : return $[m^* = \hat{m}]$	1 : if $(m, c) = (m^*, c^*)$ then // $G_4 - G_5$
Random Oracle $H(m, c)$:	2 : if $c = c^* \wedge \text{Enc}'(\text{pk}, m) = c^*$ then // G_6
1 : return $H(m, c)$	3 : return k^*
	4 : return $H(m, c)$

Fig. 7. Games G_4 to G_8 for the proof of Theorem 4.

G_1 we have assumed that the event COLL₁ would not occur, which implies that these two conditions are equivalent. Therefore,

$$\begin{aligned} \Pr[Z_6 \wedge m' = \perp | \text{flag} = 0] &= \Pr[Z_5 \wedge m' = \perp | \text{flag} = 0] \\ \Pr[Z_6 \wedge m' \neq \perp | \text{flag} = 1] &= \Pr[Z_5 \wedge m' \neq \perp | \text{flag} = 1]. \end{aligned} \quad (21)$$

Note that at this point, the G random oracle does not depend on the knowledge of m^* , so it can be simulated with only access to the H oracle. Therefore, we can construct a new oracle algorithm $\bar{B}^{O_{\text{Dec}}, |H|}(\text{pk}, c^*, k_b)$, which is the same as $B^{O_{\text{Dec}}, |H|, |G|}$ except that if it needs to query the G oracle, it accesses H as the same way as the G random oracle in game G_6 and responds with the corresponding result.

Game G_7 . This game replaces the oracle algorithm $B^{O_{\text{Dec}}, |H|, |G|}$ by $\bar{B}^{O_{\text{Dec}}, |H|}$ in game G_6 . By the above analysis, we have

$$\begin{aligned} \Pr[Z_7 \wedge m' = \perp | \text{flag} = 0] &= \Pr[Z_6 \wedge m' = \perp | \text{flag} = 0] \\ \Pr[Z_7 \wedge m' \neq \perp | \text{flag} = 1] &= \Pr[Z_6 \wedge m' \neq \perp | \text{flag} = 1]. \end{aligned} \quad (22)$$

Denote two functions $f : \mathcal{M} \times \mathcal{C} \rightarrow \mathcal{C} \cup \perp$ and $g : \mathcal{K} \rightarrow \mathcal{K}$ as follows. If flag = 0, the challenger sets

$$f(m, c) := \begin{cases} c & \text{if } m = \star \\ \perp & \text{otherwise;} \end{cases}$$

while if $\text{flag} = 1$, the challenger sets

$$f(m, c) := \begin{cases} c & \text{if } \text{Enc}'(\text{pk}, m) = c \\ \perp & \text{otherwise} . \end{cases}$$

Let g be an identity function, i.e., $g(k) = k$ for all $k \in \mathcal{K}$. It is obvious that for $c \neq c^*$, if $m' = \perp$ when $\text{flag} = 0$, or if $m' \neq \perp$ when $\text{flag} = 1$, the process of O_{Dec} is equivalent to computing $(m, c) := f^{-1}(c)$, (classically) accessing the \mathbb{H} random oracle to obtain $k := \mathbb{H}(m, c)$, and returning $g(k)$, where (m, c) is the unique preimage of c under f . Then by Theorem 3, there exists a new algorithm \hat{B} that only needs to know how to calculate f and g , such that

$$\begin{aligned} 2 \Pr[\text{Ev} : \hat{B}(z, \text{flag}) | m' = \perp \wedge \text{flag} = 0] &\geq \Pr[\text{Ev} : \bar{B}^{O_{\text{Dec}}, |\mathbb{H}|}(z) | m' = \perp \wedge \text{flag} = 0] \\ 2 \Pr[\text{Ev} : \hat{B}(z, \text{flag}) | m' \neq \perp \wedge \text{flag} = 1] &\geq \Pr[\text{Ev} : \bar{B}^{O_{\text{Dec}}, |\mathbb{H}|}(z) | m' \neq \perp \wedge \text{flag} = 1], \end{aligned}$$

for any classical event Ev , where $z = (\text{pk}, c^*, k_0)$ ⁸.

Game G_8 . This game replaces the oracle algorithm $\bar{B}^{O_{\text{Dec}}, |\mathbb{H}|}$ by \hat{B} in game G_7 . By the above analysis, we have

$$\begin{aligned} 2 \Pr[Z_8 | m' = \perp \wedge \text{flag} = 0] &\geq \Pr[Z_7 | m' = \perp \wedge \text{flag} = 0] \\ 2 \Pr[Z_8 | m' \neq \perp \wedge \text{flag} = 1] &\geq \Pr[Z_7 | m' \neq \perp \wedge \text{flag} = 1] . \end{aligned}$$

Since the event $m' = \perp$ is independent of the event $\text{flag} = 0$, we have

$$\begin{aligned} \Pr[\text{Ev} | m' = \perp \wedge \text{flag} = 0] &= \frac{\Pr[\text{Ev} \wedge m' = \perp \wedge \text{flag} = 0]}{\Pr[m' = \perp \wedge \text{flag} = 0]} \\ &= \frac{\Pr[\text{Ev} \wedge m' = \perp \wedge \text{flag} = 0]}{\Pr[m' = \perp] \Pr[\text{flag} = 0]} = \frac{\Pr[\text{Ev} \wedge m' = \perp | \text{flag} = 0]}{\Pr[m' = \perp]} \end{aligned}$$

for any classic event Ev . Therefore, we obtain

$$\begin{aligned} 2 \Pr[Z_8 \wedge m' = \perp | \text{flag} = 0] &= 2 \Pr[Z_8 | m' = \perp \wedge \text{flag} = 0] \Pr[m' = \perp] \\ &\geq \Pr[Z_7 | m' = \perp \wedge \text{flag} = 0] \Pr[m' = \perp] \quad (23) \\ &= \Pr[Z_7 \wedge m' = \perp | \text{flag} = 0] . \end{aligned}$$

Similarly, we can obtain

$$2 \Pr[Z_8 \wedge m' \neq \perp | \text{flag} = 1] \geq \Pr[Z_7 \wedge m' \neq \perp | \text{flag} = 1] . \quad (24)$$

⁸ The extra input flag for \hat{B} is used to determine which f should be used.

Combining (20)–(24), we obtain

$$\begin{aligned}
\Pr[Z_8] &\geq \Pr[Z_8 \wedge m' = \perp \wedge \text{flag} = 0] + \Pr[Z_8 \wedge m' \neq \perp \wedge \text{flag} = 1] \\
&= \frac{1}{2} (\Pr[Z_8 \wedge m' = \perp | \text{flag} = 0] + \Pr[Z_8 \wedge m' \neq \perp | \text{flag} = 1]) \\
&\geq \frac{1}{4} (\Pr[Z_7 \wedge m' = \perp | \text{flag} = 0] + \Pr[Z_7 \wedge m' \neq \perp | \text{flag} = 1]) \\
&= \frac{1}{4} (\Pr[Z_6 \wedge m' = \perp | \text{flag} = 0] + \Pr[Z_6 \wedge m' \neq \perp | \text{flag} = 1]) \\
&= \frac{1}{4} (\Pr[Z_5 \wedge m' = \perp | \text{flag} = 0] + \Pr[Z_5 \wedge m' \neq \perp | \text{flag} = 1]) \\
&= \frac{1}{4} \Pr[Z_4] .
\end{aligned} \tag{25}$$

At this point, we can find that sk is useless in game G_8 . Therefore, if the event Z_8 occurs, we can construct an adversary \mathcal{B} to attack the OW-CPA security of PKE' as follows: Upon receiving the public key pk and the challenge ciphertext c^* from the OW-CPA game of PKE' , \mathcal{B} randomly chooses $k_0 \leftarrow \$_\mathcal{K}$ and $\text{flag} \leftarrow \$_\{0, 1\}$, and uses $(\text{pk}, c^*, k_0, \text{flag})$ as input to run \hat{B} . When the game ends, \mathcal{B} outputs \hat{m} that outputted by \hat{B} . Therefore,

$$\Pr[Z_8] \leq \text{Adv}_{\text{PKE}'}^{\text{OW-CPA}}(\mathcal{B}) . \tag{26}$$

Combining (13)–(19) and (25)–(26), we obtain

$$\text{Adv}_{\text{KEM}_{RH}}^{\text{IND-1CCA}}(\mathcal{A}) \leq 4\sqrt{\text{Adv}_{\text{PKE}'}^{\text{OW-CPA}}(\mathcal{B}')} + 2\delta .$$

That completes the proof of the theorem. \square

Remark 2. The bound given by Jiang et al. [33] in this case is

$$\text{Adv}_{\text{KEM}_{RH}}^{\text{IND-1CCA}}(\mathcal{A}) \leq 6(q_H + 1)\sqrt{\text{Adv}_{\text{PKE}'}^{\text{OW-CPA}}(\mathcal{B}) + 1/|\mathcal{K}| + \delta} ,$$

where q_H is the number of H random oracle queries made by \mathcal{A} . Despite the unavoidable quadratic reduction loss [37], our reduction is also *tight* in the QROM, with a loss factor of $\mathcal{O}(1)$.

Remark 3. The security proof technique used by Jiang et al. [33] is called (single-classical-query) measure-and-reprogram lemma, which is first proposed by Don et al. [18, 19] and then is extended by Jiang et al. [33]. In this technique, to simulate the decapsulation oracle without sk , the basic strategy adopted by the challenger is to randomly choose one of the q random oracle queries made by \mathcal{A} , measure its input register, consider it as the point that needs reprogramming, and use the reprogrammed random oracle to respond to subsequent random oracle queries. This analysis method needs to consider the impact of different measurements at different times on the final state of \mathcal{A} , and ultimately derives an upper bound for the norm of the final state of \mathcal{A} , which is a sum of

approximately q terms. When considering probability, it is necessary to square this upper bound, and when using Jensen's inequality to relate the probability in a specific case, a coefficient of $\mathcal{O}(q^2)$ will be generated. Therefore, using this technique in security proofs can introduce a loss factor related to q . However, the strategy adopted here is similar to that in the proof in the ROM, where the random oracle and decapsulation oracle are modified during the execution of \mathcal{A} , resulting in a loss factor of only $\mathcal{O}(1)$ for the derived bound. Thus, using our proposed new technique for security proofs will not introduce an additional loss factor exceeding $\mathcal{O}(1)$.

5 The Tightness of the Reduction

Note that Jiang et al. [33] proved that there are unavoidable reduction losses of $\mathcal{O}(q)$ and $\mathcal{O}(q^2)$ in the security proof of T_{RH} in the ROM and the QROM, respectively, where q is the number of random oracle queries made by the adversary. We should stress that instead of indicating any flaw in Jiang et al.'s unavoidability conclusion, our work merely demonstrates that there is a special case which is not captured in their given proofs.

The technique used by Jiang et al. to prove the unavoidability conclusion is a so-called meta-reduction technique [5, 17, 28]. In the case where the underlying PKE' is malleable, the main idea is to use the decapsulation oracle to construct an adversary \mathcal{B} to attack the IND-1CCA security of KEM_{RH} directly.

In the ROM, roughly speaking, given the challenge encapsulation $c^* \leftarrow \text{Enc}'(\text{pk}, m^*)$, \mathcal{B} uses the malleability of PKE' to construct a new encapsulation c' from c^* such that $\text{Dec}(\text{sk}, c') = f(m^*)$, where f is a special function related to the property of the malleability. Then, \mathcal{B} makes a decapsulation query on c' to obtain a $\text{tag} := H(f(m^*), c')$. Finally, through q many H random oracle queries, \mathcal{B} attempts to get $m^* \in \mathcal{M}$ such that $H(f(m^*), c') = \text{tag}$, and then uses $H(m^*, c^*)$ to distinguish k_0 from k_1 . Assume that the underlying PKE scheme PKE' is λ -bit secure, which implies that the probability for any PPT adversary breaking the OW-CPA security of PKE' is no more than $\mathcal{O}(1/2^\lambda)$. Therefore, after q many H random oracle queries, the probability of getting m^* is no more than $\mathcal{O}(q/2^\lambda)$. Therefore, they claim that there is an unavoidable reduction loss of $\mathcal{O}(q)$ in the security proof for T_{RH} in the ROM.

However, we can note that, in the aforementioned attack, the role of the decapsulation oracle is to generate a tag , which is the image of m^* under a deterministic mapping $g(\cdot) := H(f(\cdot), c')$, and the q many H random oracle queries are used to guess the preimage of tag under g . However, in the case where PKE' is deterministic, Enc' itself can provide a deterministic mapping from m^* to c^* that neither relies on the use of the decapsulation oracle nor on queries to the H random oracle. This implies that, at this point, \mathcal{B} does not require access to the decapsulation oracle or the H random oracle; he simply invokes Enc' q times to achieve the effect of invoking the H random oracle q times. Note that \mathcal{B} 's advantage in the OW-CPA game of PKE' surpasses the probability of successfully guessing the plaintext m^* corresponding to the ciphertext c^* by invoking Enc'

q times. Consequently, the aforementioned conclusion regarding an unavoidable reduction loss of $\mathcal{O}(q)$ in the ROM is inapplicable when PKE' is a deterministic public-key encryption scheme. (Clearly, we also present a security proof with a reduction loss of $\mathcal{O}(1)$ as evidence.) Note that in the case where PKE' is probabilistic, the security proof given by Jiang et al. [33] incurs a reduction loss of $\mathcal{O}(q^2)$, which is not as *tight* as claimed in their unavoidability conclusion. Hence, an intriguing open question is whether this $\mathcal{O}(q^2)$ reduction loss is unavoidable when PKE' is probabilistic, or if the reduction loss in this case can be refined.

In the QROM, similar to the case in the ROM, the core idea is to use the malleability of PKE' to generate a new encapsulation c' from c^* , where c^* is the challenge encapsulation, $\text{Enc}(\text{pk}, m^*) = c^*$, $\text{Dec}(\text{sk}, c') = f(m^*)$, and f is a function related to the malleability of PKE' , make the decapsulation oracle query on c' to obtain $k' = H(f(m^*), c')$, use the Grover's algorithm [26] to find m^* from k' , and then use $H(m^*, c^*)$ to distinguish k_0 from k_1 , where the Grover's algorithm needs q times Grover iterations, and each Grover iteration needs to make a random oracle query. Jiang et al. [33] show that this method to distinguish k_0 from k_1 can succeed with probability at least $(q+1)^2/|\mathcal{M}|$, and can derive the conclusion that reduction loss $\mathcal{O}(q^2)$ in the security proof for T_{RH} in the QROM is unavoidable.

Similar to the analysis in the ROM, the use of the random oracle is to compute the deterministic mapping $g(\cdot) := H(f(\cdot), c')$ in order to recover m^* from k' , but the deterministic PKE' itself can provide the deterministic mapping Enc' from m^* to c^* . Therefore, the Grover iteration can use Enc' instead of the random oracle to achieve the same purpose, which implies that the conclusion about the unavoidable reduction loss $\mathcal{O}(q^2)$ for the security proof of T_{RH} in the QROM is inapplicable when PKE' is deterministic.

We should note that in the above analysis, we do not require that the deterministic PKE' should be rigid. Therefore, there is an open problem that when the underlying PKE is deterministic but not rigid, whether the reduction losses of $\mathcal{O}(q)$ and $\mathcal{O}(q^2)$ given by Jiang et al. [33] in the ROM and the QROM, respectively, can be improved or not.

Acknowledgments. We would like to thank all anonymous reviewers for their valuable comments. This work is supported in part by the National Natural Science Foundation of China (Grant No.62122092, No.62202485, No.62032005, No.62425205).

References

1. Alagic, G., Majenz, C., Russell, A., Song, F.: Quantum-access-secure message authentication via blind-unforgeability. In: EUROCRYPT (3). Lecture Notes in Computer Science, vol. 12107, pp. 788–817. Springer (2020). https://doi.org/10.1007/978-3-030-45727-3_27
2. Alkim, E., Bos, J.W., Ducas, L., Longa, P., Mironov, I., Naehrig, M., Nikolaenko, V., Peikert, C., Raghunathan, A., Stebila, D.: Frodokem: Learning with errors key encapsulation (2021), <https://frodokem.org/files/FrodoKEM-specification-20210604.pdf>

3. Ambainis, A., Hamburg, M., Unruh, D.: Quantum security proofs using semi-classical oracles. In: CRYPTO (2). Lecture Notes in Computer Science, vol. 11693, pp. 269–295. Springer (2019). https://doi.org/10.1007/978-3-030-26951-7_10
4. Ananth, P., Kaleoglu, F., Li, X., Liu, Q., Zhandry, M.: On the feasibility of unclonable encryption, and more. In: CRYPTO (2). Lecture Notes in Computer Science, vol. 13508, pp. 212–241. Springer (2022). https://doi.org/10.1007/978-3-031-15979-4_8
5. Bader, C., Jager, T., Li, Y., Schäge, S.: On the impossibility of tight cryptographic reductions. In: EUROCRYPT (2). Lecture Notes in Computer Science, vol. 9666, pp. 273–304. Springer (2016). https://doi.org/10.1007/978-3-662-49896-5_10
6. Bartusek, J., Malavolta, G.: Indistinguishability obfuscation of null quantum circuits and applications. In: ITCS. LIPIcs, vol. 215, pp. 15:1–15:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2022). <https://doi.org/10.4230/LIPICS.ITCS.2022.15>
7. Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: CCS. pp. 62–73. ACM (1993). <https://doi.org/10.1145/168588.168596>
8. Bernstein, D.J., Persichetti, E.: Towards KEM unification. IACR Cryptol. ePrint Arch. p. 526 (2018), <https://eprint.iacr.org/2018/526>
9. Beullens, W., Faugère, J., Koussa, E., Macario-Rat, G., Patarin, J., Perret, L.: Pkp-based signature scheme. In: INDOCRYPT. Lecture Notes in Computer Science, vol. 11898, pp. 3–22. Springer (2019). https://doi.org/10.1007/978-3-030-35423-7_1
10. Bindel, N., Hamburg, M., Hövelmanns, K., Hülsing, A., Persichetti, E.: Tighter proofs of CCA security in the quantum random oracle model. In: TCC (2). Lecture Notes in Computer Science, vol. 11892, pp. 61–90. Springer (2019). https://doi.org/10.1007/978-3-030-36033-7_3
11. Boneh, D., Dagdelen, Ö., Fischlin, M., Lehmann, A., Schaffner, C., Zhandry, M.: Random oracles in a quantum world. In: ASIACRYPT. Lecture Notes in Computer Science, vol. 7073, pp. 41–69. Springer (2011). https://doi.org/10.1007/978-3-642-25385-0_3
12. Boneh, D., Shoup, V.: A Graduate Course in Applied Cryptography (2023), <http://toc.cryptobook.us/>
13. Bos, J.W., Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schanck, J.M., Schwabe, P., Seiler, G., Stehlé, D.: CRYSTALS - kyber: A cca-secure module-lattice-based KEM. In: EuroS&P. pp. 353–367. IEEE (2018). <https://doi.org/10.1109/EUROSP.2018.00032>
14. Brendel, J., Fiedler, R., Günther, F., Janson, C., Stebila, D.: Post-quantum asynchronous deniable key exchange and the signal handshake. In: Public Key Cryptography (2). Lecture Notes in Computer Science, vol. 13178, pp. 3–34. Springer (2022). https://doi.org/10.1007/978-3-030-97131-1_1
15. Chia, N., Chung, K., Yamakawa, T.: Classical verification of quantum computations with efficient verifier. In: TCC (3). Lecture Notes in Computer Science, vol. 12552, pp. 181–206. Springer (2020). https://doi.org/10.1007/978-3-030-64381-2_7
16. Cini, V., Ramacher, S., Slamanig, D., Striecks, C.: Cca-secure (puncturable) kems from encryption with non-negligible decryption errors. In: ASIACRYPT (1). Lecture Notes in Computer Science, vol. 12491, pp. 159–190. Springer (2020). https://doi.org/10.1007/978-3-030-64837-4_6
17. Coron, J.: Optimal security proofs for PSS and other signature schemes. In: EUROCRYPT. Lecture Notes in Computer Science, vol. 2332, pp. 272–287. Springer (2002). https://doi.org/10.1007/3-540-46035-7_18

18. Don, J., Fehr, S., Majenz, C.: The measure-and-reprogram technique 2.0: Multi-round fiat-shamir and more. In: CRYPTO (3). Lecture Notes in Computer Science, vol. 12172, pp. 602–631. Springer (2020). https://doi.org/10.1007/978-3-030-56877-1_21
19. Don, J., Fehr, S., Majenz, C., Schaffner, C.: Security of the fiat-shamir transformation in the quantum random-oracle model. In: CRYPTO (2). Lecture Notes in Computer Science, vol. 11693, pp. 356–383. Springer (2019). https://doi.org/10.1007/978-3-030-26951-7_13
20. Don, J., Fehr, S., Majenz, C., Schaffner, C.: Online-extractability in the quantum random-oracle model. In: EUROCRYPT (3). Lecture Notes in Computer Science, vol. 13277, pp. 677–706. Springer (2022). https://doi.org/10.1007/978-3-031-07082-2_24
21. Dowling, B., Fischlin, M., Günther, F., Stebila, D.: A cryptographic analysis of the TLS 1.3 handshake protocol. *J. Cryptol.* **34**(4), 37 (2021). <https://doi.org/10.1007/S00145-021-09384-1>
22. Fujisaki, E., Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. In: CRYPTO. Lecture Notes in Computer Science, vol. 1666, pp. 537–554. Springer (1999). https://doi.org/10.1007/3-540-48405-1_34
23. Fujisaki, E., Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. *J. Cryptol.* **26**(1), 80–101 (2013). <https://doi.org/10.1007/S00145-011-9114-1>
24. Galbraith, S.D., Petit, C., Silva, J.: Identification protocols and signature schemes based on supersingular isogeny problems. In: ASIACRYPT (1). Lecture Notes in Computer Science, vol. 10624, pp. 3–33. Springer (2017). https://doi.org/10.1007/978-3-319-70694-8_1
25. Ge, J., Shan, T., Xue, R.: Tighter qcca-secure key encapsulation mechanism with explicit rejection in the quantum random oracle model. In: CRYPTO (5). Lecture Notes in Computer Science, vol. 14085, pp. 292–324. Springer (2023). https://doi.org/10.1007/978-3-031-38554-4_10
26. Grover, L.K.: Quantum mechanics helps in searching for a needle in a haystack. *Phys. Rev. Lett.* **79**, 325–328 (1997). <https://doi.org/10.1103/PhysRevLett.79.325>
27. Hofheinz, D., Hövelmanns, K., Kiltz, E.: A modular analysis of the fujisaki-okamoto transformation. In: TCC (1). Lecture Notes in Computer Science, vol. 10677, pp. 341–371. Springer (2017). https://doi.org/10.1007/978-3-319-70500-2_12
28. Hofheinz, D., Jager, T., Knapp, E.: Waters signatures with optimal security reduction. In: Public Key Cryptography. Lecture Notes in Computer Science, vol. 7293, pp. 66–83. Springer (2012). https://doi.org/10.1007/978-3-642-30057-8_5
29. Hosoyamada, A., Iwata, T.: On tight quantum security of HMAC and NMAC in the quantum random oracle model. In: CRYPTO (1). Lecture Notes in Computer Science, vol. 12825, pp. 585–615. Springer (2021). https://doi.org/10.1007/978-3-030-84242-0_21
30. Hövelmanns, K., Kiltz, E., Schäge, S., Unruh, D.: Generic authenticated key exchange in the quantum random oracle model. In: Public Key Cryptography (2). Lecture Notes in Computer Science, vol. 12111, pp. 389–422. Springer (2020). https://doi.org/10.1007/978-3-030-45388-6_14
31. Huguenin-Dumittan, L., Vaudenay, S.: On ind-qcca security in the ROM and its applications - CPA security is sufficient for TLS 1.3. In: EUROCRYPT (3). Lecture Notes in Computer Science, vol. 13277, pp. 613–642. Springer (2022). https://doi.org/10.1007/978-3-031-07082-2_22

32. Jaeger, J., Song, F., Tessaro, S.: Quantum key-length extension. In: TCC (1). Lecture Notes in Computer Science, vol. 13042, pp. 209–239. Springer (2021). https://doi.org/10.1007/978-3-030-90459-3_8
33. Jiang, H., Ma, Z., Zhang, Z.: Post-quantum security of key encapsulation mechanism against CCA attacks with a single decapsulation query. In: ASIACRYPT (4). Lecture Notes in Computer Science, vol. 14441, pp. 434–468. Springer (2023). https://doi.org/10.1007/978-981-99-8730-6_14
34. Jiang, H., Zhang, Z., Chen, L., Wang, H., Ma, Z.: Ind-cca-secure key encapsulation mechanism in the quantum random oracle model, revisited. In: CRYPTO (3). Lecture Notes in Computer Science, vol. 10993, pp. 96–125. Springer (2018). https://doi.org/10.1007/978-3-319-96878-0_4
35. Jiang, H., Zhang, Z., Ma, Z.: Key encapsulation mechanism with explicit rejection in the quantum random oracle model. In: Public Key Cryptography (2). Lecture Notes in Computer Science, vol. 11443, pp. 618–645. Springer (2019). https://doi.org/10.1007/978-3-030-17259-6_21
36. Jiang, H., Zhang, Z., Ma, Z.: Tighter security proofs for generic key encapsulation mechanism in the quantum random oracle model. In: PQCrypto. Lecture Notes in Computer Science, vol. 11505, pp. 227–248. Springer (2019). https://doi.org/10.1007/978-3-030-25510-7_13
37. Jiang, H., Zhang, Z., Ma, Z.: On the non-tightness of measurement-based reductions for key encapsulation mechanism in the quantum random oracle model. In: ASIACRYPT (1). Lecture Notes in Computer Science, vol. 13090, pp. 487–517. Springer (2021). https://doi.org/10.1007/978-3-030-92062-3_17
38. Katsumata, S., Yamada, S., Yamakawa, T.: Tighter security proofs for GPV-IBE in the quantum random oracle model. In: ASIACRYPT (2). Lecture Notes in Computer Science, vol. 11273, pp. 253–282. Springer (2018). https://doi.org/10.1007/978-3-030-03329-3_9
39. Kitagawa, F., Nishimaki, R.: KDM security for the fujisaki-okamoto transformations in the QROM. In: Public Key Cryptography (2). Lecture Notes in Computer Science, vol. 13178, pp. 286–315. Springer (2022). https://doi.org/10.1007/978-3-030-97131-1_10
40. Kuchta, V., Sakzad, A., Stehlé, D., Steinfeld, R., Sun, S.: Measure-rewind-measure: Tighter quantum random oracle model proofs for one-way to hiding and CCA security. In: EUROCRYPT (3). Lecture Notes in Computer Science, vol. 12107, pp. 703–728. Springer (2020). https://doi.org/10.1007/978-3-030-45727-3_24
41. Lyu, Y., Liu, S.: Two-message authenticated key exchange from public-key encryption. In: ESORICS (1). Lecture Notes in Computer Science, vol. 14344, pp. 414–434. Springer (2023). https://doi.org/10.1007/978-3-031-50594-2_21
42. Nielsen, M.A., Chuang, I.L.: Quantum Computation and Quantum Information (10th Anniversary edition). Cambridge University Press (2016), <https://www.cambridge.org/de/academic/subjects/physics/quantum-physics-quantum-information-and-quantum-computation/quantum-computation-and-quantum-information-10th-anniversary-edition?format=HB>
43. Okamoto, T., Pointcheval, D.: REACT: rapid enhanced-security asymmetric cryptosystem transform. In: CT-RSA. Lecture Notes in Computer Science, vol. 2020, pp. 159–175. Springer (2001). https://doi.org/10.1007/3-540-45353-9_13
44. Pan, J., Wagner, B., Zeng, R.: Tighter security for generic authenticated key exchange in the QROM. In: ASIACRYPT (4). Lecture Notes in Computer Science, vol. 14441, pp. 401–433. Springer (2023). https://doi.org/10.1007/978-981-99-8730-6_13

45. Pan, J., Zeng, R.: Selective opening security in the quantum random oracle model, revisited. In: Public Key Cryptography (3). Lecture Notes in Computer Science, vol. 14603, pp. 92–122. Springer (2024). https://doi.org/10.1007/978-3-031-57725-3_4
46. Sageloli, É., Pébereau, P., Méaux, P., Chevalier, C.: Shorter and faster identity-based signatures with tight security in the (Q)ROM from lattices. In: ACNS (1). Lecture Notes in Computer Science, vol. 13905, pp. 634–663. Springer (2023). https://doi.org/10.1007/978-3-031-33488-7_24
47. Saito, T., Xagawa, K., Yamakawa, T.: Tightly-secure key-encapsulation mechanism in the quantum random oracle model. In: EUROCRYPT (3). Lecture Notes in Computer Science, vol. 10822, pp. 520–551. Springer (2018). https://doi.org/10.1007/978-3-319-78372-7_17
48. Shan, T., Ge, J., Xue, R.: Qcca-secure generic transformations in the quantum random oracle model. In: Public Key Cryptography (1). Lecture Notes in Computer Science, vol. 13940, pp. 36–64. Springer (2023). https://doi.org/10.1007/978-3-031-31368-4_2
49. Tanaka, Y., Ueno, R., Xagawa, K., Ito, A., Takahashi, J., Homma, N.: Multiple-valued plaintext-checking side-channel attacks on post-quantum kems. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2023**(3), 473–503 (2023). <https://doi.org/10.46586/TCHES.V2023.I3.473-503>
50. Targhi, E.E., Unruh, D.: Post-quantum security of the fujisaki-okamoto and OAEP transforms. In: TCC (B2). Lecture Notes in Computer Science, vol. 9986, pp. 192–216 (2016). https://doi.org/10.1007/978-3-662-53644-5_8
51. Unruh, D.: Quantum position verification in the random oracle model. In: CRYPTO (2). Lecture Notes in Computer Science, vol. 8617, pp. 1–18. Springer (2014). https://doi.org/10.1007/978-3-662-44381-1_1
52. Xagawa, K., Yamakawa, T.: (tightly) qcca-secure key-encapsulation mechanism in the quantum random oracle model. In: PQCrypto. Lecture Notes in Computer Science, vol. 11505, pp. 249–268. Springer (2019). https://doi.org/10.1007/978-3-030-25510-7_14
53. Yamakawa, T., Zhandry, M.: Classical vs quantum random oracles. In: EUROCRYPT (2). Lecture Notes in Computer Science, vol. 12697, pp. 568–597. Springer (2021). https://doi.org/10.1007/978-3-030-77886-6_20
54. Zhandry, M.: Secure identity-based encryption in the quantum random oracle model. In: CRYPTO. Lecture Notes in Computer Science, vol. 7417, pp. 758–775. Springer (2012). https://doi.org/10.1007/978-3-642-32009-5_44
55. Zhandry, M.: How to record quantum queries, and applications to quantum indistinguishability. In: CRYPTO (2). Lecture Notes in Computer Science, vol. 11693, pp. 239–268. Springer (2019). https://doi.org/10.1007/978-3-030-26951-7_9
56. Zhang, J., Yu, Y., Feng, D., Fan, S., Zhang, Z.: On the (quantum) random oracle methodology: New separations and more. IACR Cryptol. ePrint Arch. p. 1101 (2019), <https://eprint.iacr.org/2019/1101>

Secure Data Structures



Deletions and Dishonesty: Probabilistic Data Structures in Adversarial Settings

Mia Filić^(✉), Keran Kocher, Ella Kummer, and Anupama Unnikrishnan

ETH Zurich, Zürich, Switzerland
mfilic@ethz.ch

Abstract. Probabilistic data structures (PDS) are compact representations of high-volume data that provide approximate answers to queries about the data. They are commonplace in today’s computing systems, finding use in databases, networking and more. While PDS are designed to perform well under benign inputs, they are frequently used in applications where inputs may be adversarially chosen. This may lead to a violation of their expected behaviour, for example an increase in false positive rate.

In this work, we focus on PDS that handle approximate membership queries (AMQ). We consider adversarial users with the capability of making adaptive insertions, deletions and membership queries to AMQ-PDS, and analyse the performance of AMQ-PDS under such adversarial inputs.

We argue that deletions significantly empower adversaries, presenting a challenge to enforcing honest behaviour when compared to insertion-only AMQ-PDS. To address this, we introduce a new concept of an honest setting for AMQ-PDS with deletions. By leveraging simulation-based security definitions, we then quantify how much harm can be caused by adversarial users to the functionality of AMQ-PDS. Our resulting bounds only require calculating the maximal false positive probability and insertion failure probability achievable in our novel honest setting.

We apply our results to Cuckoo filters and Counting filters. We show how to protect these AMQ-PDS at low cost, by replacing or composing the hash functions with keyed pseudorandom functions in their construction. This strategy involves establishing practical bounds for the probabilities mentioned above. Using our new techniques, we demonstrate that achieving security against adversarial users making both insertions *and* deletions remains practical.

Keywords: probabilistic data structures · Counting filters · Cuckoo filters · security · simulation-based proofs

1 Introduction

Probabilistic data structures (PDS) are widespread in today’s data-driven world. They find a multitude of uses across our computing systems, in databases,

networking and communication. By compactly representing data, they offer improved efficiency, with the tradeoff of providing approximate (rather than exact) answers to queries about the data.

Each PDS is specifically designed to answer certain kinds of queries. An important category of PDS is those providing approximate answers to membership queries, i.e. “is an element x a member of a set S ?”. We refer to this category as AMQ-PDS, which are the focus of this work. Examples of AMQ-PDS include Bloom filters [5], Counting filters [12] and Cuckoo filters [10]. While Bloom filters only support insertions of elements into the set, Counting and Cuckoo filters also allow elements to be deleted. Other categories of PDS include frequency estimators such as Count-Min sketches [9] and Heavy Keepers [17], and cardinality estimators such as HyperLogLog [14] and KMV sketches [3].

PDS find a myriad of applications, from estimating the number of distinct Google search queries [19] and detecting anomalies in network traffic [17, 22] to building privacy-preserving recommendation systems [28]. AMQ-PDS are beneficial for database query speedup [34], spam detection [38], resource and packet routing in networks [6], certificate revocation systems [23], DNA sequence analysis [29, 37], and more [26]. In particular, AMQ-PDS that support deletions, such as Counting and Cuckoo filters, are useful for cache sharing among web proxies [12], speedup of post-quantum TLS handshakes [36], efficient certificate revocation checking [35], mobile private contact discovery [18, 20], and fighting fake news [25].

The wide deployment of PDS across applications, however, comes with an increasing risk of adversarial interference. By carefully choosing inputs, malicious users can force specific elements to become false positives in AMQ-PDS [16], cause frequencies of elements to be overestimated [8, 27], or artificially inflate cardinality estimates [33], for example. This leads to dangerous consequences for the use of PDS in practice. In spite of this, such adversarial settings are typically not covered by the performance guarantees of PDS; their expected behaviour is characterised assuming honest inputs. To protect PDS against adversarial influence, cryptographic techniques can be a powerful tool. Combining PDS with cryptography results in a significant new research area with many critical open questions.

1.1 Our Contributions

In this paper, we study the correctness of AMQ-PDS in adversarial settings. We focus on malicious users interacting with an AMQ-PDS hosted by an honest service provider. In practice, users interact with the AMQ-PDS through an API, allowing dynamic updates to the stored dataset, through insertions and deletions, as well as membership queries. In this work, we address how malicious users can leverage adaptive insertions, membership queries *and* deletions to manipulate the performance of AMQ-PDS. We will argue that deletions, in particular, are a powerful tool for adversaries. While we focus on two commonly used AMQ-PDS, Cuckoo filters [10] and Counting filters [12], our definitions are general and can be applied to a broad range of AMQ-PDS.

Syntax for AMQ-PDS. Inspired by [8, 13], we establish a syntax for AMQ-PDS that support insertions, deletions and membership queries. We identify consistency rules for the behaviour of AMQ-PDS, satisfied by Counting and Cuckoo filters, that will allow us to prove results on their adversarial correctness.

Simulation-Based framework. We employ a simulation-based approach [24] to define security, following recent work [13, 33]. In this approach, the adversary is modelled as interacting with the AMQ-PDS in either a “real world” or an “ideal world”. In the real world, the adversary has access to the AMQ-PDS through an API that allows it to insert and delete items, and make membership queries. In the ideal world, the adversary instead interacts with a simulator that models honest behaviour of the AMQ-PDS. At the end of its execution, the adversary produces an output, which is used to distinguish between the two worlds. By quantifying the distance between the worlds, we bound how much harm the adversary can do in the real world by relating it to the honest operation of the AMQ-PDS in the ideal world.

Simulation-based security definitions are traditionally used to analyse notions of privacy (for example, in searchable encryption [7]), where the simulator is given some leakage. By proving that the two worlds are indistinguishable, one concludes that the adversary can only learn this leakage, which is deemed acceptable. In contrast, our approach does not require indistinguishability between the worlds in order to give useful bounds; we will show how they can be used to set parameters for secure PDS in practice.

The power of the simulation-based approach in analysing correctness is that it covers all adversarial goals, in contrast to the game-based approach with a specific adversarial goal [8]. In practice, this means that one only needs to compute the probability of achieving a particular goal in the honest setting (which is well-studied in the PDS literature), in order to upper bound the probability of achieving it in the adversarial setting.

Adversarial Correctness for AMQ-PDS with Insertions and Deletions. To analyse adversarial correctness using the simulation-based approach, the first question to address is how to define “honest” behaviour. Allowing deletions (in addition to insertions and membership queries), however, introduces substantial hurdles.

In [13], the notion of a non-adversarially-influenced (NAI) state was proposed for insertion-only AMQ-PDS. Intuitively, this captures the idea that the state of an AMQ-PDS can be thought of as honest if one cannot predict the effect of each insertion on the state, prior to the insertion. To achieve this for many prominent AMQ-PDS, one can replace the hash functions used in their constructions with keyed Pseudo-Random Functions (PRFs).

With deletions, however, the above idea no longer suffices to capture honesty. The ability to delete elements after inserting them means that an adversary could effectively reset the state if not satisfied. For example, consider cache summarisation for content routing [1, 12]. Here, an element is automatically added to or removed from the filter whenever the cache is updated. The cache’s size poses

a natural bound on the number of elements the filter stores. So, the attacker might want to force removal of elements that do not contribute to its goal of, for example, increasing the false positive probability (FPP) or making a specific target a false positive. The former significantly increases time for content retrieval on average, while the latter substantially increases retrieval time of the targeted content. While such a final state satisfy insertion unpredictability, it would still be adversarially influenced. Therefore, the deletion functionality of AMQ-PDS forms an intrinsic barrier to enforcing honesty.

Further, another complication arises from false negatives. While insertion-only AMQ-PDS may have false positives (elements that appear to be in the set when they have not been inserted), deletions may also lead to false negatives (elements that appear to not be in the set when they have not been deleted). The FPP of AMQ-PDS is typically well-characterised; false negatives, which can arise (for example) through deleting elements that were never inserted, are often assumed not to occur under honest operation. In an adversarial setting, we can no longer assume this.

We circumvent these obstacles by proposing a new notion of honesty for AMQ-PDS with both insertions and deletions, which we call NAI*. We show that building a simulator that satisfies NAI* suffices to analyse adversarial correctness for our AMQ-PDS of interest.

Our results show how to provably protect AMQ-PDS by replacing or composing public hash functions with PRFs and giving concrete bounds on the probability of achieving any adversarial goal through adaptive queries. Practitioners can use our concrete bounds to set AMQ-PDS parameters that guarantee security even with adversarial users. This is in contrast to how parameters are currently set in practice, with bounds on (for example) FPP being easily violated through precomputation attacks (on public hash functions). Using our results, practitioners can guarantee that FPP will stay below a certain threshold even with adaptive queries. This extends to any adversarial goal, e.g. creating false negatives, causing insertion failures. By showing how to ensure AMQ-PDS behave as expected even with malicious users, our work impacts any application of AMQ-PDS - in particular, applications requiring dynamic deletions, insertions and membership queries (e.g. cache sharing, coupon validation, etc.).

We emphasise that our focus is on users exploiting the API that allows interaction with an AMQ-PDS hosted by an honest service provider. To our knowledge, such an API typically does not allow users to view its internal state, e.g. [2]. Of course, in a different adversarial scenario with a compromised service provider, users could gain access to the state. While out of scope in this work, we later discuss why our results are not directly applicable to such a setting in Remark 3.

Analysis of Counting and Cuckoo filters. We conclude by providing a concrete evaluation of our security theorems by analysing Counting and Cuckoo filters. The usage of public hash functions in their original formulations leads to vulnerabilities from precomputation attacks [8, 16]. Using our theorems, we demonstrate how to provably protect them by replacing or composing the hash functions with

PRFs (at the cost of needing secure key management). This requires deriving novel bounds on their NAI* false positive probability, as well as their NAI* insertion failure probability, both of which we show how to upper bound using results from the (insertion-only) AMQ-PDS literature.

Finally, we investigate the impact of our analysis for choosing appropriate parameters to secure AMQ-PDS in practice. Our results illustrate that protecting AMQ-PDS against adversarial users who can harness their full functionality is practical. Further, as a result of our new insights and techniques, extending the user’s capabilities to include deletions does not compromise security.

1.2 Related Work

In [13], Filić *et al.* proposed a simulation-based framework for analysing the adversarial correctness and privacy of AMQ-PDS that only support insertions. By building a simulator that models the non-adversarial operation of AMQ-PDS, they derived bounds on the closeness of an adversarially generated state to that of an honest one, applying their framework to derive correctness guarantees for Bloom and insertion-only Cuckoo filters under adversarial inputs. In our work, we use a similar methodology but cover the full functionality of AMQ-PDS, i.e. allowing deletions as well as insertions. Thus, we solve an important question left unanswered by their work, resulting in a more complete analysis of adversarial correctness of AMQ-PDS.

A simulation-based approach was also employed in [33] to study the Hyper-LogLog cardinality estimator in adversarial settings. While our proof technique is conceptually similar, the types of queries supported by AMQ-PDS lead to more powerful adversarial strategies, and thus a more complicated analysis.

The work of Clayton *et al.* [8] focused on the adversarial correctness of AMQ-PDS Bloom and Counting filters. They examined an “ ℓ -thresholded” variant of Counting filters, where insertions are disallowed if more than ℓ counters are set. Their approach utilised a game-based formalism, which required defining a specific winning condition for the adversary, i.e. finding a certain number of false positives or false negatives. We provide a more detailed comparison of our work with [8] in the full version. A similar approach was adopted in [4] with an adversary who tries to maximise the false positive rate of AMQ-PDS by repeating membership queries. In contrast to these game-based methods, the simulation-based formalism does not require specifying an adversarial goal. This allows one to use our results to re-derive bounds for any specific adversary.

In [31,32], Naor and Yogev studied the adversarial correctness of Bloom filters, again using a game-based approach. Recent work by Naor and Oved [30] further extended this to propose various robustness notions for Bloom filters. However, their adversarial model is more restricted than ours, without the ability to make adaptive insertions and membership queries. Further, as their focus is on Bloom filters, deletions do not play a role.

In [39], Yeo analysed Cuckoo hash tables, which are closely related to Cuckoo filters. However, they considered a static adversarial setting, where a set of elements is inserted at the start, with a specific adversarial goal of causing the

insertion of this set to fail. In this work, we are interested in a more powerful setting where adversaries can dynamically update the dataset and can have any goal. Adversarial influence on the false positive rate of Cuckoo filters was studied in [21], but in a similarly restricted adversarial model.

Therefore, in comparison to previous work, we are the first to rigorously analyse adversarial correctness of Counting and Cuckoo filters in their full capability, for any adversarial goal. This fills a significant gap in the literature.

For scenarios where the data itself is sensitive, studying privacy might also become important. Leveraging the power of deletions to deduce information about elements in Counting filters, [15] proposed attacks on their privacy. This highlights an intrinsic challenge in enforcing privacy for AMQ-PDS with deletions, leaving the task an interesting open question.

1.3 Paper Organisation

We start with preliminaries in Sect. 2. In Sect. 3.1, we define the syntax for AMQ-PDS with deletions, the notion of a non-adversarial setting, and properties of our AMQ-PDS of interest. We analyse adversarial correctness in Sect. 4, and discuss the usefulness of our results in practice in Sect. 5.

2 Preliminaries

Notation. We follow the notation of [13], repeated here for clarity. For an integer $m \in \mathbb{Z}_{\geq 1}$, we write $[m]$ to denote the set $\{1, 2, \dots, m\}$. We consider all logarithms to be in base 2. Given two sets \mathfrak{D} and \mathfrak{R} , we define $\text{Funcs}[\mathfrak{D}, \mathfrak{R}]$ to be the set of functions from \mathfrak{D} to \mathfrak{R} . We write $F \leftarrow_s \text{Funcs}[\mathfrak{D}, \mathfrak{R}]$ to mean that F is a random function $\mathfrak{D} \xrightarrow{F} \mathfrak{R}$. Given a set S , we denote the identity function over S as $\text{Id}_S: S \rightarrow S$. For a probability distribution D , we write $x \leftarrow_s D$ to mean that x is sampled according to D . We define the statistical distance between two random variables X, Y with finite support $D = \text{Supp}(X) = \text{Supp}(Y)$ as $SD(X, Y) := \frac{1}{2} \sum_{z \in D} |\Pr[X = z] - \Pr[Y = z]|$. For a set S (resp. a list L), we denote by $|S|$ (resp. $|L|$) the number of elements in S (resp. L). A fixed-length list of length s initialised empty is denoted by $a \leftarrow \perp^s$. We denote by $\text{load}(a)$ the number of set entries of a . To insert an entry x into the first unused slot in a we write $a' \leftarrow a \diamond x$ such that $a' = x \perp \dots \perp$ with $s-1$ trailing \perp s and $\text{load}(a') = 1$. A further insertion $a'' \leftarrow a' \diamond y$ results in $a'' = xy \perp \dots \perp$ with $\text{load}(a'') = 2$, and so on. We refer to the i -th entry in a list a as $a[i]$. In algorithms, we assume that all key-value stores are initialised with value \perp at every index, using the convention that $\perp < n, \forall n \in \mathbb{R}$, and we denote it as $\{\}$. For a key-value store a , we refer to the value of the entry with key k as $a[k]$. We write variable assignments using \leftarrow , unless the value is output by a randomised algorithm, for which we use \leftarrow_s .

For a randomised algorithm alg , we write output $\leftarrow \text{alg}(\text{input}_1, \text{input}_2, \dots, \text{input}_\ell; r)$, where $r \in \mathcal{R}$ denotes the coins that can be used by alg and \mathcal{R} is the set of possible coins. We may also suppress coins whenever it is notationally convenient to do so. For a deterministic algorithm, r can be set to \perp . We remark

$Exp_R^{PRF}(\mathcal{B})$	Oracle $\mathbf{RoR}(x)$
1 $K \leftarrow_s \mathcal{K}; F \leftarrow_s \text{Func}[\mathcal{D}, \mathfrak{R}]$	1 if $b = 0$: $y \leftarrow R_K(x)$
2 $b \leftarrow_s \{0, 1\}; b' \leftarrow_s \mathcal{B}^{\mathbf{RoR}}$	2 else : $y \leftarrow F(x)$
3 return b'	3 return y

Fig. 1. The PRF experiment.

that the output of a randomised algorithm can be seen as a random variable over the output space of the algorithm. Unless otherwise specified, we will consider random coins to be sampled uniformly from \mathcal{R} , independently from all other inputs and/or state, and refer to such r as “freshly sampled”. If alg is given oracle access to functions f_1, \dots, f_n , we denote it by $\text{alg}^{f_1 \dots f_n}$.

We will consider AMQ-PDS that can store elements from finite domains \mathcal{D} by letting $\mathcal{D} = \cup_{\ell=0}^L \{0, 1\}^\ell$ for some large but finite value of L , say $L = 2^{64}$. In our constructions, we will make use of pseudorandom functions, which we will model as truly random functions to which the AMQ-PDS has oracle access.

Definition 1. Consider the PRF experiment in Fig. 1. We say a pseudorandom function family $R: \mathcal{K} \times \mathcal{D} \rightarrow \mathfrak{R}$ is (q, t, ε) -secure if for all adversaries \mathcal{B} running in time at most t and making at most q queries to its \mathbf{RoR} oracle in Exp_R^{PRF} ,

$$Adv_R^{PRF}(\mathcal{B}) := |\Pr[b' = 1|b = 0] - \Pr[b' = 1|b = 1]| \leq \varepsilon.$$

We say \mathcal{B} is a (q, t) -PRF adversary.

3 AMQ-PDS

In this section, we formalise the syntax of AMQ-PDS and their behaviour under non-adversarial inputs. We formally define our AMQ-PDS of interest, Counting and Cuckoo filters, and discuss some common properties that they satisfy.

3.1 Syntax

We now define the syntax of an AMQ-PDS, extending that of [13] to include deletions. Let Π be an AMQ-PDS. We denote its public parameters by pp , and its state as $\sigma \in \Sigma$, where Σ denotes the space of possible states of Π . The set of elements that can be inserted into Π is denoted by \mathcal{D} , unless stated otherwise. We consider a syntax consisting of four algorithms:

- The setup algorithm $\sigma \leftarrow \text{setup}(pp; r)$ sets up the initial state of an empty PDS with public parameters pp ; it will always be called first to initialise the AMQ-PDS.
- The insertion algorithm $(b, \sigma') \leftarrow \text{ins}(x, \sigma; r)$, given an element $x \in \mathcal{D}$, attempts to insert it into the AMQ-PDS, and returns a bit $b \in \{\perp, \top\}$ representing whether the insertion was successful ($b = \top$) or not ($b = \perp$), and the state σ' of the AMQ-PDS after the insertion.

- The deletion algorithm $(b, \sigma') \leftarrow \text{del}(x, \sigma)$, given an element $x \in \mathfrak{D}$, attempts to delete x from the AMQ-PDS, i.e. attempts to remove everything that a successful insertion on x added to σ . The algorithm return a bit $b \in \{\perp, \top\}$ representing whether the deletion was successful ($b = \top$) or not ($b = \perp$), and the state σ' of the AMQ-PDS after the deletion.
- The membership querying algorithm $b \leftarrow \text{qry}(x, \sigma)$, given an element $x \in \mathfrak{D}$, returns a bit $b \in \{\perp, \top\}$ (approximately) answering whether x was previously inserted ($b = \top$) or not ($b = \perp$) into the AMQ-PDS.

We remark that we only consider AMQ-PDS where membership queries do not change the state of the AMQ-PDS; thus, `qry` does not need to output a new σ' value. This includes popular AMQ-PDS such as Counting and Cuckoo filters.

Due to the approximate nature of AMQ-PDS, `qry` calls may return a false positive result with a certain probability. That is, we may have $\top \leftarrow \text{qry}(x, \sigma)$ even though no call $\text{ins}(x, \sigma'; r)$ was made post setup and prior to the membership query. We refer to the probability $\Pr[\top \leftarrow \text{qry}(x, \sigma) \mid x \text{ was not inserted into } \Pi]$ as the *false positive probability* of an AMQ-PDS Π . In addition, since Counting and Cuckoo filters support deletions, `qry` calls may return a false negative result, where we may have $\perp \leftarrow \text{qry}(x, \sigma)$ even though an $\text{ins}(x, \sigma'; r)$ call was made beforehand. We refer to the probability $\Pr[\perp \leftarrow \text{qry}(x, \sigma) \mid x \text{ was inserted into } \Pi]$ as the *false negative probability* of an AMQ-PDS Π .

Moreover, the insertion algorithm may fail to insert an element, for example if the AMQ-PDS has reached capacity. We denote the probability $\Pr[(\perp, \sigma) \leftarrow \text{ins}(x, \sigma)]$ as the *insertion failure probability*.

3.2 AMQ-PDS Under Non-adversarial Inputs

We now define the expected behaviour of AMQ-PDS in a non-adversarial setting, since we will later quantify how much the state of an AMQ-PDS can deviate from this under adversarial inputs. As in [13], we will focus on AMQ-PDS that satisfy the following properties of *function-decomposability* and *reinsertion invariance*.

Definition 2 (Function-decomposability [13]). *Let Π be an AMQ-PDS and let $F \leftarrow_s \text{Funcs}[\mathfrak{D}, \mathfrak{R}]$ with $\mathfrak{R} \subset \mathfrak{D}$ be a random function to which Π has oracle access. We say Π is F -decomposable if*

$$\begin{aligned} \text{ins}^F(x, \sigma; r) &= \text{ins}^{Id_{\mathfrak{R}}}(F(x), \sigma; r) \quad \forall x \in \mathfrak{D}, \sigma \in \Sigma, r \in \mathcal{R}, \\ \text{del}^F(x, \sigma) &= \text{del}^{Id_{\mathfrak{R}}}(F(x), \sigma) \quad \forall x \in \mathfrak{D}, \sigma \in \Sigma, \\ \text{qry}^F(x, \sigma) &= \text{qry}^{Id_{\mathfrak{R}}}(F(x), \sigma) \quad \forall x \in \mathfrak{D}, \sigma \in \Sigma, \end{aligned}$$

where $\text{ins}^{Id_{\mathfrak{R}}}$, $\text{del}^{Id_{\mathfrak{R}}}$ and $\text{qry}^{Id_{\mathfrak{R}}}$ cannot internally evaluate F due to not having oracle access to it and F being truly random. Function-decomposability also applies to AMQ-PDS with oracle access to multiple functions.

Definition 3 (Reinsertion invariance [13]). *Let Π be an AMQ-PDS. We say Π is reinsertion invariant if for all $x \in \mathfrak{D}, \sigma \in \Sigma$ such that $\top \leftarrow \text{qry}(x, \sigma)$, we have $(\top, \sigma') \leftarrow \text{ins}(x, \sigma; r) \implies \sigma = \sigma' \forall r \in \mathcal{R}$.*

Reinsertion invariance is a natural property to expect from AMQ-PDS since they are designed to represent sets and not multisets. Note that if reinsertion invariance does not apply, simply repeatedly inserting a single element could lead to blocking of further insertions.

If a reinsertion-invariant AMQ-PDS contains multiple copies of the same element, deleting one copy will result in all other copies being deleted. However, reinsertion invariance does not require the state of the AMQ-PDS to remain unchanged if elements are reinserted after being deleted.

For an *insertion-only* AMQ-PDS satisfying function-decomposability and reinsertion invariance, the notion of a non-adversarially influenced state was proposed in [13]. We give an alternative (but equivalent) definition below.

Definition 4 (*n*-NAI state). *Let Π be an AMQ-PDS with public parameters pp using $F = \text{Id}_{\mathfrak{R}}$ satisfying reinsertion invariance, and let $\sigma \leftarrow \text{setup}(pp)$. Let n be a non-negative integer. Let $X_1, \dots, X_n \leftarrow_{\$} \mathfrak{R}$. Let L be the list of operations on σ , where $L = [\text{ins}^{Id_{\mathfrak{R}}}(X_1, \sigma), \dots, \text{ins}^{Id_{\mathfrak{R}}}(X_n, \sigma)]$. Then, σ is an n -NAI state.*

We then give an alternative (but equivalent) definition of the NAI false positive probability from [13].

Definition 5 (NAI false positive probability). *Let Π be an AMQ-PDS with public parameters pp , using a random function $F : \mathfrak{D} \rightarrow \mathfrak{R}$ satisfying F -decomposability and reinsertion invariance. Let n be a non-negative integer. Define the NAI false positive probability after n distinct insertions as*

$$P_{\Pi, pp}(FP | n) := \Pr \left[\begin{array}{c} \sigma \leftarrow \text{setup}(pp) \\ \text{for } i \in [n] : (b, \sigma) \leftarrow_{\$} \text{ins}^{Id_{\mathfrak{R}}}(X_i \leftarrow_{\$} \mathfrak{R}, \sigma) : \\ \top \leftarrow_{\$} \text{qry}^{Id_{\mathfrak{R}}}(X \leftarrow_{\$} \mathfrak{R}, \sigma) \end{array} \right].$$

Remark 1. Definitions 4 and 5 are equivalent to that of [13, Def. 3.4] for F -decomposable AMQ-PDS. Sampling n distinct elements from \mathfrak{D} is equivalent to sampling n strings $X \leftarrow_{\$} \mathfrak{R}$. Similarly, sampling the queried element from $\mathfrak{D} \setminus V$, where V is the set of n inserted elements, is equivalent to sampling $X \leftarrow_{\$} \mathfrak{R}$.

As mentioned, the NAI state constructed in Definition 4 captures honesty for insertion-only AMQ-PDS. As long as the effect of every insertion on the state is unpredictable, the final state cannot deviate from “honest”. However, for AMQ-PDS that also allow deletions, defining an honest setting is more involved. The deletion capability means that a user could insert elements, observe their effects, and then decide whether to delete them, i.e. to reset the state if not satisfied. In other words, even if every insertion is unpredictable, the final state may still be adversarially influenced (i.e. no longer an NAI state).

We overcome these issues with a new definition of the non-adversarial setting for function-decomposable, reinsertion-invariant AMQ-PDS, which we call *NAI**. *NAI** captures honesty up to the extent that can be achieved with both insertions and deletions. We will show that the final state of the AMQ-PDS satisfying *NAI**

suffices to capture a non-adversarial setting that we can analyse using results from the PDS literature.

A key component of NAI* will be the following notion: for any element not previously inserted, the effect of its insertion on the state is unpredictable (*insertion unpredictability*). Intuitively, this can be thought of as replacing every insertion of an element $x \in \mathfrak{D}$ with $X \in \mathfrak{X}$ sampled uniformly at random. This is not necessarily ensured only by F -decomposability, since the interplay between `ins`, `del` and `qry` on the same input could reveal information about F . We define insertion unpredictability in Definition 6.

Definition 6 (Insertion unpredictability). *Let Π be an AMQ-PDS with public parameters pp , using a random function $F : \mathfrak{D} \leftarrow \mathfrak{X}$, and satisfying F -decomposability and reinsertion invariance. Let $\sigma \leftarrow \text{setup}(pp)$. Let $\{z_i\}$ be the elements that are successfully inserted into σ . For every first insertion of z_i , let $(\top, \sigma') \leftarrow \text{ins}^F(z_i, \sigma_i)$ and $(\top, \bar{\sigma}) \leftarrow \text{ins}^{Id_{\mathfrak{X}}}(X \leftarrow_s \mathfrak{X}, \sigma_i)$. We say σ has insertion unpredictability if $SD(\sigma', \bar{\sigma}) = 0$.*

We are now ready to define an n -NAI* state. Although an NAI* state of an AMQ-PDS can be constructed through both insertions and deletions of elements, our definition will require that all insertions are unpredictable, deletions only happen on currently inserted elements, and repeated insertions of elements only change the state if that element has been deleted. These requirements essentially capture what we would expect from honest insertions and deletions on function-decomposable, reinsertion-invariant AMQ-PDS.

Definition 7 (n -NAI* state). *Let Π be an AMQ-PDS with public parameters pp using $F = Id_{\mathfrak{X}}$ satisfying reinsertion invariance, and let $\sigma \leftarrow \text{setup}(pp)$. Let n be a non-negative integer. Let $X_1, \dots, X_n \leftarrow_s \mathfrak{X}$. Let L be the list of operations on σ , where each item in L is either $\text{ins}^{Id_{\mathfrak{X}}}(\cdot, \sigma)$ or $\text{del}^{Id_{\mathfrak{X}}}(\cdot, \sigma)$ on X_1, \dots, X_n . Then, σ is an n -NAI* state if:*

- for all X_i there is an operation in L equal to $\text{ins}^{Id_{\mathfrak{X}}}(X_i, \sigma)$,
- for all successful $\text{del}^{Id_{\mathfrak{X}}}(X_i, \sigma)$ operations in L , the preceding successful operation in L on X_i is $\text{ins}^{Id_{\mathfrak{X}}}(X_i, \sigma)$,
- all successful $\text{ins}^{Id_{\mathfrak{X}}}(X_i, \sigma)$ operations in L for which any prior successful operation in L on X_i is $\text{ins}^{Id_{\mathfrak{X}}}(X_i, \sigma)$ either do not change the state, or have $\text{del}^{Id_{\mathfrak{X}}}(X_i, \sigma)$ as their preceding successful operation on X_i in L .

It is clear to see that every n -NAI state (Definition 4) is then an n -NAI* state. We now give an analogous formulation of Definition 7 for F -decomposable AMQ-PDS, where unsuccessful insertions do not change the state.

Corollary 1. *Let Π be an AMQ-PDS with public parameters pp using a random function $F : \mathfrak{D} \rightarrow \mathfrak{X}$ satisfying F -decomposability and reinsertion invariance, where unsuccessful insertions do not change the state. Let $\sigma \leftarrow \text{setup}(pp)$ and n be a non-negative integer. Let L be the list of operations on σ , where each item in L is either $\text{ins}^F(\cdot, \sigma)$ or $\text{del}^F(\cdot, \sigma)$. Then, σ is an n -NAI* state if:*

- it satisfies Definition 6,
- there are n distinct elements $\{z_i\}_{i \in [n]}$ for which an operation in L on z_i is $\text{ins}^F(z_i, \sigma)$,
- for all successful $\text{del}^F(z_i, \sigma)$ operations in L , the preceding successful operation in L on z_i is $\text{ins}^F(z_i, \sigma)$, and
- all $\text{ins}^F(z_i, \sigma)$ operations in L for which any prior successful operation in L on z_i is $\text{ins}^F(z_i, \sigma)$ either do not change the state, or have $\text{del}^F(z_i, \sigma)$ as their preceding successful operation on z_i in L .

A natural next step would be to define the false positive probability and insertion failure probability for NAI* states, analogous to that of the insertion-only setting [13]. However, while deleting an inserted element may be an operation allowed under NAI*, a user could insert elements, observe their effects, and then decide whether to delete them, i.e. to reset the state. This means that, using only n distinct elements, a user can create many different NAI* states. Therefore, a more useful notion for NAI* states is the *maximal* false positive and insertion failure probability, defined in terms of the “worst possible” NAI* state.

Definition 8 (Maximal NAI* false positive probability). *Let Π be an AMQ-PDS with public parameters pp using a random function $F : \mathfrak{D} \rightarrow \mathfrak{R}$ satisfying F -decomposability and reinsertion invariance. Let n be a non-negative integer. Define the maximal NAI* false positive probability after n insertions as*

$$P_{\Pi, pp}^*(FP | n) := \Pr \left[\begin{array}{l} X_1, \dots, X_n \leftarrow \mathfrak{R} \\ \sigma \leftarrow \mathcal{U}_{\Pi, pp}^{\text{Id}_{\mathfrak{R}}} (X_1, \dots, X_n) : \\ \top \leftarrow \mathfrak{qry}^{\text{Id}_{\mathfrak{R}}} (X \leftarrow \mathfrak{R}, \sigma) \end{array} \right],$$

where $\mathcal{U}_{\Pi, pp}^{\text{Id}_{\mathfrak{R}}} (X_1, \dots, X_n)$ outputs an NAI* state created using $\text{ins}^{\text{Id}_{\mathfrak{R}}}(\cdot, \sigma)$, $\text{del}^{\text{Id}_{\mathfrak{R}}}(\cdot, \sigma)$ on X_1, \dots, X_n that has the maximal false positive probability.

Definition 8 captures the false positive probability of the “worst possible” NAI* state that can be created with insertions and deletions. The algorithm \mathcal{U} gets n strings sampled uniformly at random from \mathfrak{R} as input, and finds the ordering of insertions and deletions of these strings (possibly excluding some) that maximises the false positive probability. Since the queried $X \leftarrow \mathfrak{R}$ is sampled randomly, \mathcal{U} is not increasing the probability that a particular element is a false positive; rather, it is creating a state with the highest false positive probability in general.

Definition 9 (Maximal NAI* insertion failure probability). *Let Π be an AMQ-PDS with public parameters pp , using a random function $F : \mathfrak{D} \rightarrow \mathfrak{R}$ satisfying F -decomposability and reinsertion invariance. Let n be a non-negative integer. Define the maximal NAI* insertion failure probability within n insertions as*

$$P_{\Pi, pp}^*(IF | n) := \Pr \left[\begin{array}{l} X_1, \dots, X_n \leftarrow \mathfrak{R} \\ \sigma \leftarrow \mathcal{V}_{\Pi, pp}^{\text{Id}_{\mathfrak{R}}} (X_1, \dots, X_n) : \\ \text{for some } l \in [n], (\perp, \sigma) \leftarrow \text{ins}^{\text{Id}_{\mathfrak{R}}} (X_l, \sigma) \end{array} \right],$$

where $\mathcal{V}_{\Pi,pp}^{Id_{\mathfrak{R}}}(X_1, \dots, X_n)$ outputs an NAI* state created using $ins^{Id_{\mathfrak{R}}}(\cdot, \sigma)$, $del^{Id_{\mathfrak{R}}}(\cdot, \sigma)$ on X_1, \dots, X_n that has the maximal probability of an insertion on one of X_1, \dots, X_n failing.

Definition 9 captures the insertion failure probability of the “worst possible” NAI* state that can be created with insertions and deletions. The algorithm \mathcal{V} gets as an input n strings sampled uniformly at random from \mathfrak{R} , and then finds the ordering of insertions and deletions of these strings (possibly excluding some) that maximises the probability that inserting one of these strings will fail. Note that the definition is of a slightly different flavour to Definition 8; \mathcal{V} can optimise its output in respect to X_l that is most likely to result in an insertion failure. Definition 9 naturally extends upon insertion failure definitions found in the literature [10, 12], where the probability is defined as one among n insertions failing.

3.3 Counting Filters

Counting filters are an extension of the popular Bloom filters, with the added capability of supporting deletions of elements. A Counting filter consists of an array of counters σ of length m initially set to 0^m , and a family of k independent hash functions $H_i : \{0, 1\}^* \rightarrow [m]$, for $i \in [k]$. To insert an element x into the filter, all k counters $H_i(x)$ of σ are incremented; if any counter reaches the maximum value $maxVal$, the insertion fails. To delete an element x from the filter, if all k counters $H_i(x)$ are greater than zero, they are all decremented; if not, the deletion fails. A membership query on x returns \top if all k counters $H_i(x)$ are greater than zero. Due to collisions in the hash functions H_i , Counting filters can have both false positives and false negatives. As in [13], we will bundle the k hash functions H_i into a single function $F : \mathfrak{D} \rightarrow [m]^k$.

We now formally define Counting filters.

Definition 10. *Let $m, k, maxVal$ be positive integers. We define an $(m, k, maxVal)$ -Counting filter to be the AMQ-PDS with algorithms defined in Fig. 2, with $pp = (m, k, maxVal)$, and $F : \mathfrak{D} \rightarrow [m]^k$.*

We recall from the literature a bound on the NAI false positive probability for Counting filters. Due to their membership query algorithm only checking for non-zero counters, as in the case of Bloom filters, this bound is the same for both Counting and Bloom filters.

Lemma 1 ([12],[13, Lemma 3.7]). *Let Π be an $(m, k, maxVal)$ -Counting filter using a random function $F : \mathfrak{D} \rightarrow [m]^k$. Then, for any n , $P_{\Pi,pp}(FP | n) \leq [1 - e^{-\frac{(n+0.5)k}{m-1}}]^k$.*

We now derive upper bounds on the maximal NAI* false positive probability and the maximal NAI* insertion failure probability for Counting filters.

Lemma 2. *Let Π be an $(m, k, maxVal)$ -Counting filter using a random function $F : \mathfrak{D} \rightarrow [m]^k$. Then, for any n , $P_{\Pi,pp}^*(FP | n) \leq [1 - e^{-\frac{(n+0.5)k}{m-1}}]^k$.*

setup(pp)	ins ^F (x, σ)	del ^F (x, σ)
1 $m, k, maxVal \leftarrow pp$	1 $(p_1, \dots, p_k) \leftarrow F(x)$	1 $(p_1, \dots, p_k) \leftarrow F(x)$
2 $\sigma \leftarrow 0^m$	2 $a \leftarrow \top$	2 for $i \in [k]$
3 return σ	3 for $i \in [k]$	3 if $\sigma[p_i] = 0$
qry ^F (x, σ)	4 if $\sigma[p_i] = 0$	4 return \perp, σ
	5 $a \leftarrow \perp$	5 for $i \in [k] : \sigma[p_i] - = 1$
1 $(p_1, \dots, p_k) \leftarrow F(x)$	6 if $a = \top : \text{return } \top, \sigma$	6 return \top, σ
2 for $i \in [k]$	7 for $i \in [k]$	
3 if $\sigma[p_i] = 0$	8 if $\sigma[p_i] \geq maxVal$	
4 return \perp	9 return \perp, σ	
5 return \top	10 for $i \in [k] : \sigma[p_i] + = 1$	
	11 return \top, σ	

Fig. 2. AMQ-PDS syntax instantiation for the Counting filter.

Proof (sketch). We construct an algorithm that inserts all X_1, \dots, X_n with $maxVal$ set to ∞ , and show that the false positive probability of the resulting state (which follows from Lemma 1) is an upper bound on the false positive probability of any n -NAI* state. For the full proof, see the full version.

Lemma 3. *Let Π be an $(m, k, maxVal)$ -Counting filter using a random function $F : \mathfrak{D} \rightarrow [m]^k$. Then, for any n , $P_{\Pi, pp}^*(IF | n) \leq m \cdot \left[\frac{e \cdot n \cdot k}{maxVal \cdot m} \right]^{maxVal}$.*

Proof (sketch). We construct an algorithm that inserts all X_1, \dots, X_n , using a modified insertion algorithm that always increments counters (i.e. the check in line 6 of the ins algorithm in Fig. 2 is skipped), and with $maxVal$ set to ∞ . Let the resulting state be denoted by Δ . We show that the insertion failure probability of any n -NAI* state with $maxVal$ equal to some *limit* can be upper bounded by the probability that any counter in Δ exceeds *limit*. For the full proof, see the full version.

3.4 Cuckoo Filters

Cuckoo filters were proposed as an alternative to Bloom filters with improved performance and support for deletions [10]. A Cuckoo filter consists of a collection $(\sigma_i)_i$ of 2^{λ_I} buckets, each indexed by $i \in [2^{\lambda_I}]$ and containing s slots, together with a stash σ_{stash} containing one slot. They use two hash functions $H_I : \mathfrak{D} \rightarrow \{0, 1\}^{\lambda_I}$ and $H_T : \mathfrak{D} \rightarrow \{0, 1\}^{\lambda_T}$. To insert (resp. delete) an element x into the filter, its tag is computed as $H_T(x)$ and inserted (resp. deleted) into its first or second bucket, whose indices are computed as $i_1 = H_I(x), i_2 = i_1 \oplus H_I(H_T(x))$ respectively. If both buckets are full, an eviction process begins. A membership

query on x returns \top if $H_T(x)$ is found in either of its corresponding buckets or the stash. As for Counting filters, membership queries can return both false positive and false negative responses.

In [13], a variant of the standard Cuckoo filter called the *PRF-wrapped Cuckoo filter* was proposed, which was required for the proofs of adversarial correctness and privacy. In this variant, inputs to the `ins`, `del` and `qry` algorithms are simply preprocessed with a random function $F : \mathfrak{D} \rightarrow \mathfrak{R}$, resulting in a function-decomposable filter that remains easy to implement, while satisfying the desired properties. For this reason, our work will also make use of PRF-wrapped Cuckoo filters, which we formally define below.

Definition 11. Let $pp = (s, \lambda_I, \lambda_T, num)$ be a tuple of positive integers. We define an $(s, \lambda_I, \lambda_T, num)$ -PRF-wrapped Cuckoo filter to be the AMQ-PDS with algorithms defined in Appendix A, with $pp = (s, \lambda_I, \lambda_T, num)$, making use of hash functions $H_T : \mathfrak{D} \rightarrow \{0, 1\}^{\lambda_T}$ and $H_I : \mathfrak{D} \rightarrow \{0, 1\}^{\lambda_I}$.

Our next step is to derive upper bounds on the NAI* false positive probability and the NAI* insertion failure probability for PRF-wrapped Cuckoo filters.

Lemma 4. Let Π be an $(s, \lambda_I, \lambda_T, num)$ -PRF-wrapped Cuckoo filter using random functions $F : \mathfrak{D} \rightarrow \mathfrak{R}$, $H_T : \mathfrak{D} \rightarrow \{0, 1\}^{\lambda_T}$, and $H_I : \mathfrak{D} \rightarrow \{0, 1\}^{\lambda_I}$. Then, for any n , $P_{\Pi, pp}^*(FP | n) \leq 1 - (1 - 2^{-\lambda_T})^{2^{s+1}} + \frac{n}{|\mathfrak{R}|}$.

Proof (sketch). We demonstrate that, apart from the collision probability in the range of F between the queried element and those used to create the state, the false positive probability bound in [10] upper bounds the probability for any n -NAI* state. For the full proof, see the full version.

Lemma 5. Let Π be an $(s, \lambda_I, \lambda_T, num)$ -PRF-wrapped Cuckoo filter using random functions $F : \mathfrak{D} \rightarrow \mathfrak{R}$, $H_T : \mathfrak{D} \rightarrow \{0, 1\}^{\lambda_T}$, and $H_I : \mathfrak{D} \rightarrow \{0, 1\}^{\lambda_I}$. Then, for any n ,

$$P_{\Pi, pp}^*(IF | n) \leq \frac{2}{(|\mathfrak{R}| \cdot 2^{\lambda_T + \lambda_I - 1})^{s-1}} \binom{n}{s} \prod_{i=1}^{s-1} [(|\mathfrak{R}| - i)(2^{\lambda_T} - i)].$$

Proof (sketch). We construct an algorithm that inserts all X_1, \dots, X_n , using a modified insertion algorithm where an element's tag is added to both of its buckets (if they do not already contain it), and with s set to ∞ . Let the resulting state be denoted by Δ . We show that the insertion failure probability of any n -NAI* state with s equal to some *limit* can be upper bounded by the probability that the load of any bucket in Δ exceeds *limit*. For the full proof, see the full version.

3.5 Consistency Rules

In this work, we will consider AMQ-PDS that satisfy some properties that we refer to as consistency rules, specified below. These rules are satisfied by Counting and Cuckoo filters.

Definition 12 (AMQ-PDS consistency rules). Consider an AMQ-PDS Π . We say Π has

- Successful deletion of positives if for all $x \in \mathfrak{D}, \sigma \in \Sigma, \top \leftarrow \text{qry}(x, \sigma) \implies (\top, \sigma') \leftarrow \text{del}(x, \sigma)$.
- Unsuccessful deletion of negatives if for all $x \in \mathfrak{D}, \sigma \in \Sigma, \perp \leftarrow \text{qry}(x, \sigma) \implies (\perp, \sigma) \leftarrow \text{del}(x, \sigma)$.
- Unsuccessful operation invariance if for all $x \in \mathfrak{D}, \sigma \in \Sigma, (\perp, \sigma') \leftarrow \text{ins}(x, \sigma) \implies \sigma' = \sigma$ and $(\perp, \sigma') \leftarrow \text{del}(x, \sigma) \implies \sigma' = \sigma$.

In the insertion-only setting, AMQ-PDS satisfy an additional consistency rule: for all $x \in \mathfrak{D}, \sigma \in \Sigma, (\top, \sigma) \leftarrow \text{ins}(x, \sigma) \implies \top \leftarrow \text{qry}(x, \sigma)$. In other words, a membership query on an inserted element will always return \top , meaning that σ has no false negative elements. In a setting with both insertions and deletions, one might expect the same rule to hold as long as x has not yet been deleted from σ . In Definition 13, we define σ having no false negatives more precisely.

Definition 13 (No false negatives). Let Π be an AMQ-PDS with public parameters pp satisfying reinsertion invariance, and let $\sigma \leftarrow \text{setup}(pp)$. Let $\{z_i\}$ be the elements that are successfully inserted into σ . Let L_i be the list of successful operations on z_i , where each item in L_i is either $\text{ins}(z_i, \sigma)$ or $\text{del}(z_i, \sigma)$. We say σ has no false negatives if, for all z_i , if the last item in L_i is $\text{ins}(z_i, \sigma)$, then $\top \leftarrow \text{qry}(z_i, \sigma)$.

Unfortunately, with deletions, we cannot say that σ contains no false negatives. They can arise as a result of inserting or deleting false positive elements, as we will see later. Therefore, Definition 13 is not satisfied by AMQ-PDS in general, and we will not require this from the AMQ-PDS we consider. Instead, we will analyse false negatives in our security proofs by using their relationship to false positives.

4 Adversarial Correctness

In this section, we analyse the correctness of AMQ-PDS under adversarial inputs. Our starting point is the simulation-based security definition for adversarial correctness in [13]. However, while their focus was on AMQ-PDS that only support insertions and membership queries, we are now interested in a more complex scenario with insertions, membership queries *and* deletions. As we will see, this increase in adversarial power requires tackling some new obstacles.

We derive bounds on the correctness of AMQ-PDS that satisfy function-decomposability, reinsertion invariance, and the consistency rules in Definition 12. Then, we apply our results to analyse Counting filters instantiated using a PRF, and PRF-wrapped Cuckoo filters. In both cases, we provide concrete guarantees on their adversarial correctness.

In the following, we consider an adversary \mathcal{A} interacting with an AMQ-PDS Π through an API, which we model as three oracles: **Ins**, which inserts elements of its choice into Π , **Del**, which deletes elements of its choice from Π , and **Qry**, which responds to membership queries (i.e. whether x has been inserted into Π).

Real-or-Ideal($\mathcal{A}, \mathcal{S}, \mathcal{D}, pp$)	Oracle $\mathbf{Ins}(x)$
1 $d \leftarrow_{\mathcal{S}} \{0, 1\}$	1 $(b, \sigma) \leftarrow_{\mathcal{S}} \mathbf{ins}^F(x, \sigma)$
2 if $d = 0$ // <i>Real</i>	2 return b
3 $K \leftarrow_{\mathcal{S}} \mathcal{K}; F \leftarrow R_K$	Oracle $\mathbf{Del}(x)$
4 $\sigma \leftarrow \mathbf{setup}(pp)$	Oracle $\mathbf{Del}(x)$
5 $out \leftarrow_{\mathcal{S}} \mathcal{A}^{\mathbf{Ins}, \mathbf{Del}, \mathbf{Qry}}$	1 $(b, \sigma) \leftarrow_{\mathcal{S}} \mathbf{del}^F(x, \sigma)$
6 else // <i>Ideal</i>	2 return b
7 $out \leftarrow_{\mathcal{S}} \mathcal{S}(\mathcal{A}, pp)$	Oracle $\mathbf{Qry}(x)$
8 return $d' \leftarrow_{\mathcal{S}} \mathcal{D}(out)$	1 return $\mathbf{qry}^F(x, \sigma)$

Fig. 3. Correctness game for AMQ-PDS II.

4.1 Notions of Correctness

We employ a simulation-based approach to analysing the adversarial correctness of AMQ-PDS, which proceeds as follows. The adversary \mathcal{A} plays in either the “real” or “ideal” world. In the real world, \mathcal{A} interacts with a keyed AMQ-PDS II, through oracles allowing it to make insertions, deletions and membership queries on elements of its choice. In the ideal world, \mathcal{A} instead interacts with a simulator \mathcal{S} , constructed such that it provides an NAI* view of II to \mathcal{A} . (Note that this differs from the definition of adversarial correctness in [13], which required \mathcal{S} to provide an NAI view.)

\mathcal{A} then produces some arbitrary output, which the distinguisher \mathcal{D} uses to compute which world \mathcal{A} was operating in. Finally, we bound \mathcal{D} ’s ability to distinguish between the two worlds. This allows us to quantify \mathcal{A} ’s probability of achieving any adversarial goal in the real world (through adaptive insertions, deletions and membership queries) by relating it to the ideal world, which we know how to analyse.

In Fig. 3, we define the Real-or-Ideal game.

Definition 14. *Let II be an AMQ-PDS with public parameters pp , and let R_K be a keyed function family. We say II is $(q_{ins}, q_{qry}, q_{del}, t_a, t_d, t_s, \varepsilon)$ -adversarially correct if, for all adversaries \mathcal{A} running in time at most t_a and making $q_{ins}, q_{qry}, q_{del}$ queries to oracles $\mathbf{Ins}, \mathbf{Qry}, \mathbf{Del}$ respectively in the Real-or-Ideal game (Fig. 3) with a simulator \mathcal{S} that provides an NAI* view of II to \mathcal{A} and runs in time at most t_s , and for all distinguishers \mathcal{D} running in time at most t_d , we have:*

$$Adv_{II, \mathcal{A}, \mathcal{S}}^{RoI}(\mathcal{D}) := |\Pr[\mathbf{Real}(\mathcal{A}, \mathcal{D})=1] - \Pr[\mathbf{Ideal}(\mathcal{A}, \mathcal{D}, \mathcal{S})=1]| \leq \varepsilon.$$

Remark 2. We discuss why Definition 14 captures adversarial correctness, by outlining how it can be used to analyse a specific adversarial goal. Consider an adversary \mathcal{A} that, throughout its execution, makes \mathbf{Ins} and \mathbf{Del} queries on adversarially selected inputs x_1, \dots, x_n , interspersed with \mathbf{Qry} queries, and

Simulator $\mathcal{S}(\mathcal{A}, pp)$	Oracle $\mathbf{DelSim}(x)$
1 $F \leftarrow \text{Funcs}[\mathcal{D}, \mathfrak{R}]$ 2 $\sigma \leftarrow \text{setup}(pp)$ 3 $\sigma^* \leftarrow \text{setup}(pp)$ 4 $\text{inserted} \leftarrow \{\}$ 5 $f \leftarrow \{\}$ 6 return $\mathcal{A}^{\text{InsSim}, \text{DelSim}, \text{QrySim}}$	1 $(b^{G^*}, \sigma^*) \leftarrow \text{del}^F(x, \sigma^*)$ 2 $b^{Ideal} \leftarrow \perp$ 3 if $\text{inserted}[x] = \top$ 4 $b^{Ideal} \leftarrow \top$ 5 $\text{inserted}[x] = \perp$ 6 $(-, \sigma) \leftarrow \text{del}^{\text{Id}_{\mathfrak{R}}}(f[x], \sigma)$ 7 return b^{Ideal} return b^{G^*}
Oracle $\mathbf{InsSim}(x)$	Oracle $\mathbf{QrySim}(x)$
1 $(c^{G^*}, \sigma^*) \leftarrow \text{ins}^F(x, \sigma^*)$ 2 $c^{Ideal} \leftarrow \top$ 3 // If it's not already inserted 4 if $\text{inserted}[x] = \perp$ 5 $f[x] \leftarrow \mathfrak{R}$ 6 $(c^{Ideal}, \sigma) \leftarrow \text{ins}^{\text{Id}_{\mathfrak{R}}}(f[x], \sigma)$ 7 // If the insertion succeeded 8 if $c^{Ideal} = \top$ 9 $\text{inserted}[x] \leftarrow \top$ 10 return c^{Ideal} return c^{G^*}	1 $a^{G^*} \leftarrow \text{qry}^F(x, \sigma^*)$ 2 $a^{Ideal} \leftarrow \top$ 3 // If it isn't inserted 4 if $\text{inserted}[x] = \perp$ 5 $a^{Ideal} \leftarrow \text{qry}^{\text{Id}_{\mathfrak{R}}}(X \leftarrow \mathfrak{R}, \sigma)$ 6 return a^{Ideal} return a^{G^*}

Fig. 4. Simulator and $\boxed{G^*}$ for AMQ-PDS with deletions.

ending with a final membership query $\mathbf{Qry}(x)$ with $x \leftarrow \mathcal{D} \setminus \{x_1, \dots, x_n\}$. Suppose the output of \mathcal{A} is the result of that final query, and \mathcal{D} 's output is identical to that of \mathcal{A} . Then, $\Pr[\text{Real}(\mathcal{A}, \mathcal{D})]$ is the adversarial false positive probability of Π produced by \mathcal{A} , for which we cannot directly compute an upper bound, since \mathcal{A} makes adaptive queries. However, $\Pr[\text{Ideal}(\mathcal{A}, \mathcal{D}, \mathcal{S})]$ is the NAI* false positive probability, for which we can derive upper bounds for our AMQ-PDS of interest. Then, if Definition 14 is satisfied, it means we can upper bound $\Pr[\text{Real}(\mathcal{A}, \mathcal{D})]$ by $\Pr[\text{Ideal}(\mathcal{A}, \mathcal{D}, \mathcal{S})] + \varepsilon$. Note that our definition covers any adversarial goal (see [13, Appendix C.2]).

In Fig. 4, we construct a simulator \mathcal{S} providing an NAI* view for function-decomposable AMQ-PDS supporting insertions, deletions and membership queries. We first observe that the state constructed by \mathcal{S} is always an NAI* state (Definition 7). Every insertion of element z_i either executes $\text{ins}^{\text{Id}_{\mathfrak{R}}}(\cdot, \sigma)$ on fresh $X_i \leftarrow \mathfrak{R}$ or does not change the state if on a currently inserted element. Moreover, only deletions of z_i that are currently inserted run $\text{del}^{\text{Id}_{\mathfrak{R}}}(X_i, \sigma)$. Further, by inspection, the runtime of \mathcal{S} is not significantly higher than that of the underlying AMQ-PDS.

Real-or- $G^*(\mathcal{A}, \mathcal{D}, pp)$	G^* -or-Ideal($\mathcal{A}, \mathcal{S}, \mathcal{D}, pp$)
1 $d \leftarrow_{\$} \{0, 1\}$	1 $d \leftarrow_{\$} \{0, 1\}$
2 if $d = 0$ // <i>Real</i>	2 if $d = 0$ // G^*
3 $K \leftarrow_{\$} \mathcal{K}; F \leftarrow R_K$	3 $F \leftarrow_{\$} \text{Funcs}[\mathfrak{D}, \mathfrak{R}]$
4 else // G^*	4 $\sigma \leftarrow \text{setup}(pp)$
5 $F \leftarrow_{\$} \text{Funcs}[\mathfrak{D}, \mathfrak{R}]$	5 $out \leftarrow_{\$} \mathcal{A}^{\text{Ins, Del, Qry}}$
6 $\sigma \leftarrow \text{setup}(pp)$	6 else // <i>Ideal</i>
7 $out \leftarrow_{\$} \mathcal{A}^{\text{Ins, Del, Qry}}$	7 $out \leftarrow_{\$} \mathcal{S}(\mathcal{A}, pp)$
8 return $d' \leftarrow_{\$} \mathcal{D}(out)$	8 return $d' \leftarrow_{\$} \mathcal{D}(out)$

Fig. 5. Intermediate game G^* for the proof of Theorem 1.

Theorem 1. Let $q_{ins}, q_{del}, q_{qry}$ be non-negative integers, and let $t_a, t_d > 0$. Let $F: \mathfrak{D} \rightarrow \mathfrak{R}$. Let Π be an AMQ-PDS with public parameters pp and oracle access to F , such that Π satisfies the consistency rules from Definition 12, F -decomposability (Definition 2), and reinsertion invariance (Definition 3). Let α, β, γ be the number of calls to F required to insert, query, delete an element respectively in Π using its *ins*, *qry*, *del* algorithms.

If $R_K: \mathfrak{D} \rightarrow \mathfrak{R}$ is an $(\alpha q_{ins} + \beta q_{qry} + \gamma q_{del}, t_a + t_d, \varepsilon)$ -secure pseudorandom function with key $K \leftarrow_{\$} \mathcal{K}$, then Π is $(q_{ins}, q_{qry}, q_{del}, t_a, t_s, t_d, \varepsilon')$ -adversarially correct with respect to the simulator in Fig. 4, where $t_s \approx t_a$ and $\varepsilon' = \varepsilon + 2P_{\Pi, pp}^*(IF | q_{ins}) + (q_{ins} + 2q_{qry} + q_{del}) \cdot P_{\Pi, pp}^*(FP | q_{ins})$.

Proof. We define an intermediate game G^* in Fig. 5. Let *Real* denote the $d = 0$ version of Real-or- G^* , let G^* denote the $d = 1$ version of Real-or- G^* (or equivalently, the $d = 0$ version of G^* -or-Ideal), and let *Ideal* denote the $d = 1$ version of G^* -or-Ideal. Then,

$$\begin{aligned}
\text{Adv}_{\Pi, \mathcal{A}, \mathcal{S}}^{\text{RoI}}(\mathcal{D}) &:= |\Pr[\text{Real}(\mathcal{A}, \mathcal{D})=1] - \Pr[\text{Ideal}(\mathcal{A}, \mathcal{D}, \mathcal{S})=1]| \\
&\leq |\Pr[\text{Real}(\mathcal{A}, \mathcal{D})=1] - \Pr[G^*(\mathcal{A}, \mathcal{D})=1]| \\
&\quad + |\Pr[G^*(\mathcal{A}, \mathcal{D})=1] - \Pr[\text{Ideal}(\mathcal{A}, \mathcal{D}, \mathcal{S})=1]|. \tag{1}
\end{aligned}$$

Our proof proceeds by bounding the closeness of *Real*, G^* in Lemma 6 in terms of the PRF advantage, and that of G^* , *Ideal* in Lemma 7 in terms of the probability of some “bad” event. Then, we combine these lemmas to obtain our result.

Lemma 6. The difference in probability of an arbitrary t_d -distinguisher \mathcal{D} outputting 1 in experiments of game Real-or- G^* in Fig. 5 with a $(q_{ins}, q_{qry}, q_{del}, t_a)$ -AMQ-PDS adversary \mathcal{A} is bounded by the maximal PRF advantage ε of an $(\alpha q_{ins} + \beta q_{qry} + \gamma q_{del}, t_a + t_d, \varepsilon)$ -PRF adversary attacking R_K :

$$\text{Adv}_{\Pi, \mathcal{A}, \mathcal{S}}^{\text{Real-or-}G^*}(\mathcal{D}) := |\Pr[\text{Real}(\mathcal{A}, \mathcal{D}) = 1] - \Pr[G^*(\mathcal{A}, \mathcal{D}) = 1]| \leq \varepsilon.$$

Proof. Consider the PRF adversary \mathcal{B} (Fig. 1), instantiating the AMQ-PDS queried by \mathcal{A} using its **RoR** oracle, in relation to the Real-or- G^* game (Fig. 5). When $b = 0$, \mathcal{B} is running *Real* for \mathcal{A} , and when $b = 1$, \mathcal{B} is instead running G^* for \mathcal{A} . Then, the advantage of \mathcal{B} is $\text{Adv}_R^{PRF}(\mathcal{B}) = \text{Adv}_{\Pi, \mathcal{A}, \mathcal{S}}^{\text{Real-or-}G^*}(\mathcal{D})$. Since R_K is an $(\alpha q_{\text{ins}} + \beta q_{\text{qry}} + \gamma q_{\text{del}}, t_a + t_d, \varepsilon)$ -secure PRF, $\text{Adv}_{\Pi, \mathcal{A}, \mathcal{S}}^{\text{Real-or-}G^*}(\mathcal{D}) \leq \varepsilon$.

Lemma 7. *The difference in probability of an arbitrary t_d -distinguisher \mathcal{D} outputting 1 in experiments of game G^* -or-Ideal in Fig. 5 with a $(q_{\text{ins}}, q_{\text{qry}}, q_{\text{del}}, t_a)$ -AMQ-PDS adversary \mathcal{A} is bounded as follows:*

$$\begin{aligned} \text{Adv}_{\Pi, \mathcal{A}, \mathcal{S}}^{G^*\text{-or-Ideal}}(\mathcal{D}) &:= |\Pr[G^*(\mathcal{A}, \mathcal{D}) = 1] - \Pr[\text{Ideal}(\mathcal{A}, \mathcal{D}, \mathcal{S}) = 1]| \\ &\leq 2P_{\Pi, pp}^*(IF | q_{\text{ins}}) + (q_{\text{ins}} + 2q_{\text{qry}} + q_{\text{del}}) \cdot P_{\Pi, pp}^*(FP | q_{\text{ins}}). \end{aligned}$$

Proof. We wish to bound the probability of distinguishing between G^* and *Ideal*. Let \mathbf{E} be the divergence event between G^* and *Ideal*, which occurs due to a mismatch in responses to **Qry**, **Del**, **Ins** queries across the two games (see Fig. 4).

First, we observe that *Ideal* cannot have false negative responses to membership queries, but G^* could have. If \mathcal{A} induces a false negative for some element x in G^* and then calls **Qry**(x), the two games would diverge with probability one. False negatives lead to repercussions when comparing responses to all types of queries across the two games. Therefore, we will deal with them separately, by defining \mathbf{E}_{FN} to be the event that a false negative occurs in G^* before any other query response mismatch. We then split the analysis of event \mathbf{E} into two parts: (1) the query response mismatch occurs without a false negative occurring in G^* beforehand (i.e. $\neg \mathbf{E}_{\text{FN}}$), or (2) the query response mismatch occurs with a false negative occurring in G^* beforehand (i.e. \mathbf{E}_{FN}). Then,

$$\Pr[\mathbf{E}] \leq \Pr[\mathbf{E} \wedge \neg \mathbf{E}_{\text{FN}}] + \Pr[\mathbf{E} \wedge \mathbf{E}_{\text{FN}}] \leq \Pr[\mathbf{E} \wedge \neg \mathbf{E}_{\text{FN}}] + \Pr[\mathbf{E}_{\text{FN}}].$$

We will analyse $\mathbf{E} \wedge \neg \mathbf{E}_{\text{FN}}$ for each query type separately. Let a_i^G, b_i^G, c_i^G denote the responses to \mathcal{A} 's i -th query, deletion, and insertion query in game $G \in \{G^*, \text{Ideal}\}$. Then, the games diverge the first time that $a_i^{G^*}, b_i^{G^*}$ or $c_i^{G^*}$ does not match $a_i^{\text{Ideal}}, b_i^{\text{Ideal}}$ or c_i^{Ideal} , respectively. We define

$$\mathbf{E}_{\text{Qry}} := \left[\left[\begin{array}{c} \text{The first query response mismatch is} \\ a_i^{\text{Ideal}} \neq a_i^{G^*} \text{ for some } i \in [q_{\text{qry}}] \end{array} \right] \wedge \neg \mathbf{E}_{\text{FN}} \right], \quad (2)$$

$$\mathbf{E}_{\text{Del}} := \left[\left[\begin{array}{c} \text{The first query response mismatch is} \\ b_i^{\text{Ideal}} \neq b_i^{G^*} \text{ for some } i \in [q_{\text{del}}] \end{array} \right] \wedge \neg \mathbf{E}_{\text{FN}} \right], \quad (3)$$

$$\mathbf{E}_{\text{Ins}} := \left[\left[\begin{array}{c} \text{The first query response mismatch is} \\ c_i^{\text{Ideal}} \neq c_i^{G^*} \text{ for some } i \in [q_{\text{ins}}] \end{array} \right] \wedge \neg \mathbf{E}_{\text{FN}} \right]. \quad (4)$$

Hence,

$$\Pr[\mathbf{E}] \leq \Pr[\mathbf{E}_{\text{FN}}] + \Pr[\mathbf{E}_{\text{Qry}}] + \Pr[\mathbf{E}_{\text{Del}}] + \Pr[\mathbf{E}_{\text{Ins}}]. \quad (5)$$

We now proceed to bound the probability of each event in Eq. (5). In the following, we take the probability over the randomness used by \mathcal{A} (which we refer to as \mathcal{A} 's coins), and the randomness used by game $G \in \{G^*, \text{Ideal}\}$ to answer \mathcal{A} 's queries (which we refer to as G 's coins). We will use x_i, y_i and z_i to denote the input to \mathcal{A} 's i -th query, deletion and insertion query, respectively.

Calculation of $\Pr[\mathbf{E}_{\text{FN}}]$. We start by analysing the probability of a false negative occurring in G^* . Our key observation is that false negatives can only occur from inserting or deleting false positives.

Consider an element x that is a false positive due to insertions of $\{z_1, \dots, z_\ell\}$, where $\ell \geq 1$. By the consistency rule *successful deletion of positives*, the deletion of x will succeed, although it was never inserted. However, this may cause elements in $\{z_1, \dots, z_\ell\}$ to become false negatives. Now, consider inserting this false positive x . By *reinsertion invariance*, the state will remain unchanged, but x will become a true positive. Then, deleting any element in $\{x, z_1, \dots, z_\ell\}$ will succeed, but may cause other elements in $\{x, z_1, \dots, z_\ell\}$ to become false negatives.

Recall that we are interested in analysing the probability of a false negative occurring in G^* *before* any other mismatch in query responses across the two games. Therefore, we do not need to consider deletions of false positives; it would result in a mismatch in **Del** responses, since *Ideal* does not allow deletions of false positives while G^* does. We will then focus solely on false negatives caused by insertions of false positives in the following. We write

$$\begin{aligned} \Pr[\mathbf{E}_{\text{FN}}] &:= \Pr \left[\begin{array}{l} \text{A false negative occurs in } G^* \\ \text{before a query response mismatch occurred} \end{array} \right] \\ &\leq \Pr \left[\begin{array}{l} \text{A false positive is inserted in } G^* \\ \text{before a query response mismatch occurred} \end{array} \right] \\ &\leq \sum_{i=1}^{q_{\text{ins}}} \Pr \left[\begin{array}{l} z_i \text{ is the first false positive inserted in } G^* \\ \text{before a query response mismatch occurred} \end{array} \right]. \end{aligned}$$

Let σ_i^* denote the state of Π in game G^* just before the i -th **Ins** query. Then, since no prior query response mismatch occurred and z_i is the first false positive inserted, σ_i^* contains no false negatives up to this point. Then,

$$\begin{aligned} \Pr[\mathbf{E}_{\text{FN}}] &\leq \sum_{i=1}^{q_{\text{ins}}} \Pr \left[\begin{array}{l} [z_i \text{ is a false positive in } \sigma_i^*] \wedge \\ [\sigma_i^* \text{ has no false negatives}] \wedge \\ [\text{no prior query response mismatch occurred}] \end{array} \right] \\ &\leq \sum_{i=1}^{q_{\text{ins}}} \Pr_{\substack{G^* \text{'s coins} \\ \text{Ideal's coins} \\ \mathcal{A}' \text{s coins}}} \left[\begin{array}{l} [\text{inserted}[z_i] = \perp] \wedge [\top \leftarrow * \text{qry}^F(z_i, \sigma_i^*)] \wedge \\ [\sigma_i^* \text{ has no false negatives}] \wedge \\ [\text{no prior query response mismatch occurred}] \end{array} \right]. \quad (6) \end{aligned}$$

Let L_i be the list of successful operations on σ^* in G^* up to the i -th **Ins** query, where each item in L_i is either $\text{ins}^F(\cdot, \sigma^*)$ or $\text{del}^F(\cdot, \sigma^*)$ on z_1, \dots, z_{i-1} . By the

consistency rule *unsuccessful operation invariance*, we do not need to consider unsuccessful operations when constructing σ_i^* . So,

$$\Pr[\mathbf{E}_{\text{FN}}] \leq \sum_{i=1}^{q_{\text{ins}}} \Pr_{\substack{G^*'s \text{ coins} \\ \text{Ideal's coins} \\ \mathcal{A}'s \text{ coins}}} \left[\begin{array}{l} \sigma_i^* \leftarrow \text{setup}(pp) \\ \text{for } \text{op}^F(z_j, \sigma^*) \in L_i : (\top, \sigma_i^*) \leftarrow \text{op}^F(z_j, \sigma_i^*) : \\ \quad [\text{inserted}[z_i] = \perp] \wedge [\top \leftarrow \text{qry}^F(z_i, \sigma_i^*)] \wedge \\ \quad [\sigma_i^* \text{ has no false negatives}] \wedge \\ \quad [\text{no prior query response mismatch occurred}] \end{array} \right]. \quad (7)$$

Now, if no prior query response mismatch has occurred, $\text{inserted}[z_i] = \perp$ implies that either z_i was never inserted into σ_i^* , or z_i was inserted but then deleted. In the latter scenario, since σ_i^* contains no false negatives up to this point (as per Eq. (7)), z_i must be a positive at the time of its deletion. Then, by the consistency rule *successful deletion of positives*, its deletion will succeed, thus fully undoing the effect of its insertion on σ_i^* . Since F is a random function satisfying F -decomposability and \mathcal{A} has no information about F , we write Eq. (7) as

$$\Pr[\mathbf{E}_{\text{FN}}] \leq \sum_{i=1}^{q_{\text{ins}}} \Pr_{\substack{G^*'s \text{ coins} \\ \text{Ideal's coins} \\ \mathcal{A}'s \text{ coins}}} \left[\begin{array}{l} \sigma_i^* \leftarrow \text{setup}(pp) \\ \text{for } \text{op}^F(z_j, \sigma^*) \in L_i : (\top, \sigma_i^*) \leftarrow \text{op}^{\text{Id}_{\mathfrak{R}}}(F(z_j), \sigma_i^*) : \\ \quad [\top \leftarrow \text{qry}^{\text{Id}_{\mathfrak{R}}}(X \leftarrow \mathfrak{R}, \sigma_i^*)] \wedge \\ \quad [\sigma_i^* \text{ has no false negatives}] \wedge \\ \quad [\text{no prior query response mismatch occurred}] \end{array} \right].$$

Now, since F is a random function and \mathcal{A} has no information about F , we can replace every first insertion of an element $F(z_j)$ by $X_{z_j} \leftarrow \mathfrak{R}$ (i.e. σ_i^* satisfies *insertion unpredictability*). For repeated insertions on an element, we have two possibilities. If this element has not been deleted since its last insertion, the repeated insertion will not change the state, due to *reinsertion invariance*. However, if it has been deleted since its last insertion, it will change the state in the same way as its first insertion, since both use the same F . Therefore, we can rewrite the above by sampling $|\{z_1, \dots, z_{i-1}\}|$ random strings, and associating each string to a distinct z_j , giving

$$\Pr[\mathbf{E}_{\text{FN}}] \leq \sum_{i=1}^{q_{\text{ins}}} \Pr_{\substack{\text{Ideal's coins} \\ \mathcal{A}'s \text{ coins}}} \left[\begin{array}{l} \ell \leftarrow |\{z_1, \dots, z_{i-1}\}| \\ \{u_1, \dots, u_\ell\} \leftarrow \{z_1, \dots, z_{i-1}\} \\ X_{u_1}, \dots, X_{u_\ell} \leftarrow \mathfrak{R} \\ \sigma_i^* \leftarrow \text{setup}(pp) \\ \text{for } \text{op}^F(z_j, \sigma^*) \in L_i : (\top, \sigma_i^*) \leftarrow \text{op}^{\text{Id}_{\mathfrak{R}}}(X_{z_j}, \sigma_i^*) : \\ \quad [\top \leftarrow \text{qry}^{\text{Id}_{\mathfrak{R}}}(X \leftarrow \mathfrak{R}, \sigma_i^*)] \wedge \\ \quad [\sigma_i^* \text{ has no false negatives}] \wedge \\ \quad [\text{no prior query response mismatch occurred}] \end{array} \right]. \quad (8)$$

We now argue that every σ_i^* is an n -NAI* state, where n is upper bounded by q_{ins} , by showing that it satisfies the requirements in Corollary 1. Firstly, observe that the construction of σ_i^* in Eq. (8) enforces insertion unpredictability (Definition 6). Secondly, there are at most q_{ins} insertions in σ_i^* . Thirdly, since no query response mismatch has yet occurred, all deletions must be on elements for

which the preceding successful operation was an insertion. Finally, since there are no false negatives up to this point and *reinsertion invariance* holds, any insertion on a currently inserted element will not change the state.

Let $\mathcal{U}_{\Pi,pp}^{\text{Id}_{\mathfrak{R}}}$ be the algorithm from Definition 8 that, given X_1, \dots, X_n , outputs an NAI* state created using $\text{ins}^{\text{Id}_{\mathfrak{R}}}$, $\text{del}^{\text{Id}_{\mathfrak{R}}}$ on X_1, \dots, X_n with the maximal false positive probability. Then, no matter how σ_i^* is created, the state output by $\mathcal{U}_{\Pi,pp}^{\text{Id}_{\mathfrak{R}}}$ will result in an equal or higher false positive probability than that of σ_i^* . Since $\ell \leq q_{\text{ins}}$ and with more distinct insertions, $\mathcal{U}_{\Pi,pp}^{\text{Id}_{\mathfrak{R}}}$ may be able to create a state with even higher false positive probability,

$$\Pr[\mathbf{E}_{\text{FN}}] \leq \sum_{i=1}^{q_{\text{ins}}} \Pr_{\text{Ideal's coins}} \left[\begin{array}{c} X_1, \dots, X_{q_{\text{ins}}} \leftarrow \mathfrak{R} \\ \sigma \leftarrow \mathcal{U}_{\Pi,pp}^{\text{Id}_{\mathfrak{R}}}(X_1, \dots, X_{q_{\text{ins}}}) : \\ \top \leftarrow \mathfrak{qry}^{\text{Id}_{\mathfrak{R}}}(X \leftarrow \mathfrak{R}, \sigma) \end{array} \right].$$

Finally, applying Definition 8, we obtain

$$\Pr[\mathbf{E}_{\text{FN}}] \leq q_{\text{ins}} \cdot P_{\Pi,pp}^*(FP | q_{\text{ins}}). \quad (9)$$

Calculation of $\Pr[\mathbf{E}_{\text{Qry}}]$. We first rewrite Eq. (2) using the union bound as

$$\Pr[\mathbf{E}_{\text{Qry}}] \leq \sum_{i=1}^{q_{\text{qry}}} \Pr \left[\left[\begin{array}{c} [\mathbf{Qry}(x_i) \text{ yields the first mismatch}] \wedge \\ [(a_i^{\text{Ideal}} = \top) \wedge (a_i^{G^*} = \perp)] \\ \vee [(a_i^{\text{Ideal}} = \perp) \wedge (a_i^{G^*} = \top)] \end{array} \right] \wedge \neg \mathbf{E}_{\text{FN}} \right]. \quad (10)$$

We start by inspecting the **Qry** algorithms of G^* and *Ideal* to see where they could diverge. In G^* , the responses to \mathcal{A} 's **Qry** queries are always computed using the same function F , while in *Ideal*, a fresh random string $X \leftarrow \mathfrak{R}$ is sampled each time a non-inserted element is queried.

Let σ_i denote the state of Π in game *Ideal* just before the i -th **Qry** query, and σ_i^* denote the corresponding state in game G^* . Since **Qry**(x_i) yields the *first* query response mismatch, both G^* and *Ideal* must contain the same set of inserted elements. As \mathbf{E}_{FN} did not yet occur, σ_i^* has no false negatives. Moreover, **Qry** queries in *Ideal* do not give false negative responses. This means that **Qry** queries on elements that were inserted (and not yet deleted) will always return a positive response in both games. Therefore, in order for x_i to yield a mismatch in **Qry** query responses between the games, we must have that x_i is not currently inserted in *Ideal* (i.e. $\text{inserted}[x_i] = \perp$ in line 4 of **QrySim**). This gives

$$\begin{aligned} \Pr[\mathbf{E}_{\text{Qry}}] &\leq \sum_{i=1}^{q_{\text{qry}}} \left[\Pr \left[\left[\begin{array}{c} [\mathbf{Qry}(x_i) \text{ yields the first mismatch}] \wedge \\ [\text{inserted}[x_i] = \perp] \wedge [a_i^{\text{Ideal}} = \top] \end{array} \right] \wedge \neg \mathbf{E}_{\text{FN}} \right] \right. \\ &\quad \left. + \Pr \left[\left[\begin{array}{c} [\mathbf{Qry}(x_i) \text{ yields the first mismatch}] \wedge \\ [\text{inserted}[x_i] = \perp] \wedge [a_i^{G^*} = \top] \end{array} \right] \wedge \neg \mathbf{E}_{\text{FN}} \right] \right] \quad (11) \end{aligned}$$

$$:= \sum_{i=1}^{q_{\text{qry}}} \left[\Pr[\mathbf{E}_{\text{Qry}}^{\text{Ideal}}] + \Pr[\mathbf{E}_{\text{Qry}}^{G^*}] \right], \quad (12)$$

where, for simplicity, we will use $\Pr [\mathbf{E}_{\mathbf{Qry}}^{Ideal}]$ to denote the first term of Eq. (11), and $\Pr [\mathbf{E}_{\mathbf{Qry}}^{G^*}]$ to denote the second term.

We start by bounding $\Pr [\mathbf{E}_{\mathbf{Qry}}^{Ideal}]$. In *Ideal*, a fresh random string $X \leftarrow_s \mathfrak{R}$ is sampled each time a non-inserted element is queried, and so

$$\Pr [\mathbf{E}_{\mathbf{Qry}}^{Ideal}] \leq \Pr_{\substack{Ideal's \text{ coins} \\ \mathcal{A}'s \text{ coins}}} \left[\left[\mathbf{Qry}(x_i) \text{ yields the first mismatch} \right] \wedge \left[\top \leftarrow_s \mathbf{qry}^{Id_{\mathfrak{R}}}(X \leftarrow_s \mathfrak{R}, \sigma_i) \right] \right].$$

We now argue that every σ_i is an n -NAI* state, with n being upper bounded by q_{ins} , by showing that it satisfies the requirements in Definition 7. Firstly, from line 3 of **DelSim**, we observe that only deletions of currently inserted elements run $\text{del}^{Id_{\mathfrak{R}}}(\cdot, \sigma)$, possibly changing the state. Secondly, we note that in **InsSim**, every insertion either executes $\text{ins}^{Id_{\mathfrak{R}}}(\cdot, \sigma)$ on $X_i \leftarrow_s \mathfrak{R}$, or does not change the state if it is on a currently inserted element. Therefore, σ_i is an NAI* state containing at most q_{ins} elements. Then, we can upper bound the false positive probability of σ_i by that of the NAI* state with the maximal false positive probability (Definition 8), giving $\Pr [\mathbf{E}_{\mathbf{Qry}}^{Ideal}] \leq P_{\Pi, pp}^*(FP | q_{\text{ins}})$.

We use a reasoning similar to calculating \mathbf{E}_{FN} to compute $\Pr [\mathbf{E}_{\mathbf{Qry}}^{G^*}]$, replacing z_i with x_i . Under $\neg \mathbf{E}_{FN}$, the state σ_i^* contains no false negatives. Therefore, we can apply Eq. (6) from the \mathbf{E}_{FN} calculation to get

$$\begin{aligned} \Pr [\mathbf{E}_{\mathbf{Qry}}^{G^*}] &= \Pr_{\substack{G^*'s \text{ coins} \\ Ideal's \text{ coins} \\ \mathcal{A}'s \text{ coins}}} \left[\left[\left[\mathbf{Qry}(x_i) \text{ yields the first mismatch} \right] \wedge \left[\text{inserted}[x_i] = \perp \right] \wedge \left[\top \leftarrow_s \mathbf{qry}^F(x_i, \sigma_i^*) \right] \right] \wedge \neg \mathbf{E}_{FN} \right] \\ &\leq \Pr_{\substack{G^*'s \text{ coins} \\ Ideal's \text{ coins} \\ \mathcal{A}'s \text{ coins}}} \left[\left[\text{inserted}[x_i] = \perp \right] \wedge \left[\top \leftarrow_s \mathbf{qry}^F(x_i, \sigma_i^*) \right] \wedge \left[\sigma_i^* \text{ has no false negatives} \right] \wedge \left[\text{no prior query response mismatch occurred} \right] \right] \\ &\leq P_{\Pi, pp}^*(FP | q_{\text{ins}}). \end{aligned} \quad (13)$$

Substituting $\Pr [\mathbf{E}_{\mathbf{Qry}}^{Ideal}]$, $\Pr [\mathbf{E}_{\mathbf{Qry}}^{G^*}]$ in Eq. (12) gives

$$\Pr [\mathbf{E}_{\mathbf{Qry}}] \leq \sum_{i=1}^{q_{\text{qry}}} 2P_{\Pi, pp}^*(FP | q_{\text{ins}}) = 2q_{\text{qry}} \cdot P_{\Pi, pp}^*(FP | q_{\text{ins}}). \quad (14)$$

Calculation of $\Pr [\mathbf{E}_{\text{Del}}]$. We first rewrite Eq. (3) using the union bound as

$$\Pr [\mathbf{E}_{\text{Del}}] \leq \sum_{i=1}^{q_{\text{del}}} \Pr \left[\left[\left[\mathbf{Del}(y_i) \text{ yields the first mismatch} \right] \wedge \left[\left[(b_i^{Ideal} = \top) \wedge (b_i^{G^*} = \perp) \right] \vee \left[(b_i^{Ideal} = \perp) \wedge (b_i^{G^*} = \top) \right] \right] \right] \wedge \neg \mathbf{E}_{FN} \right]. \quad (15)$$

We now examine the **Del** algorithms of G^* and *Ideal*. In G^* , the responses to \mathcal{A} 's **Del** queries are always computed using the same function F . In *Ideal*, deletions are only allowed on an element y_i if it is currently inserted in the filter, and use the same random string $f[y_i]$ that was used for y_i 's insertion.

We note that in Eq. (15), we are only interested in the case where $\mathbf{Del}(y_i)$ is the first query response mismatch. In this case, both G^* and $Ideal$ must contain the same set of inserted elements. As \mathbf{E}_{FN} did not occur, every inserted element is a true positive in both games. We observe that in $Ideal$, true positives are always successfully deleted (see line 3 of \mathbf{DelSim}), while in G^* , successful deletion of true positives is ensured by the consistency rule *successful deletion of positives*. However, by the same consistency rule, deletions of false positives also succeed in G^* , while they do not in $Ideal$. Consequently, deletions in G^* succeed on at least the elements on which they succeed in $Ideal$. Thus, it never happens that a deletion succeeds in $Ideal$ but not in G^* , and we can rewrite Eq. (15) as

$$\Pr[\mathbf{E}_{\mathbf{Del}}] \leq \sum_{i=1}^{q_{\mathbf{del}}} \Pr \left[\left[[\mathbf{Del}(y_i) \text{ yields the first mismatch}] \wedge \left[(b_i^{Ideal} = \perp) \wedge (b_i^{G^*} = \top) \right] \right] \wedge \neg \mathbf{E}_{FN} \right].$$

Let σ_i^* denote the state of Π in game G^* just before the i -th \mathbf{Del} query. By the consistency rule *unsuccessful deletion of negatives*,

$$\Pr[\mathbf{E}_{\mathbf{Del}}] \leq \sum_{i=1}^{q_{\mathbf{del}}} \Pr_{\substack{G^*'s \text{ coins} \\ Ideal's \text{ coins} \\ \mathcal{A}'s \text{ coins}}} \left[\left[[\mathbf{Del}(y_i) \text{ yields the first mismatch}] \wedge \left[\text{inserted}[y_i] = \perp \right] \wedge [\top \leftarrow^* \mathbf{qry}^F(y_i, \sigma_i^*)] \right] \wedge \neg \mathbf{E}_{FN} \right].$$

We use a reasoning similar to calculating \mathbf{E}_{FN} to compute this, replacing z_i with y_i . Under $\neg \mathbf{E}_{FN}$, the state σ_i^* contains no false negatives. Therefore, we can apply Eq. (6) from the \mathbf{E}_{FN} calculation to get

$$\begin{aligned} \Pr[\mathbf{E}_{\mathbf{Del}}] &\leq \sum_{i=1}^{q_{\mathbf{del}}} \Pr_{\substack{G^*'s \text{ coins} \\ Ideal's \text{ coins} \\ \mathcal{A}'s \text{ coins}}} \left[\begin{array}{l} [\text{inserted}[y_i] = \perp] \wedge [\top \leftarrow^* \mathbf{qry}^F(y_i, \sigma_i^*)] \wedge \\ [\sigma_i^* \text{ has no false negatives}] \wedge \\ [\text{no prior query response mismatch occurred}] \end{array} \right] \\ &\leq \sum_{i=1}^{q_{\mathbf{del}}} P_{\Pi, pp}^*(FP | q_{\text{ins}}) = q_{\mathbf{del}} \cdot P_{\Pi, pp}^*(FP | q_{\text{ins}}). \end{aligned} \quad (16)$$

Calculation of $\Pr[\mathbf{E}_{\mathbf{Ins}}]$. We first rewrite Eq. (4) as

$$\Pr[\mathbf{E}_{\mathbf{Ins}}] = \Pr \left[\left[\begin{array}{l} [\mathbf{Ins}(z_i) \text{ yields the first mismatch}] \wedge \\ \left[\left[(c_i^{Ideal} = \perp) \wedge (c_i^{G^*} = \top) \right] \vee \left[(c_i^{Ideal} = \top) \wedge (c_i^{G^*} = \perp) \right] \right] \right. \\ \left. \text{for some } i \in [q_{\text{ins}}] \right] \wedge \neg \mathbf{E}_{FN} \right]. \quad (17)$$

Let us now compare the \mathbf{Ins} algorithms of G^* and $Ideal$. In G^* , the responses to \mathcal{A} 's \mathbf{Ins} queries are always computed using the same function F . On the other hand, in $Ideal$, a fresh random string $f[z_i] \leftarrow^* \mathfrak{R}$ is sampled at each insertion of an element z_i which is not already inserted.

Let σ_i denote the state of Π in game $Ideal$ just before the i -th \mathbf{Ins} query, and σ_i^* denote the corresponding state in game G^* . If $\mathbf{Ins}(z_i)$ is the first mismatch, it must be that both G^* and $Ideal$ contain the same set of inserted elements up

to this point. In *Ideal*, by inspecting **InsSim** we observe that the insertion of any currently inserted element z_i will always succeed. In G^* , since we are only considering the case where \mathbf{E}_{FN} did not yet occur, σ_i^* has no false negatives. This means that any element z_i that was inserted and not yet deleted will result in $\top \leftarrow \text{qry}^F(z_i, \sigma_i^*)$. Then, by *reinsertion invariance*, the insertion of z_i will succeed (but not change the state) in G^* . Therefore, for the first query response mismatch it must be that x_i is not currently inserted in *Ideal* (i.e. $\text{inserted}[z_i] = \perp$ in line 4 of **InsSim**). Then,

$$\Pr[\mathbf{E}_{\text{Ins}}] \leq \Pr \left[\left[\begin{array}{l} [\mathbf{Ins}(z_i) \text{ yields the first mismatch}] \wedge \\ [c_i^{\text{Ideal}} = \perp] \wedge [\text{inserted}[z_i] = \perp] \\ \text{for some } i \in [q_{\text{ins}}] \end{array} \right] \wedge \neg \mathbf{E}_{FN} \right] \\ + \Pr \left[\left[\begin{array}{l} [\mathbf{Ins}(z_i) \text{ yields the first mismatch}] \wedge \\ [c_i^{G^*} = \perp] \wedge [\text{inserted}[z_i] = \perp] \\ \text{for some } i \in [q_{\text{ins}}] \end{array} \right] \wedge \neg \mathbf{E}_{FN} \right] \quad (18)$$

$$:= \Pr[\mathbf{E}_{\text{Ins}}^{\text{Ideal}}] + \Pr[\mathbf{E}_{\text{Ins}}^{G^*}], \quad (19)$$

where, for simplicity, we will use $\Pr[\mathbf{E}_{\text{Ins}}^{\text{Ideal}}]$ to denote the first term of Eq. (18), and $\Pr[\mathbf{E}_{\text{Ins}}^{G^*}]$ to denote the second term.

We start by computing $\Pr[\mathbf{E}_{\text{Ins}}^{\text{Ideal}}]$. In *Ideal*, a fresh random string $X \leftarrow_{\$} \mathfrak{R}$ is sampled each time a non-inserted element is queried, and so we can write

$$\Pr[\mathbf{E}_{\text{Ins}}^{\text{Ideal}}] \leq \Pr_{\substack{\text{Ideal's coins} \\ \mathcal{A}'\text{'s coins}}} \left[\left[\begin{array}{l} [\mathbf{Ins}(z_i) \text{ yields the first mismatch}] \wedge \\ [(\perp, \sigma_i) \leftarrow_{\$} \text{ins}^{\text{Id}_{\mathfrak{R}}}(X \leftarrow_{\$} \mathfrak{R}, \sigma_i)] \\ \text{for some } i \in [q_{\text{ins}}] \end{array} \right] \right].$$

Since every σ_i is an n -NAI* state, with n being upper bounded by q_{ins} , we can upper bound the insertion failure probability of σ_i by that of the NAI* state with the maximal insertion failure probability (Definition 9), giving $\Pr[\mathbf{E}_{\text{Ins}}^{\text{Ideal}}] \leq P_{\Pi, \text{pp}}^*(IF | q_{\text{ins}})$.

We now compute $\Pr[\mathbf{E}_{\text{Ins}}^{G^*}]$. We have that

$$\Pr[\mathbf{E}_{\text{Ins}}^{G^*}] \leq \Pr_{\substack{G^*\text{'s coins} \\ \text{Ideal's coins} \\ \mathcal{A}'\text{'s coins}}} \left[\left[\begin{array}{l} [\mathbf{Ins}(z_i) \text{ yields the first mismatch}] \wedge \\ [(\perp, \sigma_i^*) \leftarrow_{\$} \text{ins}^F(z_i, \sigma_i^*)] \\ \text{for some } i \in [q_{\text{ins}}] \end{array} \right] \wedge \neg \mathbf{E}_{FN} \right].$$

Let L_i be the list of successful operations on σ^* in G^* up to the i -th **Ins** query, where each item in L_i is either $\text{ins}^F(\cdot, \sigma^*)$ or $\text{del}^F(\cdot, \sigma^*)$ on z_1, \dots, z_{i-1} . Recall that we do not need to consider unsuccessful operations when constructing σ_i^* , by the consistency rule *unsuccessful operation invariance*. Then,

$$\begin{aligned}
\Pr \left[\mathbf{E}_{\mathbf{Ins}}^{G^*} \right] &\leq \Pr_{\substack{G^*'s \text{ coins} \\ \mathcal{I}deal's \text{ coins} \\ \mathcal{A}'s \text{ coins}}} \left[\begin{array}{l} \sigma_i^* \leftarrow \text{setup}(pp) \\ \text{for } \text{op}^F(z_j, \sigma^*) \in L_i : (\top, \sigma_i^*) \leftarrow \text{op}^F(z_j, \sigma_i^*) : \\ \text{[inserted}[z_i] = \perp] \wedge [(\perp, \sigma_i^*) \leftarrow \text{s } \text{ins}^F(z_i, \sigma_i^*)] \wedge \\ \text{[}\sigma_i^* \text{ has no false negatives]} \wedge \\ \text{[}\mathbf{Ins}(z_i) \text{ yields the first mismatch]} \\ \text{for some } i \in [q_{\text{ins}}] \end{array} \right] \\
&= \Pr_{\substack{G^*'s \text{ coins} \\ \mathcal{I}deal's \text{ coins} \\ \mathcal{A}'s \text{ coins}}} \left[\begin{array}{l} \sigma_i^* \leftarrow \text{setup}(pp) \\ \text{for } \text{op}^F(z_j, \sigma^*) \in L_i : (\top, \sigma_i^*) \leftarrow \text{op}^{\text{Id}\mathfrak{R}}(F(z_j), \sigma_i^*) : \\ \text{[inserted}[z_i] = \perp] \wedge [(\perp, \sigma_i^*) \leftarrow \text{s } \text{ins}^{\text{Id}\mathfrak{R}}(F(z_i), \sigma_i^*)] \wedge \\ \text{[}\sigma_i^* \text{ has no false negatives]} \wedge \\ \text{[}\mathbf{Ins}(z_i) \text{ yields the first mismatch]} \\ \text{for some } i \in [q_{\text{ins}}] \end{array} \right],
\end{aligned}$$

by F -decomposability. Now, since F is a random function and \mathcal{A} has no information about F , we can then proceed in a similar manner as in the \mathbf{E}_{FN} calculation, with the caveat that we are now interested in *any* of the q_{ins} insertions failing:

$$\Pr[\mathbf{E}_{\mathbf{Ins}}^{G^*}] \leq \Pr_{\substack{\mathcal{I}deal's \text{ coins} \\ \mathcal{A}'s \text{ coins}}} \left[\begin{array}{l} \ell \leftarrow |\{z_1, \dots, z_{q_{\text{ins}}}\}| \\ \{u_1, \dots, u_\ell\} \leftarrow \{z_1, \dots, z_{q_{\text{ins}}}\} \\ X_{u_1}, \dots, X_{u_\ell} \leftarrow \mathfrak{R} \\ \text{for some } i \in [q_{\text{ins}}], \\ \sigma_i^* \leftarrow \text{setup}(pp) \\ \text{for } \text{op}^F(z_j, \sigma^*) \in L_i : (\top, \sigma_i^*) \leftarrow \text{op}^{\text{Id}\mathfrak{R}}(X_{z_j}, \sigma_i^*) : \\ [(\perp, \sigma_i^*) \leftarrow \text{s } \text{ins}^{\text{Id}\mathfrak{R}}(X_{z_i} \leftarrow \mathfrak{R}, \sigma_i^*)] \wedge \\ \text{[}\sigma_i^* \text{ has no false negatives]} \wedge \\ \text{[}\mathbf{Ins}(z_i) \text{ yields the first mismatch]} \end{array} \right]. \quad (20)$$

Similarly as in Eq. (8), we conclude that σ_i^* in Eq. (20) is an NAI* state. Then, we again upper bound the insertion failure probability of σ_i^* by that of the NAI* state with the maximal insertion failure probability (Definition 9), giving $\Pr[\mathbf{E}_{\mathbf{Ins}}^{G^*}] \leq P_{\Pi, pp}^*(IF | q_{\text{ins}})$. Substituting $\Pr[\mathbf{E}_{\mathbf{Ins}}^{\mathcal{I}deal}]$, $\Pr[\mathbf{E}_{\mathbf{Ins}}^{G^*}]$ in Eq. (19), we obtain

$$\Pr[\mathbf{E}_{\mathbf{Ins}}] \leq 2P_{\Pi, pp}^*(IF | q_{\text{ins}}). \quad (21)$$

Finally, substituting Eqs. (9, 14, 16, 21) in Eq. (5), we have

$$\text{Adv}_{\Pi, \mathcal{A}, \mathcal{S}}^{G^* \text{-or-}\mathcal{I}deal}(\mathcal{D}) \leq 2P_{\Pi, pp}^*(IF | q_{\text{ins}}) + (q_{\text{ins}} + 2q_{\text{qry}} + q_{\text{del}}) \cdot P_{\Pi, pp}^*(FP | q_{\text{ins}}).$$

To prove Theorem 1, we then apply Lemmas 6 and 7 to Eq. (1) to obtain

$$\text{Adv}_{\Pi, \mathcal{A}, \mathcal{S}}^{\text{RoI}}(\mathcal{D}) \leq \varepsilon + 2P_{\Pi, pp}^*(IF | q_{\text{ins}}) + (q_{\text{ins}} + 2q_{\text{qry}} + q_{\text{del}}) \cdot P_{\Pi, pp}^*(FP | q_{\text{ins}}).$$

Remark 3. We discuss why our results do not directly extend to a setting where \mathcal{A} can access the internal state σ . In Fig. 4, observe that, upon reinsertion of an element not currently in the filter, $\mathcal{I}deal$ always samples a fresh $X \leftarrow \mathfrak{R}$, while G^* inserts the same element again. This choice allowed us to obtain distinguishing bounds involving only the NAI* false positive and insertion failure probabilities. However, this difference is clearly detectable if \mathcal{A} can view σ after reinsertion, leading to $\mathcal{I}deal$ and G^* being distinguished with a probability close to 1.

4.2 Guarantees for Counting and Cuckoo Filters

In this section, we will use Theorem 1 to give concrete correctness guarantees for Counting and Cuckoo filters.

Corollary 2. *Let $q_{ins}, q_{del}, q_{qry}$ be non-negative integers, and let $t_a, t_d > 0$. Let $F: \mathfrak{D} \rightarrow \mathfrak{R}$. Let Π be a Counting filter AMQ-PDS with public parameters pp and oracle access to F . If R_K for $K \leftarrow_s \mathcal{K}$ is a $(q_{ins} + q_{qry} + q_{del}, t_a + t_d, \varepsilon)$ -secure pseudorandom function and $F = R_K$, then Π is $(q_{ins}, q_{qry}, q_{del}, t_a, t_s, t_d, \varepsilon')$ -adversarially correct, where $t_s \approx t_a$ and $\varepsilon' = \varepsilon + 2m \cdot \left[\frac{e \cdot q_{ins} \cdot k}{\maxVal \cdot m} \right]^{\maxVal} + (q_{ins} + 2q_{qry} + q_{del}) \cdot \left[1 - e^{-\frac{(q_{ins} + 0.5)k}{m-1}} \right]^k$.*

Proof. From the `ins`, `del`, `qry` algorithms in Fig. 2, we see that Counting filters with oracle access to a random function F are F -decomposable, reinsertion invariant, and satisfy the consistency rules in Def. 12. Further, each `ins`, `del` and `qry` call contains one call to the function F . Then, Theorem 1 holds with $\alpha = \beta = \gamma = 1$. Using Lemmas 2 and 3, we obtain the result.

Remark 4. The adversarial correctness bound for Bloom filters in [13, Corollary 4.4] holds for insertion-only Counting filters.

Corollary 3. *Let $q_{ins}, q_{del}, q_{qry}$ be non-negative integers, and let $t_a, t_d > 0$. Let $F: \mathfrak{D} \rightarrow \mathfrak{R}$. Let Π be a PRF-wrapped Cuckoo filter AMQ-PDS with public parameters pp and oracle access to F . If R_K for $K \leftarrow_s \mathcal{K}$ is a $(q_{ins} + q_{qry} + q_{del}, t_a + t_d, \varepsilon)$ -secure pseudorandom function and $F = R_K$, then Π is $(q_{ins}, q_{qry}, q_{del}, t_a, t_s, t_d, \varepsilon')$ -adversarially correct, where $t_s \approx t_a$ and*

$$\begin{aligned} \varepsilon' = \varepsilon + & \frac{4}{\left(|\mathfrak{R}| \cdot 2^{\lambda_T + \lambda_I - 1} \right)^{s-1}} \binom{q_{ins}}{s} \prod_{i=1}^{s-1} [(|\mathfrak{R}| - i)(2^{\lambda_T} - i)] \\ & + (q_{ins} + 2q_{qry} + q_{del}) \cdot \left[1 - (1 - 2^{-\lambda_T})^{2s+1} + \frac{q_{ins}}{|\mathfrak{R}|} \right]. \end{aligned}$$

Proof. From the `ins`, `del`, `qry` algorithms in Appendix A, we see that PRF-wrapped Cuckoo filters with oracle access to a random function F are F -decomposable, reinsertion invariant, and satisfy the consistency rules in Definition 12. Further, each `ins`, `del` and `qry` call contains one call to the function F . Then, Theorem 1 holds with $\alpha = \beta = \gamma = 1$. Using Lemmas 4 and 5, we obtain the result.

5 Secure Instances

In this section, we outline how our results can be used to secure AMQ-PDS in practice. Let us consider the example outlined in Remark 2, with the predicate $P := [\mathcal{A}'\text{'s final } \mathbf{Qry}(x) \text{ query on } x \leftarrow_s \mathfrak{D} \setminus \{x_1, \dots, x_n\} \text{ returns } \top]$.

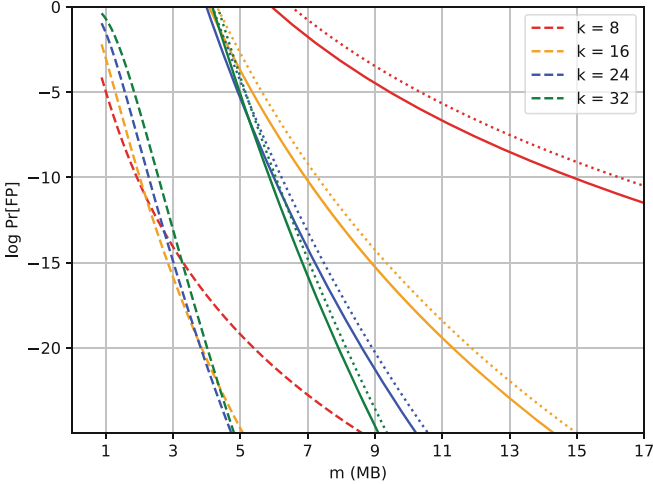


Fig. 6. Correctness guarantees vs. storage trade-offs for Counting filters with $\text{maxVal} = 16$, $q_{\text{ins}} = q_{\text{del}} = q_{\text{qry}} = 2^{20}$, $\varepsilon = 2^{-128}$. Dashed lines represent non-adversarial guarantees (Lemma 2), solid lines represent adversarial guarantees for the insertion-only setting ([13, Corollary 4.4]), and dotted lines represent adversarial guarantees for the setting with insertions and deletions (Corollary 2).

Since Theorem 1 holds for any predicate, the probability of an adversary \mathcal{A} satisfying P in the real world is given by $\Pr[\mathcal{D}(\mathcal{A})=1] \leq \varepsilon + 2P_{\Pi,pp}^*(IF | q_{\text{ins}}) + (q_{\text{ins}} + 2q_{\text{qry}} + q_{\text{del}} + 1) \cdot P_{\Pi,pp}^*(FP | q_{\text{ins}})$.

We illustrate the behaviour of this bound for the example of Counting filters. In Fig. 6, we plot an upper bound of the false positive probability against the size of the Counting filter in three settings: the non-adversarial setting, the insertion-only adversarial setting, and the setting with deletions studied in this work. By Remark 4, we can analyse the insertion-only setting using the results in [13]: $\Pr[\mathcal{D}(\mathcal{A})=1] \leq \varepsilon + (2q_{\text{qry}} + 1) \cdot P_{\Pi,pp}(FP | q_{\text{ins}})$.

From Fig. 6, we observe that guaranteeing a specific false positive probability even in an adversarial setting with deletions requires roughly trebling the size of the filter, when compared to the honest (NAI) setting. Crucially, deletions do not incur a significant cost when compared to the insertion-only setting; the additional term of $P_{\Pi,pp}^*(IF | q_{\text{ins}})$ can be made very small with the choice of an appropriate maxVal . For Cuckoo filters, the same observation holds for the choice of λ_I and λ_T . Hence, moving to the more complex scenario of allowing deletions does not hinder the practicality of our results.

A Cuckoo Filters

In Fig. 7, we give the AMQ-PDS syntax instantiation for PRF-wrapped Cuckoo filters. Following the reference implementation [11] by the authors of [10], after we delete an element, we try to empty the stash by re-inserting the stashed element. We write the procedure *evict* separately for ease of understanding.

setup(pp)	ins $^{F, H_T, H_I}(x, \sigma)$	del $^{F, H_T, H_I}(x, \sigma)$
1 $s, \lambda_I, \lambda_T, num \leftarrow pp$	1 $x \leftarrow F(x)$	1 $x \leftarrow F(x)$
2 // initialise 2^{λ_I} buckets	2 $tag \leftarrow H_T(x)$	2 $tag \leftarrow H_T(x)$
3 // and s λ_T -bit slots	3 $i_1 \leftarrow H_I(x)$	3 $i_1 \leftarrow H_I(x)$
4 for $i \in [2^{\lambda_I}]$	4 $i_2 \leftarrow i_1 \oplus H_I(tag)$	4 $i_2 \leftarrow i_1 \oplus H_I(tag)$
5 $\sigma_i \leftarrow \perp^s$	5 $a \leftarrow (tag = \sigma_{stash})$	5 // tag in stash?
6 // stashed element bucket	6 $a \leftarrow a \wedge (k_{stash} \in \{i_1, i_2\})$	6 if $[tag = \sigma_{stash}]$
7 $k_{stash} \leftarrow \perp$	7 $a \leftarrow a \vee tag \in \sigma_{i_1} \vee tag \in \sigma_{i_2}$	7 if $(k_{stash} \in \{i_1, i_2\})$
8 // stashed element tag	8 if $a = \top$: return \top, σ	8 $\sigma_{stash} \leftarrow \perp$
9 $\sigma_{stash} \leftarrow \perp$	9 // ins disabled?	9 $k_{stash} \leftarrow \perp$
10 $\sigma \leftarrow (\sigma_i)_{i \in [2^{\lambda_I}]}, \sigma_{stash}, k_{stash}$	10 if $\sigma_{stash} \neq \perp$: return \perp, σ	10 return \top, σ
11 return σ	11 // check if empty slots	11 // tag in bucket?
qry $^{F, H_T, H_I}(x, \sigma)$	12 for $i \in \{i_1, i_2\}$	12 for $i \in \{i_1, i_2\}$
1 $x \leftarrow F(x)$	13 if $load(\sigma_i) < s$	13 if $tag \in \sigma_i$
2 $tag \leftarrow H_T(x)$	14 $\sigma_i \leftarrow \sigma_i \diamond tag$	14 $\sigma_i \leftarrow \sigma_i \diamond tag$
3 $i_1 \leftarrow H_I(x)$	15 return \top, σ	15 // empty the stash
4 $i_2 \leftarrow i_1 \oplus H_I(tag)$	16 // displace something	16 if $\sigma_{stash} \neq \perp$
5 // tag in stash?	17 $i \leftarrow^s \{i_1, i_2\}$	17 $j_1 \leftarrow k_{stash}$
6 $a \leftarrow (tag = \sigma_{stash})$	18 $\sigma \leftarrow evict^{H_I}(i, tag, \sigma)$	18 $j_2 \leftarrow j_1 \oplus H_I(\sigma_{stash})$
7 $a \leftarrow a \wedge (k_{stash} \in \{i_1, i_2\})$	19 return \top, σ	19 for $j \in \{j_1, j_2\}$
8 // tag in bucket?	evict $^{H_I}(i, tag, \sigma)$	20 if $load(\sigma_j) < s$
9 $a \leftarrow a \vee tag \in \sigma_{i_1} \vee tag \in \sigma_{i_2}$	1 for $g \in [num]$	21 $\sigma_j \leftarrow \sigma_j \diamond \sigma_{stash}$
10 return a	2 $slot \leftarrow^s [s]$	22 $\sigma_{stash} \leftarrow \perp$
	3 $elem \leftarrow \sigma_i[slot]$	23 $k_{stash} \leftarrow \perp$
	4 // swap elem, tag	24 // displace something
	5 $\sigma_i[slot] \leftarrow tag; tag \leftarrow elem$	25 if $\sigma_{stash} \neq \perp$
	6 $i \leftarrow i \oplus H_I(tag)$	26 $j \leftarrow^s \{j_1, j_2\}$
	7 if $load(\sigma_i) < s$	27 $\sigma \leftarrow evict^{H_I}(j, \sigma_{stash}, \sigma)$
	8 $\sigma_i \leftarrow \sigma_i \diamond tag$	28 return \top, σ
	9 return \top, σ	29 // nothing to delete
	10 $k_{stash} \leftarrow i$	30 return \perp, σ
	11 $\sigma_{stash} \leftarrow tag$	
	12 return σ	

Fig. 7. AMQ-PDS syntax instantiation for the PRF-wrapped Cuckoo filter.

References

1. Bloom filters and cuckoo filters for cache summarization. <https://blog.fleek.network/post/bloom-and-cuckoo-filters-for-cache-summarization/>.
2. Redisbloom: Probabilistic data structures for redis. <https://redis.com/modules/redis-bloom/>.
3. Ziv Bar-Yossef, T. S. Jayram, Ravi Kumar, D. Sivakumar, and Luca Trevisan. Counting distinct elements in a data stream. In *International Workshop on Randomization and Approximation Techniques in Computer Science*, 2002. <https://doi.org/10.1007/3-540-45726-7.1>.

4. Michael A. Bender, Martin Farach-Colton, Mayank Goswami, Rob Johnson, Samuel McCauley, and Shikha Singh. Bloom filters, adaptivity, and the dictionary problem. In *FOCS*, 2018. <https://doi.org/10.1109/FOCS.2018.00026>.
5. Burton H. Bloom. Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM*, 13(7):422–426, 1970. <https://doi.org/10.1145/362686.362692>.
6. Andrei Z. Broder and Michael Mitzenmacher. Survey: Network applications of Bloom filters: A survey. *Internet Mathematics*, 1(4):485–509, 2003. <https://doi.org/10.1080/15427951.2004.10129096>.
7. Yan-Cheng Chang and Michael Mitzenmacher. Privacy preserving keyword searches on remote encrypted data. In *Applied Cryptography and Network Security*, 2005. https://doi.org/10.1007/11496137_30.
8. David Clayton, Christopher Patton, and Thomas Shrimpton. Probabilistic data structures in adversarial environments. In *ACM SIGSAC CCS*, 2019. <https://doi.org/10.1145/3319535.3354235>.
9. Graham Cormode and S. Muthukrishnan. An improved data stream summary: the count-min sketch and its applications. *Journal of Algorithms*, 55(1):58–75, 2005. <https://doi.org/10.1016/j.jalgor.2003.12.001>.
10. Bin Fan, Dave G. Andersen, Michael Kaminsky, and Michael D. Mitzenmacher. Cuckoo filter: Practically better than Bloom. In *CoNEXT*, 2014. <https://doi.org/10.1145/2674005.2674994>.
11. Bin Fan, David G. Andersen, and Michael Kaminsky. Cuckoo filter reference implementation. <https://github.com/efficient/cuckoofilter/blob/917583d6abef692dfa8e14453bd77d6e0b61eef3/src/cuckoofilter.h#L139>, 2013.
12. Li Fan, Pei Cao, J. Almeida, and A.Z. Broder. Summary cache: a scalable wide-area web cache sharing protocol. *IEEE/ACM Transactions on Networking*, 8(3):281–293, 2000. <https://doi.org/10.1109/90.851975>.
13. Mia Filić, Kenny Paterson, Anupama Unnikrishnan, and Fernando Virdia. Adversarial correctness and privacy for probabilistic data structures. In *ACM SIGSAC CCS*, 2022. <https://doi.org/10.1145/3548606.3560621>.
14. Philippe Flajolet, Eric Fusy, Olivier Gandouet, and Frédéric Meunier. Hyperloglog: the analysis of a near-optimal cardinality estimation algorithm. In *Conference on Analysis of Algorithms*, 2007. <https://doi.org/10.46298/dmtcs.3545>.
15. Sergio Galán, Pedro Reviriego, Stefan Walzer, Alfonso Sánchez-Macian, Shanshan Liu, and Fabrizio Lombardi. On the privacy of counting bloom filters under a black-box attacker. *IEEE Transactions on Dependable and Secure Computing*, 20(5), 2023. <https://doi.org/10.1109/TDSC.2022.3217115>.
16. Thomas Gerbet, Amrit Kumar, and Cédric Lauradoux. The power of evil choices in Bloom filters. In *IEEE/IFIP Conference on Dependable Systems and Networks*, 2015. <https://doi.org/10.1109/DSN.2015.21>.
17. Junzhi Gong, Tong Yang, Haowei Zhang, Hao Li, Steve Uhlig, Shigang Chen, Lorna Uden, and Xiaoming Li. HeavyKeeper: An accurate algorithm for finding top-k elephant flows. In *USENIX Annual Technical Conference*, 2018. <https://doi.org/10.1109/TNET.2019.2933868>.
18. Laura Hetz, Thomas Schneider, and Christian Weinert. Scaling mobile private contact discovery to billions of users. In *ESORICS*, 2023. https://doi.org/10.1007/978-3-031-50594-2_23.
19. Stefan Heule, Marc Nunkesser, and Alexander Hall. Hyperloglog in practice: Algorithmic engineering of a state of the art cardinality estimation algorithm. In *Conference on Extending Database Technology*, 2013. <https://doi.org/10.1145/2452376.2452456>.

20. Daniel Kales, Christian Rechberger, Thomas Schneider, Matthias Senker, and Christian Weinert. Mobile private contact discovery at scale. In *USENIX Security*, 2019.
21. Tsvi Kopelowitz, Samuel McCauley, and Ely Porat. Support optimality and adaptive cuckoo filters. In *Algorithms and Data Structures*, 2021. https://doi.org/10.1007/978-3-030-83508-8_40.
22. Anukool Lakhina, Mark Crovella, and Christiphe Diot. Characterization of network-wide anomalies in traffic flows. In *ACM SIGCOMM Conference on Internet Measurement*, 2004. <https://doi.org/10.1145/1028788.1028813>.
23. James Larisch, David Choffnes, Dave Levin, Bruce M Maggs, Alan Mislove, and Christo Wilson. Crlite: A scalable system for pushing all tls revocations to all browsers. In *IEEE S&P*, 2017. <https://doi.org/10.1109/SP.2017.17>.
24. Yehuda Lindell. How to simulate it – a tutorial on the simulation proof technique, 2017. https://doi.org/10.1007/978-3-319-57048-8_6.
25. Linsheng Liu, Daniel S. Roche, Austin Theriault, and Arkady Yerukhimovich. Fighting fake news in encrypted messaging with the fuzzy anonymous complaint tally system (facts). In *Network and Distributed Systems Security Symposium*, 2022. <https://doi.org/10.14722/ndss.2022.23109>.
26. Lailong Luo, Deke Guo, Richard T. B. Ma, Ori Rottenstreich, and Xueshan Luo. Optimizing bloom filter: Challenges, solutions, and comparisons. *IEEE Communications Surveys & Tutorials*, 21(2):1912–1949, 2019. <https://doi.org/10.1109/COMST.2018.2889329>.
27. Sam A. Markelon, Mia Filić, and Thomas Shrimpton. Compact frequency estimators in adversarial environments. In *ACM SIGSAC CCS*, 2023. <https://doi.org/10.1145/3576915.3623216>.
28. Luca Melis, George Danezis, and Emiliano De Cristofaro. Efficient private statistics with succinct sketches. In *Network and Distributed Systems Security Symposium*, 2016. <https://doi.org/10.14722/ndss.2016.23175>.
29. Páll Melsted and Jonathan K Pritchard. Efficient counting of k-mers in dna sequences using a bloom filter. *BMC Bioinformatics*, 12, 2011. <https://doi.org/10.1186/1471-2105-12-333>.
30. Moni Naor and Noa Oved. Bet-or-pass: Adversarially robust bloom filters. In *TCC*, 2022. https://doi.org/10.1007/978-3-031-22365-5_27.
31. Moni Naor and Eylon Yogev. Bloom filters in adversarial environments. In *CRYPTO*, 2015. https://doi.org/10.1007/978-3-662-48000-7_28.
32. Moni Naor and Eylon Yogev. Bloom filters in adversarial environments. *ACM Transactions on Algorithms*, 15(3):35:1–35:30, 2019. <https://doi.org/10.1145/3306193>.
33. Kenneth G. Paterson and Mathilde Raynal. HyperLogLog: Exponentially bad in adversarial settings. In *EuroS&P*, 2022. <https://doi.org/10.1109/EuroSP53844.2022.00018>.
34. Henning Perl, Yassene Mohammed, Michael Brenner, and Matthew Smith. Fast confidential search for bio-medical data using bloom filters and homomorphic cryptography. *IEEE Conference on E-Science*, pages 1–8, 2012. <https://doi.org/10.1109/eScience.2012.6404484>.
35. Xiaofeng Shi, Shouqian Shi, Minmei Wang, Jonne Kaunisto, and Chen Qian. On-device iot certificate revocation checking with small memory and low latency. In *ACM SIGSAC CCS*, 2021. <https://doi.org/10.1145/3460120.3484580>.
36. Dimitrios Sikeridis, Sean Huntley, David Ott, and Michael Devetsikiotis. Intermediate certificate suppression in post-quantum tls: An approximate membership querying approach. In *CoNEXT*, 2022. <https://doi.org/10.1145/3555050.3569127>.

37. Henrik Stranneheim, Max Käller, Tobias Allander, Björn Andersson, Lars Arvestad, and Joakim Lundeberg. Classification of dna sequences using bloom filters. *Bioinformatics*, 26(13):1595–1600, 2010. <https://doi.org/10.1093/bioinformatics/btq230>.
38. Jeff Yan and Pook Leong Cho. Enhancing collaborative spam detection with bloom filters. In *Annual Computer Security Applications Conference*, 2006. <https://doi.org/10.1109/ACSAC.2006.26>.
39. Kevin Yeo. Cuckoo hashing in cryptography: Optimal parameters, robustness and applications. In *CRYPTO*, 2023. <https://doi.org/10.1007/978-3-031-38551-3-7>.



Concurrent Encrypted Multimaps

Archita Agarwal^(✉), Seny Kamara, and Tarik Moataz

MongoDB, New York, NY 10019, USA

{archita_agarwal, seny_kamara, tarik_moataz}@mongodb.com

Abstract. Encrypted data structures have received a lot of attention due to their use as building blocks in the design of fast encrypted search algorithms and encrypted databases. An important design aspect that, as far as we know, has not been considered is that modern server architectures are concurrent in the sense that they support the execution of multiple operations simultaneously. In this work, we initiate the study of concurrent encrypted data structures. We identify new definitional and technical challenges posed by concurrency in the setting of encrypted search. In order to formalize the security of these schemes, we extend the standard framework of structured encryption to capture, among other things, fine-grained leakage which occurs at the instruction level as well as schedule-dependent leakage which changes as a function of the order in which instructions are executed. The latter is particularly challenging to handle when the scheduler is adversarial and adaptive. We provide security definitions in the ideal/real-world model which allows us to capture both security and consistency together.

We combine techniques from structured encryption and concurrent data structures to design the first concurrent encrypted multi-map. We show that it is not only secure and efficient, but also satisfies a strong consistency guarantee called *linearizability* while supporting *lock-free* append operations and requiring no inter-client communication.

Keywords: Encrypted search · Concurrent data structures · Concurrent encrypted data structures

1 Introduction

Encrypted multi-maps (EMM) are end-to-end encrypted data structures that store label/value pairs and support get and put operations in sub-linear time. EMMs are a core building block in the design of sub-linear encrypted databases and searchable symmetric encryption (SSE) schemes. As encrypted databases gain popularity and interest from industry, new problems at the intersection of cryptography and distributed systems are emerging. One example is the problem of designing encrypted *distributed* data structures studied in [1, 2]. These are encrypted data structures designed to be stored and managed by clusters of machines as opposed to a single server as is traditionally considered in the encrypted search literature. Encrypted distributed structures are a crucial

building block for the design of real-world encrypted databases since, in practice, most databases are distributed and run on clusters.

Another practical problem that, as far as we know, has received very little attention is the problem of designing *concurrent* encrypted structures by which we mean dynamic encrypted structures that can be accessed concurrently while providing strong consistency guarantees and high throughput. This is a fundamental problem because every real-world database system is concurrent. More precisely, the literature in encrypted search usually models database server executions as sequential in the sense that the server is assumed to execute operations in their entirety one after the other. For example, upon receiving a sequence of operations $(\text{op}_1, \dots, \text{op}_n)$ from a client it executes op_i entirely before executing op_{i+1} . Even in multi-client settings, the server is assumed to order the operations of the clients and then execute them fully in that order.

Concurrency and Consistency. In reality, database servers do not execute operations sequentially. They use multi-threading and often multiple cores to execute many operations simultaneously. This increases operation throughput since the server can make progress on an operation op_i while waiting for an expensive call from operation op_j to return (e.g., a call to disk). Database servers do not view operations as atomic objects that must be executed in their entirety at once but, instead, as sequences of lower-level atomic instructions that can be context-switched at any moment by the operating system scheduler. Concurrency introduces a host of challenges, the most important of which is that the traditional notions of correctness are not meaningful. Consider a multi-map that stores a label/tuple pair (ℓ, \mathbf{v}) , where $\mathbf{v} = (v_1, \dots, v_{10})$, an append operation (ℓ, v_{11}) and a get operation on ℓ . Now suppose the append and get operations are concurrent. Should we require the get operation to return (v_1, \dots, v_{10}) or (v_1, \dots, v_{11}) ? Either answer is acceptable depending on exactly how the lower-level instructions of the two operations are scheduled. Because of this, correctness in concurrent settings is replaced with the notion of *consistency* which guarantees that the outputs of the operations are consistent with *some* sequential order of the operations; even if their instructions are actually interleaved. Returning to the example, if the get outputs (v_1, \dots, v_{11}) then its output is consistent with the sequential order (append, get), whereas if it outputs (v_1, \dots, v_{10}) its output is consistent with the sequential order (get, append).

Many different notions of consistency have been defined and studied. Some are weaker in the sense that they allow for more sequential orders and some are stronger in the sense that they allow for fewer. For example, linearizability [35] is a very strong consistency notion which, roughly speaking, guarantees that the output of the operations preserve their real-time ordering, i.e., if op_i completes before op_j begins, then any effect of op_i will be reflected in op_j 's output. In our example above, linearizability guarantees that if the append finishes before the get starts, then the get's output will include the value added by the append. In contrast, the weaker notion of sequential consistency [48], would allow the get to output either (v_1, \dots, v_{10}) or (v_1, \dots, v_{11}) as long as the get and append were made by different clients.

Concurrent EMMs. Based on the discussion above, it should be clear that real-world encrypted databases and their underlying data structures need to support concurrent executions and guarantee some notion of consistency. As far as we know, the only semi-dynamic or dynamic EMM construction that can achieve high throughput and some form of consistency under concurrent operations is the OST scheme of [42] which underlies MongoDB’s Queryable Encryption. A limitation of the scheme, however, is that the scheme itself is only designed to provide high throughput while consistency is achieved at the implementation level by making use of database transactions. This has two implications. The first is that the scheme itself is not provably consistent. The second is that the way transactions are used may not necessarily lead to optimal throughput. The design of concurrent EMMs is highly non-trivial and to see why we will go over various possible solutions and explain why they do not work.

The first solution one might think of is to start with a dynamic EMM and modify it using standard techniques from the concurrency literature. As an example, one might start with the following simplified version of the π_{bas} construction from [15]. At a high level it works by breaking down a label/tuple pair (ℓ, \mathbf{v}) , where $\mathbf{v} = (v_1, \dots, v_n)$, into n pairs where the i th pair is of the form $(\ell || i, v_i)$. Each pair is then stored as $(F(K_{\ell,1}, i), \text{Enc}(K_{\ell,2}, v_i))$ in a standard/plaintext dictionary, where F is a pseudo-random function, Enc is a symmetric encryption scheme, and $K_{\ell,1}$ and $K_{\ell,2}$ are label-specific keys computed using a pseudo-random function. To execute a get operation for ℓ , the client sends $K_{\ell,1}$ to the server who uses it as follows. It initializes a counter i to 1 and computes $\text{tag}_i := F(K_{\ell,1}, i)$. It then queries the dictionary on tag_i . If the tag is in the dictionary, the server returns the associated ciphertext, increments the counter and repeats the process. If, on the other hand, tag_i is not in the dictionary it stops. To add a value v_{n+1} to ℓ ’s tuple, the client sends the pair $(F(K_{\ell,1}, n+1), \text{Enc}(K_{\ell,2}, v_{n+1}))$. Notice that in order to guarantee correctness, the client has to keep local counters for every label (client state is needed by most semi-dynamic and dynamic EMMs that achieve standard notions of security).

Naive Server-Side Synchronization. To achieve consistency with a multi-client stateful dynamic EMM, the clients need to synchronize on their state. This is clear in the case of π_{bas} and to see why consider a setting where two clients \mathbf{C}_i and \mathbf{C}_j concurrently append values v and v' , respectively, to a label ℓ ’s tuple (v_1, \dots, v_n) . In this case, both \mathbf{C}_i and \mathbf{C}_j will use their local state to send $(F(K_{\ell,1}, n+1), v)$ and $(F(K_{\ell,1}, n+1), v')$ to the server which will result in one pair overwriting the other. One approach to address this could be to encrypt and outsource the state to the server and synchronize using locks, which is a primitive that ensures exclusive access to shared resources. To append values, clients would then need to acquire the lock, retrieve and decrypt the state, send the new pair together with an updated and encrypted state and release the lock.

This naive form of server-side synchronization has both consistency and security issues. Suppose we have two clients \mathbf{C}_i and \mathbf{C}_j and that when \mathbf{C}_i acquires the lock the counter is at 5 and when \mathbf{C}_j acquires the lock the counter is at 6. In this case, \mathbf{C}_i sends $(F(K_{\ell,1}, 6), \text{Enc}(K_{\ell,2}, v_i))$ to the server whereas \mathbf{C}_j

sends $(F(K_{\ell,1}, 7), \text{Enc}(K_{\ell,2}, v_j))$. Now, we will describe an execution schedule in which \mathbf{C}_j 's append will not be output by its own get which proves that the construction is not linearizable. This happens if \mathbf{C}_j 's append is scheduled but \mathbf{C}_i 's append is not. In this case, $(F(K_{\ell,1}, 7), \text{Enc}(K_{\ell,2}, v_j))$ is inserted into the dictionary whereas $(F(K_{\ell,1}, 6), \text{Enc}(K_{\ell,2}, v_i))$ is not. Notice that if \mathbf{C}_j executes a get on ℓ after its append is completed, it will not receive v_j because when the server queries the dictionary on the tag $F(K_{\ell,1}, 6)$, it will not find it and, therefore, stop and return the values associated with counters 1 through 5. Linearizability requires gets to return all the values of appends that finished before the get started, so this construction is not linearizable. In fact, this construction does not even achieve the weaker notion of sequentially consistency and only satisfies the weakest form of consistency known as *eventual consistency*.

A General Approach for Linearizability. The problem with the previous approach is that only locking the state is not enough to synchronize append and get operations. A general-purpose way to fix this is to put both the state and the EMM—the dictionary in the case of π_{bas} —together under one lock. In this case, only one operation can change or read the structure/dictionary at a time. This solution, however, severely limits scalability and throughput and essentially operates like a sequential implementation.

Naive State Access. Another issue with naive server-side synchronization is that naively accessing the state could leak additional information. Specifically, if the clients only retrieves the relevant (encrypted) counter during an append, the server would learn append-to-append correlations (i.e., whether two appends are for the same label or not). This can be avoided if the clients retrieve the entire state each time or store and query it using an oblivious RAM (ORAM) but both approaches are costly.

1.1 Our Contributions

In this work, we initiate the study of concurrent encrypted data structures. Specifically, we formalize, define and construct a multi-map encryption scheme that encrypts multi-maps in such a way that they can be accessed concurrently with high throughput and that satisfies linearizability. Though our construction is one of our main contributions, we also identify a variety of interesting definitional and modeling issues that need to be addressed to even formalize the security of concurrent encrypted structures.

Instruction-Level Leakage. One of our core observations is that leakage in the concurrent setting is quite different than leakage in the sequential setting and, therefore, needs to be modeled differently. The observation stems from the fact that, as discussed above, operations are really sequences of instructions and that, in reality, leakage occurs at the instruction level and is produced little by little with every instruction that is executed. To see this, consider the simplified version of π_{bas} described above. Notice that during a get, the server learns

information piece-by-piece as it queries the dictionary. For example, if the dictionary query for $F(K_{\ell,1}, i)$ is successful the server learns that the length of the tuple is *at least* i . And when the dictionary query fails, it finally learns the exact length of the tuple. The fact that leakage really occurs at the instruction level does not necessarily contradict the standard operation-level leakage model [20, 23] which is (implicitly) used in the sequential setting. This is because, in the sequential setting, all the instructions of an operation are executed together before the instructions of the next operation are started. The consequence is that the operation-level leakage is the union of the instruction-level leakage. For example, in π_{bas} the operation-level get leakage is the length of the tuple which is also the instruction-level get leakage in the sequential setting.

Schedule-Dependent Leakage. Another important observation is that instruction-level leakage depends on how it is scheduled. For example, consider a π_{bas} EMM that stores a label pair (ℓ, \mathbf{v}) , where $\mathbf{v} = (v_1, \dots, v_{10})$, and the following two concurrent operations: an append to ℓ 's tuple and a get for ℓ . Now, consider one schedule where the append adds the pair $(F(K_{\ell,1}, 11), \text{Enc}(K_{\ell,2}, v_{11}))$ to the underlying dictionary before the get operation queries the dictionary for $F(K_{\ell,1}, 11)$ and another schedule where the order is reversed. In the first schedule, the server learns that \mathbf{v} has size at least 11 from the dictionary query for $F(K_{\ell,1}, 11)$, whereas in the second schedule it learns that the tuple length is exactly 11¹.

Adversarial Schedulers. As discussed above, real-world database servers are multi-threaded and instructions are ordered and scheduled for execution by the OS scheduler. If the server is corrupted—which is the standard adversarial model considered in encrypted search—then the scheduler is also corrupted and the execution schedule of a sequence of operations will be adversarially-chosen and because of this the observations above have important implications on the security of encrypted structures. In particular, recall that it is standard for dynamic encrypted structures to achieve forward privacy. Roughly speaking, forward privacy guarantees that updates to the structure cannot be correlated with previous queries but can reveal correlations between queries and past updates. The observation that instruction-level leakage is schedule-dependent implies that, in the presence of an adversarial scheduler, the notions of forward and backward privacy are not meaningful. This is simply because an adversarial scheduler can render the guarantees of forward privacy useless by controlling the schedule. Specifically, given a sequence of queries followed by updates, forward privacy guarantees that the updates cannot be correlated to any of the queries. But by scheduling all the updates first followed by the queries these correlations can be revealed. A similar issue occurs with backward privacy which, roughly speaking, guarantees that queries do not reveal information about items that have been previously inserted and then deleted. As in the case of forward privacy, if an

¹ Note that, in this example, the ordering of the operations also impacts the leakage at the operation level.

adversarial scheduler re-orders the operations such that an item is inserted, then queried and then deleted, then it can learn whether the item was inserted.

Note that, so far, we only discussed adversarial schedulers that choose fixed schedules, i.e., in a non-adaptive manner. But a scheduler could determine a schedule *adaptively*, as a function of previous instructions, their outputs and their leakage.

Modeling Instruction-Level Leakage. To capture instruction-level leakage and to properly capture the interaction between adaptive schedulers and leakage, we formalize leakage profiles in a more fine-grained manner. Specifically, we define leakage functions as stateful functions that take as input a sequence of *instructions* (as opposed to a sequence of operations), a schedule for the instructions executed so far and the next instruction to be executed. Based on these inputs, the leakage function determines the leakage produced by the next instruction which, in our security definition, is provided to the adaptive scheduler.

Formalizing Security. We formalize the security of a concurrent multi-map encryption scheme in the ideal/real-world paradigm. At a high level, in the real world the clients and server execute the real multi-map encryption scheme operations in the presence of a semi-honest adversary that corrupts the server. In the ideal world, the clients and server interact with an ideal concurrent multi-map functionality. Defining this ideal functionality is challenging for the following reasons. The ideal functionality should produce a sequence of outputs that is consistent. There are, however, many possible consistent output sequences. We could make the functionality produce a specific one of them but this would be too strong and not achievable since, in the real world, the adversary controls the scheduler and can, therefore, influence the outputs of the encrypted multi-map. To capture this, we need to relax the functionality and allow the simulator to provide it with information that it can use to generate an output sequence. But, crucially, we require the functionality to abort if the simulator leads it to output a sequence that violates consistency. This guarantees that the functionality and the scheme always produce consistent output sequences but that the adversary can influence which consistent output sequence it produces. As with traditional (i.e., sequential) STE security definitions, we explicitly model leakage in our definitions.

Another interesting feature of our definition is how adaptive adversarial schedulers are handled. Recall that adaptive schedulers choose the next atomic instruction to execute based on previous instructions, their results and possibly their leakage. During simulation, this is handled by the simulator forwarding the scheduler's next instruction to the functionality which returns the instruction-level leakage so that the simulation can proceed.

A New Concurrent Multi-map Encryption Scheme. We describe a linearizable multi-map encryption scheme called TST which, as far as we know, is the first such construction. In addition, TST achieves lock-free append operations which, roughly speaking, means that if an append gets scheduled it will never have to

wait on another operation. At a very high level, the scheme works as follows. Suppose we have n clients C_1, \dots, C_n . A label ℓ 's tuple \mathbf{v} can be split into n sub-tuples $(\mathbf{v}_{1,\ell}, \dots, \mathbf{v}_{n,\ell})$, where $\mathbf{v}_{i,\ell}$ holds the values appended by C_i . For all clients C_i and labels ℓ , the encrypted multi-map will store an encrypted form of a reverse linked list $\text{list}_{i,\ell}$, where each node stores a value of $\mathbf{v}_{i,\ell}$ and a pointer to the previous node in the list. It also stores an encrypted structure that stores pointers to the heads of the list. To append a value v_{m+1} to ℓ 's tuple, C_i sends a new node that stores v_{m+1} and points to the head of $\text{list}_{i,\ell}$. To retrieve ℓ 's tuple, the server walks each $\text{list}_{i,\ell}$, for $i \in [n]$, starting from their heads and returns the nodes in $\cup_{i \in [n]} \text{list}_{i,\ell}$. Note that, for security reasons, append operations do not update the structure that stores the heads of the list so that structure can store old/stale head pointers which could result in the get only returning a subset of ℓ 's tuple. To address this, we augment the construction with additional encrypted data structures that the server can use at get time to find the unreachable nodes (i.e., the nodes that cannot be reached from the old head). This adds false positives to the server's response, but the client can locally filter them out. Additionally, the client can use the results to update the structures at the server so that future gets have less false positives (and depending on the distribution of operations possibly none). Like many dynamic EMM constructions [8, 9, 15, 28, 59], TST also uses *lazy deletion* to handle deletes, where deletions are treated as additions with special delete markers that the client can locally use to filter out the deleted values. The description provided here is very high level and ignores many subtleties and technical challenges which we discuss in detail in Sects. 5 and 6.

One of the main challenges in designing TST was to find a way to achieve linearizability, security and efficiency. Traditional techniques from concurrent data structures are designed to provide consistency and efficiency whereas traditional techniques from structured encryption are designed to achieve security and efficiency. In our setting, we need to find ways of using, adapting and creating new techniques so that we achieve all three. Interestingly, while TST is very different than all previous EMM constructions, it does make use of and combine ideas from previously-known influential constructions. Specifically, it is both a list-based scheme like the SSE-1 construction of [23] and a dictionary- and counter-based scheme like the π_{bas} construction of [15].

A New Linearizable Range Dictionary. Our TST construction makes use of a plaintext range dictionary which stores label/value pairs where the labels are integers. In addition to get and put operations, the range dictionary also needs to support a greater-than ℓ operation that returns the set of values associated with labels greater than ℓ . We construct such an efficient and linearizable range dictionary which may be of independent interest.

2 Related Work

Structured Encryption. Structured encryption was introduced by Chase and Kamara [20] as a generalization of index-based searchable symmetric encryption (SSE) [23, 58]. The most common and important type of STE schemes are

multi-map encryption schemes which are a basic building block in the design of sub-linear SSE schemes [15, 23, 41], expressive SSE schemes [13, 29, 38, 39, 55] and encrypted databases [16, 39]. STE and encrypted multi-maps have been studied along several dimensions including dynamism [15, 33, 40, 41, 55] and I/O efficiency [5, 7, 14, 15, 24, 25, 52]. The notion of forward privacy was introduced by Stefanov, Papamanthou and Shi [60] and formally defined by Bost [8], who also proposed the first forward-private encrypted multi-map construction. Kamara and Moataz pointed out in [38] that the definition of [8] does not necessarily capture the intuitive security guarantee of forward-privacy and suggested that it be formalized as requiring that updates be leakage-free. Backward privacy was introduced by Bost, Minaud and Ohrimenko [9]. Several follow up works showed how to improve on the constructions of [9], sometimes achieving both forward and backward privacy [3, 9, 28, 30, 45, 59]. All these works focus on designing SSE/STE constructions for non-concurrent settings. While these non-concurrent constructions have several applications, concurrent constructions that allow multiple operations to operate on the encrypted data structure simultaneously are more practical and useful.

Multi-user Schemes. Multi-user STE/SSE refers to schemes that allow multiple clients to operate on the encrypted structure/collection. Multi-user schemes can be single-writer multi-reader as proposed in [23], multi-writer single-reader, or multi-writer multi-reader. As far as we know all the multi-user constructions proposed—except for [42]—assume the server is sequential and do not support concurrent operations.

Oblivious Parallel RAMs. Oblivious parallel RAM (OPRAM) was introduced by Boyle, Chung, and Pass [10] as a generalization of ORAM that compiles an m -CPU PRAM program into an oblivious m -CPU PRAM. Numerous subsequent work improved OPRAM overhead [6, 17–19, 22, 37, 53]. Although the notions of parallel and concurrent computation are related, they are not the same. Parallel computation involves executing multiple operations at the same time in order to accelerate computationally-intensive tasks by using multiple processing units. In contrast, concurrent computation involves executing multiple operations that can be interleaved with one of another. This means that operations can begin, run, and complete in any sequence and they can share resources such as memory and processors. Our work is focused on concurrency not parallelism.

Concurrent Dictionaries. Our construction makes use of linearizable dictionaries which can be instantiated with hash tables based on closed or open addressing. Existing solutions can be lock-based [27, 47, 49], partially lock-free [34, 36], lock-free [32, 50, 57], or wait-free [61]. Search trees can also be used to instantiate concurrent dictionaries and there are various lock-based designs [11, 46] and lock-free linearizable implementations [26, 62].

3 Preliminaries

Notation. The set of all binary strings of length n is denoted as $\{0, 1\}^n$, and the set of all finite binary strings as $\{0, 1\}^*$. $[n]$ is the set of integers $\{1, \dots, n\}$. $a := b$ means that a is set to b . The output y of a deterministic algorithm \mathcal{A} on input x is denoted by $y := \mathcal{A}(x)$. If S is a set then $x \stackrel{\$}{\leftarrow} S$ denotes sampling from S uniformly at random. If S is a set then $\#S$ refers to its cardinality. Throughout, k will denote the security parameter.

Cryptographic Primitives. We make use of CPA-secure symmetric encryption schemes $\text{SKE} = (\text{Gen}, \text{Enc}, \text{Dec})$ and pseudo-random functions (PRF) in our construction. We denote the evaluation of a pseudo-random function F with a key K on an input x as $F(K, x)$. Sometimes for visual clarity, we denote $F(F(K, x_1), x_2), \dots, x_n$ as $F[K, x_1, \dots, x_n]$. We refer the reader to [44] for standard notions of security for PRFs and symmetric encryption schemes.

Plaintext Data Structure Schemes. A dictionary scheme $\Delta_{\text{DX}} = (\text{Init}, \text{Get}, \text{Put})$ supports three algorithms where $\text{Init}(k_1, k_2)$ initializes an empty dictionary DX that maps labels of length k_1 to values of length k_2 , $\text{Put}(\text{DX}, \ell, v)$ adds a label/value pair (ℓ, v) to DX, and $\text{Get}(\text{DX}, \ell)$ returns the value v associated with label ℓ in DX. A cas-dictionary scheme $\Delta_{\overline{\text{DX}}} = (\text{Init}, \text{Get}, \text{Put}, \text{CompareAndSwap})$ is a dictionary scheme that supports, in addition to Get and Put, a CompareAndSwap algorithm. $\text{CompareAndSwap}(\overline{\text{DX}}, \ell, v_{\text{old}}, v_{\text{new}})$ compares the value of label ℓ with v_{old} and updates it with v_{new} if they are equal [51, 54, 56]. A range dictionary scheme $\Delta_{\text{RDX}} = (\text{Init}, \text{Put}, \text{GetGreater})$ is a dictionary scheme that supports GetGreater instead of Get. In particular, the $\text{GetGreater}(\text{RDX}, \ell)$ algorithm returns all the values in the dictionary that have labels ℓ' greater than ℓ . Finally, $\Delta_{\text{CTR}} = (\text{Init}, \text{FetchAndInc})$ is a counter scheme where $\text{Init}(v)$ initializes a counter count_g with value v , and $\text{FetchAndInc}(\text{count}_g)$ returns the current value of count_g and increments the counter by 1.

Operations. We define an operation op as a tuple $(\text{opid}, \text{name}, \text{inp}, \text{cid})$ which includes a unique operation id, the operation's name, its input, and the id of the client who issued the operation. For example, $\text{op} = (\text{opid}, \text{Get}, \ell, i)$ is a get operation that takes a label ℓ as its input and is issued by client \mathbf{C}_i .

Atomic Instructions. We assume that each operation op consists of atomic instructions $(\text{ins}_1, \dots, \text{ins}_\lambda)$, which are instructions that guarantee uninterrupted access and updates of shared single-word variables. We use $\text{op}.\text{[First]}$ and $\text{op}.\text{[Last]}$ to indicate the first and last atomic instruction in a sequence of atomic operations that make up an operation. We also use $\text{op}.\text{[ins}_i\text{]}$ or $\text{opid}.\text{[ins}_i\text{]}$ to refer to the i th instruction of operation op .

Execution Schedules. We assume the server has a scheduler that is responsible for scheduling operations on the CPU. Given a set of operations $\Omega = \{\text{op}_1, \dots, \text{op}_\lambda\}$, the execution process is as follows: (1) the scheduler schedules an operation; (2)

the CPU executes the *next* atomic instruction of the scheduled operation; and (3) then the scheduler de-schedules the operation. After that, the scheduler schedules another operation, the CPU which carries out the next atomic instruction of that operation, and so on. This process defines an *execution schedule*, sched_Ω , for a set of operations Ω .

It is important to note that the *next* atomic instruction of the scheduled operation is not necessarily next atomic instruction in the literal code of the operation. Instead, it is determined by accounting for branches and other control flow constructs. Consequently, two get operations with identical code but different inputs can follow entirely different execution paths due to different inputs and can, therefore, have different execution schedules.

We define a schedule sched_Ω for a set of operations Ω as a prefix of the following sequence: $(\text{opid}_1.\text{[First]}, \dots, \text{opid}_1.\text{[Last]}) \times \dots \times (\text{opid}_\lambda.\text{[First]}, \dots, \text{opid}_\lambda.\text{[Last]})$, where the \times operator denotes the interleaving of executions. We stress that execution schedules need not be complete in the sense that they do not have to contain all the atomic instructions of an operation. Intuitively, they represent the execution that has happened *so far* on the machine. Given an operation set Ω , and a schedule sched_Ω , we partition Ω into two disjoint sets $\Omega_f \cup \Omega_p$, where Ω_f is the set of operations that are completed, and Ω_p is the set of operations that are partially completed. Formally an operation $(\text{opid}, \star, \star, \star) \in \Omega$ is in Ω_f if $(\text{opid}.\text{[First]}, \dots, \text{opid}.\text{[Last]}) \in \text{sched}_\Omega$, otherwise it is in Ω_p .

Concurrent Schedules. Given a schedule sched_Ω , we write $\text{time}_{\text{sched}}(\text{op}.\text{[ins]})$ to refer to the logical time the instruction $\text{op}.\text{[ins]}$ is executed. For instance, for an operation op , $\text{time}_{\text{sched}}(\text{op}.\text{[First]})$ and $\text{time}_{\text{sched}}(\text{op}.\text{[Last]})$ refer to the start and end times of the operation. If $\text{op}.\text{[Last]} \notin \text{sched}_\Omega$, then $\text{time}_{\text{sched}}(\text{op}.\text{[Last]}) = \infty$. A schedule sched_Ω is called *concurrent* if there exists a time when two operations are “active”. Formally, there exist a time t , such that $\text{time}_{\text{sched}}(\text{op}.\text{[First]}) \leq t \leq \text{time}_{\text{sched}}(\text{op}.\text{[Last]})$ and that $\text{time}_{\text{sched}}(\text{op}'.\text{[First]}) \leq t \leq \text{time}_{\text{sched}}(\text{op}'.\text{[Last]})$. We say that op and op' are concurrent if this condition is true.

Execution Histories. An *execution history* is a record of the operations executed on a data structure, when they were executed, and their outputs were. We model it as a triple $H = (\Omega, \text{sched}_\Omega, \text{out}_{\Omega_f})$, where sched_Ω is the execution schedule of the operations in Ω and out_{Ω_f} is an output function that assigns an output to finished operations $\Omega_f \subseteq \Omega$.

Correctness of Concurrent Data Structures. A concurrent data structure is a data structure that allows multiple operations to be executed simultaneously without violating its correctness. The correctness of a concurrent data structure is formalized by a notion of consistency which, intuitively, guarantees that the operations executed on a data structure should always appear to be executed one after the other, even if their executions were interleaved. For instance, a get operation on a multi-map should output all the values that appear to have been added by “previous” appends, without including any of the values that appear

to have been added by “later” appends. We capture the notion of *sequential correctness* with a function called `exec`. It takes as input a total order seq_Ω of operations Ω , an operation $\text{op} \in \Omega$, and returns the output of op as if the operations were executed in the order of seq_Ω . In the particular case of a multi-map, `exec` is defined as follows. If $\text{op} = (\text{opid}, \text{Get}, \ell, \star)$, $\text{exec}(\text{seq}_\Omega, \text{op}) = \{v : (\text{opid}', \text{Append}, (\ell, v), \star) \in \Omega \text{ and } \text{seq}_\Omega \text{ orders } \text{opid}' \text{ before } \text{opid}\}$. Various consistency notions define how operations in Ω can be ordered in relation to their original ordering in sched_Ω , determining what is considered “previous” or “later”. Stricter consistency results in fewer possible orderings, while weaker consistency allows for more. We formalize this intuition in the definition in the definition below where we use a predicate χ to capture constraints on how operations can be ordered. Given a schedule sched_Ω and a sequential order seq_Ω , it outputs 1 if seq_Ω orders certain operation pairs in the same way as sched_Ω , and 0 otherwise.

Definition 1 (χ -consistent histories). *A history $H = (\Omega, \text{sched}_\Omega, \text{out}_{\Omega_f})$ is χ -consistent if there exists a sequence seq_Ω such that:*

$$\chi(\text{sched}_\Omega, \text{seq}_\Omega) = 1 \quad \text{and} \quad \forall \text{op} \in \Omega_f, \text{exec}(\text{seq}_\Omega, \text{op}) = \text{out}_{\Omega_f}(\text{op}),$$

where $\Omega_f \subseteq \Omega$ is the set of completed operations, and `exec` is a function that assigns op an output that conforms to the sequential correctness of the data structure when executing operations in Ω in the sequential order seq_Ω .

We say that a concurrent data structure is called χ -consistent, if all its execution histories are χ -consistent.

Linearizability. Linearizability [35] is a popular and strong consistency notion that requires operations to preserve their real-time ordering, i.e., if op completes before operation op' begins, then op should take effect before op' . Said differently, it implies that operations should appear to be interleaved at the granularity of complete operations, and the order of non-overlapping operations is preserved. We formally define this notion below.

Definition 2 (Linearizability). *An execution history $H = (\Omega, \text{sched}_\Omega, \text{out}_{\Omega_f})$ is linearizable if the two conditions below are verified:*

1. (span membership): *for each operation $\text{op} \in \Omega$, there exists a point in time $\text{linp}(\text{op}) \in \mathbb{R}$, called its linearization point, such that*

$$\text{time}_{\text{sched}}(\text{op}.\text{First}) \leq \text{linp}(\text{op}) \leq \text{time}_{\text{sched}}(\text{op}.\text{Last})$$

2. (correctness): *for all $\text{op} \in \Omega_f$, $\text{exec}(\text{seq}_\Omega^{\text{linp}}, \text{op}) = \text{out}_\Omega(\text{op})$, where $\text{seq}_\Omega^{\text{linp}}$ is created from the linearization points as follows. Let op and op' be two operations in Ω and let opid and opid' be their operation ids. If $\text{linp}(\text{op}) < \text{linp}(\text{op}')$, then order opid before opid' in $\text{seq}_\Omega^{\text{linp}}$.*

Intuitively, the linearization point of an operation captures the instant when the operation appears to have taken effect. For example, the linearization point of an append operation is the point in its execution before which its value was not in the multi-map but after which it definitely is.

Progress Guarantees. The concept of termination is more complicated in concurrent execution than in sequential execution because an operation's completion depends not only on its own execution but also on the execution of other operations. For instance, an operation that is waiting on a lock cannot proceed until the operation that holds the lock releases it. Progress guarantees define conditions under which an operation is ensured to complete. These guarantees are broadly classified as non-blocking and blocking, where the former allows other operations to proceed even if one operation is delayed, while the latter does not. Non-blocking guarantees are further classified into wait-freedom and lock-freedom. An operation is considered wait-free if its executions complete in a finite number of scheduler steps. In contrast, an operation is lock-free if it guarantees that some operation call will finish in a finite number of scheduler steps, even if not all calls will. Similarly, blocking guarantees are further classified into starvation-freedom and deadlock-freedom. An operation is starvation-free if its executions can make progress provided that the locks are not held infinitely by the other executions. On the other hand, an operation is deadlock-free if some call will make progress.

4 Definitions

Structured encryption (STE) was introduced in [20] as a generalization of index-based² SSE schemes [23]. The notion of SSE was introduced in [58] and formalized in [23]. There are several forms of structured encryption. The original definition of [20] considered schemes that encrypt both a structure and a set of associated data items (e.g., documents, emails, user profiles etc.). In [21], the authors also describe *structure-only* schemes which only encrypt structures. One can also distinguish between *response-hiding* and *response-revealing* schemes: the former reveal the response to queries whereas the latter do not.

Definition 3 (Append-only multi-map encryption scheme). *A response-hiding multi-client append-only multi-map encryption scheme $\Sigma_{\text{MM}} = (\text{Init}, \text{Append}, \text{Get})$ consists of three two-party protocols that are executed by n clients C_1, \dots, C_n and a server S and work as follows:*

- $(K_1, \text{st}_1; \dots; K_n, \text{st}_n; \text{EMM}) \leftarrow \text{Init}_{C_1, \dots, C_n, S}(1^k; \dots; 1^k; 1^k)$: is a probabilistic algorithm that takes as input from the clients and server a security parameter 1^k . It outputs to a client C_i a key K_i and state st_i and to the server an encrypted multi-map EMM ;
- $(\text{st}'_i; \text{EMM}') \leftarrow \text{Append}_{C_i, S}(K_i, \text{st}_i, (\ell, v); \text{EMM})$: takes as input from the client its key K_i and state st_i and a label/value pair (ℓ, v) ; and from the server an encrypted multi-map EMM . It outputs to the client an updated state st'_i and to the server an updated encrypted multi-map EMM' ;
- $(\text{st}'_i, \mathbf{v}; \perp) \leftarrow \text{Get}_{C_i, S}(K_i, \text{st}_i, \ell; \text{EMM})$: takes as input from the client its key K_i and state st_i and a label ℓ ; and from the server an encrypted multi-map

² In the literature structure-based schemes are also called index-based schemes.

EMM. It outputs to the client a (possibly) updated state st'_i and a tuple \mathbf{v} and to the server \perp ;

We stress that all the protocols (except the `Init`) can be executed by many clients concurrently. For parameters $\theta, \lambda \in \mathbb{N}_{\geq 1}$, we use $\mathbb{L}_{\text{MM}} = \{0, 1\}^\theta$ to denote the label space and $\mathbb{V}_{\text{MM}} = \{0, 1\}^\lambda$ to denote the value space of the multi-map.

4.1 Security Definition

We now turn to formalizing the security of a *concurrent* multi-map encryption scheme. We do this by combining the definitional approaches used in secure multi-party computation [12] and in structured encryption [20, 23]. The security of multi-party protocols is generally formalized using the ideal/real-world paradigm. To capture the fact that a protocol could leak information to the adversary, we parameterize the definition with a leakage profile that consists of a leakage function \mathcal{L} that captures the information leaked by the execution of the operations.

Adversarial Model. In this work, we consider semi-honest adversaries that corrupt the server and, therefore, see all its stored data, randomness, client operations, and shared memory instructions. Furthermore, we assume that the adversary has control over the scheduler and can determine which atomic instruction is executed at any given time. This implies that the adversary selects a schedule sched_Ω for a given operation set Ω .

The Real-World Execution. The real-world experiment is executed between a set of n clients $\mathbf{C}_1, \dots, \mathbf{C}_n$, a server \mathbf{S} , an environment \mathcal{Z} and an adversary \mathcal{A} . Given $z \in \{0, 1\}^*$, the environment \mathcal{Z} sends a message to the adversary \mathcal{A} to corrupt the server \mathbf{S} . The clients and the server then execute $\Sigma_{\text{CMM}}.\text{Init}(1^k)$. \mathcal{Z} then adaptively chooses a polynomial number of operations $(\text{op}_1, \dots, \text{op}_q)$, where op_j is either a (Get, ℓ, i) tuple or a $(\text{Append}, (\ell, v), i)$ tuple. For all $j \in [q]$, \mathcal{Z} sends op_j to client \mathbf{C}_i . If op_j is a get operation \mathbf{C}_i executes $\Sigma_{\text{CMM}}.\text{Get}$ with the server but if it is an append operation, \mathbf{C}_i executes $\Sigma_{\text{CMM}}.\text{Append}$. The adversary also communicates with the environment throughout the run of the experiment. Since the adversary controls the scheduler and also communicates with the environment, \mathcal{A} and \mathcal{Z} decide how to schedule operations. When an operation op_j finishes, the server returns the response to the right client \mathbf{C}_i which, in turn, sends it to the environment \mathcal{Z} . After all the operations are executed, the adversary \mathcal{A} sends a message m to \mathcal{Z} who returns a bit that is output by the experiment. We let $\text{Real}_{\mathcal{A}, \mathcal{Z}}(k)$ be the random variable denoting \mathcal{Z} 's output bit.

The Ideal-World Experiment. The experiment is executed between a set of n dummy clients $\mathbf{C}_1, \dots, \mathbf{C}_n$, an environment \mathcal{Z} and a simulator Sim , where the environment and the simulator can communicate at any point in the experiment. Each party also has access to the ideal functionality $\mathcal{F}_{\text{CMM}}^{\lambda, \mathcal{L}}$. Given $z \in \{0, 1\}^*$, \mathcal{Z} sends a message to the simulator Sim to corrupt the simulated server \mathbf{S} . \mathcal{Z}

then adaptively chooses a polynomial number of operations $(\text{op}_1, \dots, \text{op}_q)$, where op_j is either a (Get, ℓ, i) tuple or an $(\text{Append}, (\ell, v), i)$ tuple. For all $j \in [q]$, \mathcal{Z} sends op_j to a dummy client \mathbf{C}_i which forwards it to the functionality $\mathcal{F}_{\text{CMM}}^{\chi, \mathcal{L}}$. Upon receiving a message the functionality executes its prescribed procedure from Fig. 1 with simulator Sim . When a dummy client receives an output from the functionality, it forwards it to \mathcal{Z} . In the end, Sim computes a message m from its view and sends it to \mathcal{Z} . Finally, \mathcal{Z} returns a bit that is output by the experiment. We let $\mathbf{Ideal}_{\text{Sim}, \mathcal{Z}}(k)$ be the random variable denoting \mathcal{Z} 's output bit.

Definition 4 ((χ, \mathcal{L}) -security). *We say that a concurrent encrypted multi-map scheme $\Sigma_{\text{MM}} = (\text{Init}, \text{Get}, \text{Append})$ is (χ, \mathcal{L}) -secure, if for all PPT adversaries \mathcal{A} , and all PPT environments \mathcal{Z} , there exists a PPT simulator Sim such that for all $z \in \{0, 1\}^*$, $|\Pr[\mathbf{Real}_{\mathcal{A}, \mathcal{Z}}(k) = 1] - \Pr[\mathbf{Ideal}_{\text{Sim}, \mathcal{Z}}(k) = 1]| \leq \text{negl}(k)$.*

Note that in both experiments, the environment can send an operation to any client at any time so there can be multiple operations executed concurrently at the server.

4.2 An Ideal Concurrent Multi-map Functionality

Our ideal functionality captures all the properties of a secure concurrent multi-map, in particular, its consistency and security guarantees.

Capturing Consistency in the Ideal Functionality is Challenging. We begin by discussing two challenges that arise in capturing the consistency guarantees of a concurrent multi-map in the ideal/real-world paradigm. The first challenge is that operations take time to execute and do not finish instantaneously in the real world. To address this, we need to create an ideal functionality that can account for this behavior. One option is to allow the functionality to choose an arbitrary time to return output. However, this approach is problematic because in the real world, the adversary controls when an operation ends, and there is no way for the functionality to know this. The second challenge is determining the outputs that the functionality should produce for operations. We want to achieve χ -consistency, but there are multiple possible output sequences that can satisfy this. Creating a functionality that chooses a specific sequence is also not achievable because the scheduler can influence the outputs by forcing a specific interleaving of atomic instructions. To address these challenges, we relax the functionality and allow the simulator (i.e., the ideal adversary) to influence the functionality's output. When an operation ends, the simulator gives the functionality a sequential order of operations that it uses to compute the operation's output.

The Functionality Overview. We formally describe the ideal concurrent multi-map functionality $\mathcal{F}_{\text{CMM}}^{\chi, \mathcal{L}}$ in Fig. 1. The functionality stores two operation sets Ω and Ω_f , where Ω stores all the client operations, and $\Omega_f \subseteq \Omega$ stores all the

completed operations. Both the sets start out as empty sets. The functionality also stores a schedule `sched` of operations, and an output function `out`, both of which it builds over time. When the functionality receives an operation `op` from a client C_i , the functionality assigns the operation a unique `opid`, adds the operation to Ω , and sends to the simulator the operation id, its name, and the client id. Note that our functionality implicitly leaks the type of the operation, and the client making that operation. As previously discussed, in concurrent settings, the scheduler selects the next instruction to be executed. This means that the simulator must receive information on the leakages at the instruction level. In order to obtain the leakages of an operation `op`'s instruction `ins`, the simulator sends a message `(opid, ins)` to the functionality to obtain its leakage. The functionality first checks its local schedule `sched` to confirm if `ins` is the next instruction that needs to be executed for `op`. If yes, it sends $\mathcal{L}(\Omega, \text{sched}, \text{opid.ins})$ to the simulator and updates the schedule `sched` with `opid.ins`. Otherwise, it aborts.

Moreover, when `opid.ins` is the last instruction for operation `op`, the simulator additionally sends the functionality a sequential order `seq` of operations. Recall that we want to capture the notion of χ -consistency in the functionality. Therefore, the functionality makes the checks required by the definition of χ consistency. In particular, it checks if `seq` is sequential, if $\chi(\text{sched}, \text{seq}) = 1$, and if for all the operations `op'` $\in \Omega_f$ completed so far, if $\text{exec}(\text{seq}, \text{op}') = \text{out}(\text{op}')$. If `seq` passes all the checks, the functionality accepts `seq`, else it aborts. It finally computes the output $r = \text{exec}(\text{seq}, \text{op})$ of the operation just completed and returns it to C_i . Finally, it updates its set of completed operations Ω_f and adds `op` to it.

5 A (Plaintext) Linearizable Multi-map

In this section, we describe a linearizable multi-client plaintext multi-map. This structure underlies our main multi-map encryption scheme TST and will make its description in Sect. 6 easier to understand. We start with a straw-man construction and gradually build towards a final construction while highlighting various security, efficiency and concurrency considerations.

5.1 An Initial Design

We construct a multi-client plaintext multi-map data structure $\text{MM} = (\text{dDX}, \text{cDX}_1, \dots, \text{cDX}_n)$ that is composed of a *data dictionary* `dDX` and n *checkpoint dictionaries* `cDXi`, where n is the number of clients. For simplicity, we assume the structure is accessed by a fixed number of clients, C_1, \dots, C_n , but note that the structure can handle a variable number of clients. The data dictionary `dDX` will store labels and tuple values and the checkpoint dictionaries will store meta-data necessary for correctness and fast get operations.

Intuitively, the structure stores a reverse linked list `listi,ℓ` for each client C_i that inserts a label/tuple pair (ℓ, \mathbf{v}_i) . The list `listi,ℓ` = $(\text{node}_{i,\ell,1}, \dots, \text{node}_{i,\ell,m})$

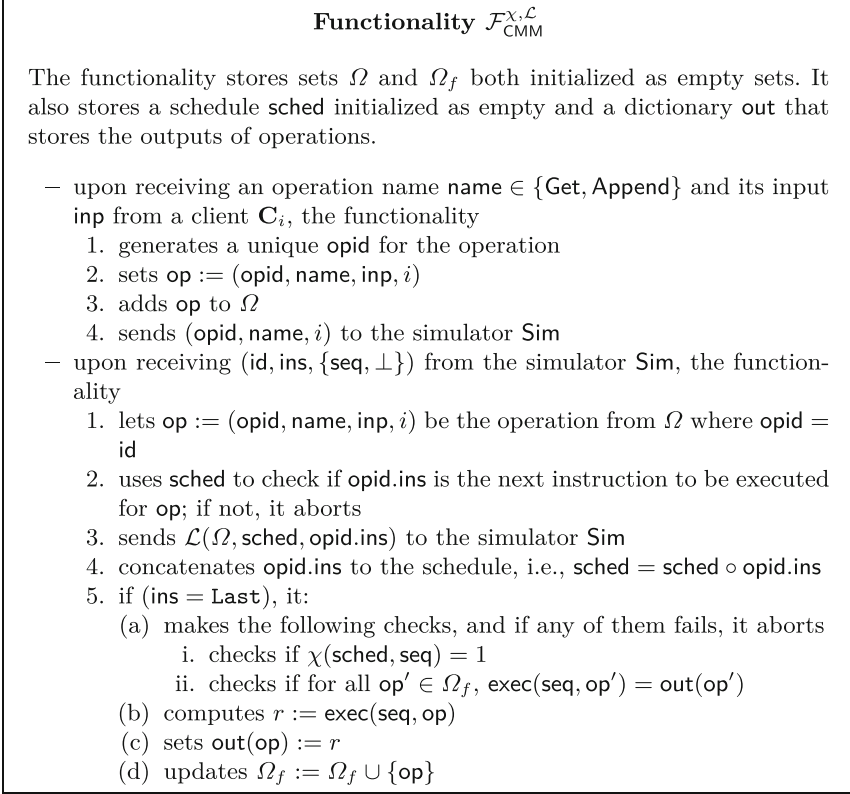


Fig. 1. The concurrent multi-map functionality parameterized with consistency guarantee χ and a leakage function \mathcal{L} .

is composed of $m = \#\mathbf{v}_i$ nodes $\text{node}_{i,\ell,j} = (v_j, \text{addr}_{i,\ell,j-1})$, each of which stores a value $v_j \in \mathbf{v}_i$ and a pointer $\text{addr}_{i,\ell,j-1}$ to the previous node, where $\text{addr}_{i,\ell,0} = \perp$. The address of $\text{list}_{i,\ell}$'s head is then stored in \mathbf{C}_i 's checkpoint dictionary cDX_i . Storing label/tuple pairs using per-client lists has several advantages, one of which is that it enables concurrent appends without needing clients to synchronize on their state. Typically, nodes of the lists are stored in memory and pointers are memory addresses but, in our case, we store them in the data dictionary dDX so addresses and pointers are dDX labels. Specifically, each node $\text{node}_{i,\ell,j} = (v_j, \text{addr}_{i,\ell,j-1})$ is stored in dDX by setting $\text{dDX}[\text{addr}_{i,\ell,j}] := \text{node}_{i,\ell,j}$, where $\text{addr}_{i,\ell,j}$ is a k -bit string chosen uniformly at random. The address $\text{addr}_{i,\ell,m}$ of $\text{list}_{i,\ell}$'s head is then stored in \mathbf{C}_i 's checkpoint dictionary by setting $\text{cDX}_i[\ell] := \text{addr}_{i,\ell,m}$. Throughout, we will sometimes refer to the nodes of $\text{list}_{i,\ell}$ as (i, ℓ) -nodes, to the nodes in dDX inserted by client \mathbf{C}_i as i -nodes and to the nodes in dDX that hold ℓ 's values as ℓ -nodes.

Get. To get the tuple associated with a label ℓ , a client \mathbf{C}_i sends ℓ to the server. The latter then retrieves the head address of every ℓ -list and recovers the values

from those lists. More precisely, for all $i \in [n]$, the server computes $\text{addr}_{i,\ell,m} := \text{cDX}_i[\ell]$ and $\text{node}_{i,\ell,m} := \text{dDX}[\text{addr}_{i,\ell,m}]$, parses $\text{node}_{i,\ell,m}$ as $(v_m, \text{addr}_{i,\ell,m-1})$, adds v_m to the response \mathbf{r} , and then performs the same steps for the node at address $\text{addr}_{i,\ell,m-1}$ until it reaches a node such that $\text{addr}_{i,\ell,j-1} = \perp$.

Append. To append a value v_{m+1} from client \mathbf{C}_i to the tuple of a label ℓ , the client sends (ℓ, v_{m+1}) to the server. The latter computes $\text{addr}_{i,\ell,m} := \text{cDX}_i[\ell]$, creates a new node $\text{node}_{i,\ell,m+1} = (v_{m+1}, \text{addr}_{i,\ell,m})$, samples a new dDX label $\text{addr}_{i,\ell,m+1} \stackrel{\$}{\leftarrow} \{0,1\}^k$ and inserts the node into the data dictionary by setting $\text{dDX}[\text{addr}_{i,\ell,m+1}] := \text{node}_{i,\ell,m+1}$. It then updates \mathbf{C}_i 's checkpoint dictionary by setting $\text{cDX}_i[\ell] := \text{addr}_{i,\ell,m+1}$.

5.2 Towards a Secure Design

The structure described above is efficient and is straightforward to encrypt using any of a variety of practical dictionary encryption schemes. The resulting encrypted structure would also be efficient but it would not achieve the level of security we want. To see why, consider the case where \mathbf{C}_i performs an append on ℓ twice. Even if the multi-map is encrypted, the server would learn that \mathbf{C}_i appended to the same label twice because the two operations cause the server to set $\text{cDX}_i[\ell]$ twice. These append-to-append correlations are problematic because they reveal the length of the tuple already at append time. Another issue is that this approach also reveals get-to-append correlations which can be exploited using adaptive injection attacks [63].

Leakage-Free Appends Through Client State. To address this, we want a solution with no append leakage at all. This could be achieved by storing and accessing the checkpoint dictionaries using black-box ORAM simulation or by using a leakage-free dictionary [31, 43], but this would result in high overhead and multiple rounds of interaction. Instead, we take a different approach and modify the append operation as follows. Specifically, appends will store the new node in the data dictionary dDX but will not update the checkpoint dictionaries. We also require the clients to store local state that maps labels to the head addresses of their own lists (but not of other clients' lists). During an append operation for (ℓ, v_{m+1}) , the client \mathbf{C}_i sends to the server a pair $(\text{addr}_{i,\ell,m+1}, \text{node}_{i,\ell,m+1})$, where $\text{addr}_{i,\ell,m+1} \stackrel{\$}{\leftarrow} \{0,1\}^k$ and $\text{node}_{i,\ell,m+1} := (v_{m+1}, \text{addr}_{i,\ell,m})$ and $\text{addr}_{i,\ell,m}$ is retrieved from its local state. The client then updates its local state to map ℓ to $\text{addr}_{i,\ell,m+1}$ and the server inserts the new node in the data dictionary dDX by setting $\text{dDX}[\text{addr}_{i,\ell,m+1}] := \text{node}_{i,\ell,m+1}$.

This guarantees that updates are leakage-free (modulo the fact that an append occurred) but introduces a correctness issue for get operations. Specifically, the addresses stored in the checkpoint dictionaries can be out of date in the sense that they do not necessarily point to the heads of the lists anymore. This, in turn, means that there are nodes in each list that could be unreachable and not returned by get operations. Throughout, we will refer to the addresses stored in

the checkpoint dictionaries as *checkpoint addresses* and to the nodes pointed to by those addresses as *checkpoint nodes* and recall that, with the current design, checkpoint nodes are not necessarily heads.

Scanning and Filtering. One way to solve the correctness issue above is to store the label ℓ in the nodes so that they now have the form $(v_j, \text{addr}_{i,\ell,j-1}, \ell)$ instead of $(v_j, \text{addr}_{i,\ell,j-1})$ and to change the get operations to work in two phases as follows. The first phase works as before; that is, for all $i \in [n]$, the server retrieves the checkpoint address $\text{addr}_{i,\ell,j} := \text{cDX}_i[\ell]$ and traverses the list to recover the values stored in the nodes. In the second phase, it recovers the unreachable nodes by scanning all the untouched nodes in dDX (i.e., the nodes in dDX it did not access in the first phase), checking if they hold ℓ or not and, if so, returning the value in the node.

5.3 Towards a Secure and Efficient Design

With the changes made so far, we solved the leakage and correctness issues but introduced non-trivial efficiency overhead. While appends remain optimal, gets are now linear in the size of the multi-map due to the scanning step. In this section, we show how solve this issue.

Time. We first provide an intuitive explanation of how we can address this limitation and then show how to instantiate it concretely. Recall that the purpose of the linear scan in the second phase of gets is to find the nodes that are unreachable from the address stored in the checkpoint dictionary. Our solution will be to build a new set of data structures that map labels to the unreachable nodes so that we can replace the linear scan with an efficient data structure query. Building such structures is possible because of the following key observation: the nodes in the lists $\text{list}_{1,\ell}, \dots, \text{list}_{n,\ell}$ that are unreachable are the ones that were inserted into dDX after the checkpoint nodes. Based on this observation we can modify the operations to work as follows. First, we make the server include a timestamp in every node when it inserts them in dDX during an append. The server also maintains n auxiliary range dictionaries $\text{RDX}_1, \dots, \text{RDX}_n$ such that RDX_i maps timestamps to the labels/addresses of \mathbf{C}_i 's nodes in dDX with the associated timestamp. During a get, the server then does the following. For all $i \in [n]$, it will retrieve the i th checkpoint node by computing $\text{addr}_{i,\ell,j} := \text{cDX}_i[\ell]$ and $\text{node}_{i,\ell,j} := \text{dDX}[\text{addr}_{i,\ell,j}]$. Recall that $\text{node}_{i,\ell,j}$ now has form $(v_j, \text{addr}_{i,\ell,j-1}, \ell, \text{time}_j)$. It then retrieves the unreachable nodes by: (1) computing $(\text{addr}_1, \dots, \text{addr}_p) := \text{RDX}_i[\text{time}_j, \infty]$; (2) retrieving nodes $\text{node}_z := \text{dDX}[\text{addr}_z]$, for $z \in [p]$; and (3) filtering out the nodes that hold ℓ . Note that this time-based solution relies on the assumption that timestamps are strictly increasing; that is, even if two append operations on the same label are concurrent, the server will never assign their nodes the same timestamp and will never assign them a timestamp smaller than any previous append. This could possibly be achieved in practice with a clock that has high enough resolution but we provide an instantiation that does not rely on any assumption.

Our approach is to implement the timestamps using a linearizable counter count_g . The linearizability of the counter guarantees that no two nodes get assigned the same counter and that if an append occurs before another, the former's counter value will be strictly smaller than the latter's. So nodes now have the form $\text{node}_{i,\ell,j} = (v_j, \text{addr}_{i,\ell,j-1}, \ell, \text{count})$ and the range dictionaries RDX_i map counter values to the addresses of \mathbf{C}_i 's nodes in dDX that have that counter. We denote the counter of a node $\text{node}_{i,\ell,j}$ as $\text{count}_{i,\ell,j}$.

Updating the Checkpoint Dictionaries. Up to this point, our solution is correct (i.e., the unreachable nodes are now returned), sub-linear (i.e., we do not need to scan anymore) but we can still improve it. Notice that the efficiency of gets now depends on the number of nodes inserted in dDX since the checkpoint node. It follows then that the more up to date the checkpoint dictionaries are the better. To achieve this, we update the checkpoint dictionaries during get operations (as opposed to append operations). More precisely, at the end of a get the server returns the reachable nodes of $\text{list}_{1,\ell}$ through $\text{list}_{n,\ell}$, and all the nodes inserted after the i th checkpoint node together with their addresses. The client then parses this set of nodes and finds, for all $i \in [n]$, the (i, ℓ) -nodes with the highest counter. Note that these nodes are the heads of the lists. The client returns the heads together with their addresses to the server who can now update the checkpoint dictionaries.

5.4 Towards an Optimal Design

The construction so far has optimal append, a single round of interaction, optimal storage overhead, but the efficiency of gets is

$$\#\text{MM}[\ell] + \sum_{i \in [n]} \sum_{\ell' \neq \ell} \#\text{list}_{i,\ell'}[\text{count} \geq c_{i,\ell}] = O\left(\#\text{MM}[\ell] + \sum_{\ell' \neq \ell} \#\text{list}_{\ell'}[\text{count} \geq \min_{i \in [n]} c_{i,\ell}]\right)$$

where $c_{i,\ell} = \text{chkcount}_{i,\ell}$ is the counter of the (i, ℓ) checkpoint node, $\text{list}_{\ell'} = \cup_{i \in [n]} \text{list}_{i,\ell'}$, and $\text{list}[\text{cond}]$ is the set of nodes in the list that satisfy the condition specified by cond . Notice that the second term in the get complexity depends on the number of non- ℓ -nodes inserted, where ℓ is the queried label. For certain workloads this can lead to non-trivial overhead so we show how to avoid it.

Skipping. To solve the issue above, we add n skip dictionaries $\text{skDX}_1, \dots, \text{skDX}_n$ such that skDX_i maps a label ℓ to the time when an i -node was inserted last, i.e., the largest counter in an i -node. We will refer to the counters stored in skDX_i as *skip counters*. Recall that previously, during the second phase of a get the server would query the range dictionaries RDX_i for all i -nodes with counters greater than the i th checkpoint counter. Now, instead, the server queries the range dictionaries for all i -nodes with counters greater than the i th skip counter. Similar to the checkpoint dictionaries, the skip dictionaries are also updated during gets, where the client parses the set of nodes it retrieves from the server and finds for $i \in [n]$, the i -nodes with the highest counter. The client returns the counters of these nodes to the server who then updates the skip dictionaries.

Efficiency. With the changes described, the complexity of appends, the round complexity and the storage complexity remain the same, whereas the complexity of gets is

$$\#MM[\ell] + \sum_{i \in [n]} \sum_{\ell' \neq \ell} \#\text{list}_{i,\ell'}[\text{count} \geq s_{i,\ell}] = O\left(\#MM[\ell] + \sum_{\ell' \neq \ell} \#\text{list}_{\ell'}[\text{count} \geq \min_{i \in [n]} s_{i,\ell}]\right)$$

where $s_{i,\ell} = \text{skcount}_{i,\ell}$ is the (i, ℓ) skip counter. Observe that for all $\ell \in \mathbb{L}_{MM}$,

$$\min_{i \in [n]} \text{skcount}_{i,\ell} \geq \min_{i \in [n]} \text{chkcount}_{i,\ell},$$

since $\text{skcount}_{i,\ell} \geq \max_{\ell \in \mathbb{L}_{MM}} \text{chkcount}_{i,\ell}$.

Note. We slightly modify the gets to perform the label-based filtering at the client instead of doing it at the server. While this modification does not change our asymptotics, it is going to be essential to describe the changes we are going to make due to various concurrency issues. Moreover, in the final TST protocol, we will encrypt the labels which makes server-side filtering impossible.

5.5 A Concurrent Design

So far we designed our plaintext structure without considering concurrency. We now present several challenges that come up when the structure is accessed concurrently. Recall that our goal is to achieve linearizability which essentially means that the operations should appear to be interleaved at the granularity of complete operations, and the order of non-overlapping operations should be preserved. In simpler terms, we should be able to order the operations in a way that a get should always output all the values added by the append operations ordered before it. Additionally, if an operation finishes before another one starts, the former should be ordered before the latter.

5.5.1 Need for Atomic Instructions

Consider the following scenario involving two clients \mathbf{C}_1 and \mathbf{C}_2 . Suppose $\text{list}_{1,\ell}$ includes 100 nodes at addresses $\text{addr}_{1,\ell,1}, \dots, \text{addr}_{1,\ell,100}$ in dDX and assume the $(1, \ell)$ checkpoint address is $\text{addr}_{1,\ell,1}$. If \mathbf{C}_2 executes a get on label ℓ it retrieves the reachable nodes of $\text{list}_{1,\ell}$ and $\text{list}_{2,\ell}$ as well as the nodes inserted after the $(2, \ell)$ skip counter. \mathbf{C}_2 will then determine that $\text{node}_{1,\ell,100}$ is the head of $\text{list}_{1,\ell}$ and will send $\text{addr}_{1,\ell,100}$ to the server so that it can update the checkpoint dictionary cDX_1 . Now assume that before the server updates $cDX_1[\ell]$, the scheduler pauses the execution of the get operation and that \mathbf{C}_1 executes a hundred appends followed by one get all on label ℓ . The one hundred appends result in the creation of one hundred nodes with addresses $\text{addr}_{1,\ell,101}, \dots, \text{addr}_{1,\ell,200}$ and the get operation results in \mathbf{C}_1 sending the server a new checkpoint address $\text{addr}_{1,200}$ which the server will use to update the checkpoint dictionary cDX_1 . If the scheduler resumes \mathbf{C}_2 's get operation at this moment, the server will set $cDX_1[\ell]$ to $\text{addr}_{1,\ell,100}$ which is clearly wrong. For simplicity, we do not describe

the updates of the skip dictionaries, but the same issue applies, where it is possible to overwrite the correct skip counter value $\text{count}_{1,\ell,200}$ with an out-of-date value $\text{count}_{1,\ell,100}$.

As seen, with our current design it is possible to update the checkpoint dictionaries with addresses that are out of date and the consequence is that the next get operation for ℓ will return incorrect results. This is the case because the server will first retrieve the old checkpoint address $\text{addr}_{1,\ell,100}$ from cDX_1 , then recover the reachable nodes of $\text{list}_{1,\ell}$ from dDX starting at $\text{addr}_{1,100}$. It will then retrieve the skip counter $\text{count}_{1,\ell,200}$ from the skip dictionary skDX_1 and query the range dictionary RDX_1 which will return the addresses that were inserted after $\text{count}_{1,\ell,200}$. The server then uses these addresses to recover the remaining nodes from dDX but this set of nodes is incorrect because it is missing the nodes with addresses between $\text{addr}_{1,\ell,101}$ and $\text{addr}_{1,\ell,200}$.

Efficiency Issue. Now consider the scenario above except that the skip dictionaries are the ones updated with out-of-date counters instead of the checkpoint dictionaries. During a get on label ℓ , the server retrieves an old counter $\text{count}_{1,\ell,100}$ from the skip dictionary instead of the correct counter $\text{count}_{1,\ell,200}$. Given this counter, the server will query the range dictionary and recover the addresses added after $\text{count}_{1,\ell,100}$. But notice that the server already retrieved the nodes between $\text{addr}_{1,\ell,1}$ and $\text{addr}_{1,\ell,200}$ since the checkpoint dictionary was updated correctly so the server could potentially retrieve a large number of unnecessary nodes just to filter them out at the end. In particular, the structure could have get efficiency

$$\#\text{MM}[\ell] + \sum_{i \in [n]} \sum_{\ell' \neq \ell} \#\text{list}_{i,\ell'} [\text{count} \geq \text{acount}_i]$$

where acount_i is the counter value of the latest i -node at the time of an arbitrarily old get operation—instead of being exactly at the time of the previous get operation as intended. In the worst case, the efficiency of the get can be linear in the size of the multi-map.

Compare and Swap Operations. The problem that leads to the issues above is that the time at which the server was supposed to update the checkpoint dictionary cDX_1 for \mathbf{C}_2 's get on ℓ and the time at which the server actually updates it are distinct and during this interval of time the server receives and executes additional append and get operations from \mathbf{C}_1 . One possible solution is to make the server check whether the to-be-written checkpoint address is the latest one and only update the checkpoint dictionary if it is still so. This approach, however, suffers from the same issue above since the time between the check and the actual write are distinct. We solve this by making use of atomic operations and, specifically, dictionaries that support **CompareAndSwap** operations. These are dictionaries which, at a high level, execute a comparison and an update operation in one atomic step. The atomicity guarantees that there is no interruption between the comparison and the update so the approach mentioned above will solve the problem. We provide more details below.

The get operation now works the same as before, except for how the server updates the checkpoint and the skip dictionaries. We focus on the case of the checkpoint dictionaries but the same modifications apply to the skip dictionaries. When the server receives a new checkpoint address addr_ℓ^* , it first retrieves $\text{node}_\ell^* := \text{dDX}[\text{addr}_\ell^*]$. It then parses node_ℓ^* as $(v^*, \text{prevAddr}^*, \ell, \text{count}^*)$ and retrieves the node at the current checkpoint address addr_ℓ^\times which it parses as $(v^\times, \text{prevAddr}^\times, \ell, \text{count}^\times)$. If $\text{count}^\times < \text{count}^*$, the server executes

$$\text{CompareAndSwap}(\text{dDX}, \ell, \text{addr}_\ell^\times, \text{addr}_\ell^*).$$

If the operation fails, there was an update to the checkpoint dictionary so the server retries all the previous steps until **CompareAndSwap** is successful or until the current checkpoint address corresponds to a node that has a higher counter than the new one, i.e., when $\text{count}^* < \text{count}^\times$. We refer to dictionaries that support *cas* operations as *cas*-dictionaries and denote such schemes as $\Delta_{\overline{\text{DX}}}$.

5.5.2 Need for Locking

Recall that our goal is to design a linearizable encrypted multi-map which means that we need to ensure that all possible execution histories are linearizable. In the following, we show that our construction so far is not linearizable which leads to situations in which concurrent gets for the same label have incoherent responses; that is, neither is a subset of the other.

First, we introduce some useful terminology. Consider an append and get operation on two possibly distinct labels ℓ and ℓ' , respectively, and let node_ℓ be the node inserted into the data dictionary by the append operation. We say that the get *sees the append* if the client who initiated the get on ℓ' retrieves node_ℓ . Recall that as part of a get, a client retrieves all the nodes that are reachable through the checkpoint dictionaries and the nodes with addresses returned by the range queries on the range dictionaries RDX_i , for $i \in [n]$. Moreover, we say that the get *outputs the append* if the get outputs the value of the node that was inserted by the append operation. Note that it is possible for a get to see an append but to not output it. This can occur due to the client-side filtering step when the non- ℓ nodes.

Why the Structure is not Linearizable. Assume there are four clients $\mathbf{C}_1, \dots, \mathbf{C}_4$ and let $\text{get}_{1,\ell}$ and $\text{get}_{2,\ell}$ be two concurrent gets on label ℓ initiated by \mathbf{C}_1 and \mathbf{C}_2 , respectively. For this example, we solely focus on the accesses to the range dictionaries. In particular, consider the part where $\text{get}_{1,\ell}$ and $\text{get}_{2,\ell}$ access the range dictionaries RDX_3 and RDX_4 corresponding to \mathbf{C}_3 and \mathbf{C}_4 in the following order: first, $\text{get}_{1,\ell}$ accesses RDX_3 , then $\text{get}_{2,\ell}$ accesses RDX_3 and RDX_4 and, finally, $\text{get}_{1,\ell}$ accesses RDX_4 . We now show that the responses of the two gets are incoherent which breaks linearizability. In the example above, $\text{get}_{2,\ell}$ accesses RDX_3 after $\text{get}_{1,\ell}$ so it is possible that $\text{get}_{2,\ell}$ sees and outputs appends from \mathbf{C}_3 that $\text{get}_{1,\ell}$ does not see. This could happen, for instance, if \mathbf{C}_3 executes an append after $\text{get}_{1,\ell}$ finishes reading RDX_3 . Similarly, since $\text{get}_{1,\ell}$ reads RDX_4 after $\text{get}_{2,\ell}$, it could see and output appends from \mathbf{C}_4 that $\text{get}_{2,\ell}$ does not see. Therefore, in

this execution there are appends that one get will output but that the other does not. However, for linearization we must be able to order the two gets such that the second get outputs (at least) the responses of the first get. Since neither $\text{get}_{1,\ell}$ nor $\text{get}_{2,\ell}$ have responses that are a superset of the other, they cannot be ordered appropriately and, therefore, the execution is not linearizable.

Coarse-Grained Locking. The reason the outputs of the gets do not have the superset structure required for linearizability is that they do not synchronize on their access to the range dictionaries. In particular, the concurrent gets can access the range dictionaries in an interleaved manner which leads to the incoherent outputs described above. To synchronize the gets' accesses, we could wrap all the range dictionaries under one big lock and require the gets to acquire the lock before accessing them. This would solve the issue since there can be no interleaved accesses but now a get might need to wait until the lock is released by another concurrent get which can greatly decrease throughput.

Hand-Over-Hand Locking. Instead of using coarse-grained locking, we solve the interleaving problem using a more granular form of locking called *hand-over-hand locking*. More precisely, given two get operations $\text{get}_{1,\ell}$ and $\text{get}_{2,\ell}$ on the same label ℓ , if $\text{get}_{1,\ell}$ accesses RDX_i before $\text{get}_{2,\ell}$, we need to ensure that $\text{get}_{1,\ell}$ accesses RDX_{i+1} before $\text{get}_{2,\ell}$ does, and this has to hold for all $i \in [n - 1]$. Accessing the range dictionaries prevents any form of interleaving and can be achieved as follows. Instead of a single monolithic lock, we use a series of locks $\text{rLock}_1, \dots, \text{rLock}_n$, one per range dictionary. When a get acquires rLock_i it then queries RDX_i but only releases the lock when it acquires the next lock RDX_{i+1} . Hand-over-hand locking leads to much better throughput.

5.5.3 Synchronizing the Gets and Appends

So far, we focused on the various synchronization issues of concurrent get operations and showed how to extend our structure with atomic instructions and hand-over-hand locking. We now highlight other synchronization issues between get and append operations which can also result in non-linearizable execution histories. Consider three clients \mathbf{C}_1 , \mathbf{C}_2 and \mathbf{C}_3 such that \mathbf{C}_3 executes $\text{get}_{3,\ell}$ which queries RDX_1 . After accessing RDX_1 but before accessing RDX_2 , \mathbf{C}_3 's get is paused and \mathbf{C}_1 executes an append $\text{append}_{1,\ell}$ followed by $\text{append}_{2,\ell}$ executed by \mathbf{C}_2 . After the two appends, server resumes \mathbf{C}_3 's get and accesses RDX_2 where it sees and outputs $\text{append}_{2,\ell}$ but not $\text{append}_{1,\ell}$ since it did not see it at the time it queried RDX_1 . The execution history in this example is not linearizable. To see why, observe that even though $\text{append}_{1,\ell}$ precedes $\text{append}_{2,\ell}$, the get outputs $\text{append}_{2,\ell}$ but not $\text{append}_{1,\ell}$ which violates the superset structure necessary for linearizability.

Using Counters to Synchronize. The problem above is due to the fact that the get and append operations are not synchronizing their accesses to the range

dictionaries. More concretely, the gets have no way of knowing if they missed an append operation in a range dictionary that they have already read. Similarly to the previous problem, this issue can be solved with coarse-grained locking; specifically, by requiring that every append and get acquire a lock on all the range dictionaries. However, as discussed, this approach would affect throughput. Instead, we synchronize get and append operations with a counter as follows. The idea is to use a linearizable counter to assign a time to a get operation in such a way that clients can distinguish between appends that are before and after the get. Recall that the appends already make use of a linearizable counter count_g to timestamp the nodes. We now make use of count_g to generate a counter value $\text{count}_{\text{get}}$ whenever a client starts executing a get. More precisely, the gets generate a counter value right before they try to acquire the lock for RDX_1 . Then, if they see an append with a larger counter than $\text{count}_{\text{get}}$, they discard it and do not output it. This avoids situations where a get outputs a later append, but misses an earlier one because it did not see that append. With this extension, and going back to the example above, the get operation will not see either of the appends, $\text{append}_{1,\ell}$ and $\text{append}_{2,\ell}$, which solves the linearizability issue.

Final Details. The structure so far is almost complete except for one detail. The new counter described above and the client-side filtering may some times violate the superset structure required for linearizability.³ Now that the counter is part of the get operation, the synchronization between the gets achieved with hand-over-hand locking is violated and a total order can no longer be attained. To solve this, we make a simple extension and wrap the counter count_g with a lock cLock . We also ensure that cLock and $\text{rLock}_1, \dots, \text{rLock}_n$ are connected through hand-over-hand locking in the sense that a get operation first needs to acquire the cLock and then wait to acquire rLock_1 , then rLock_2 and so on and so forth.

6 TST: A Linearizable Multi-map Encryption Scheme

Our construction $\text{TST} = (\text{Init}, \text{Get}, \text{Append})$ makes black-box use of a linearizable dictionary $\Delta_{\text{DX}} = (\text{Init}, \text{Get}, \text{Put})$, a linearizable cas-dictionary $\Delta_{\text{DX}} = (\text{Init}, \text{Get}, \text{Put}, \text{CompareAndSwap})$, a linearizable range dictionary $\Delta_{\text{RDX}} = (\text{Init}, \text{Put}, \text{GetGreater})$, a linearizable counter $\Delta_{\text{CTR}} = (\text{Init}, \text{FetchAndInc})$, a pseudo-random function $F : \{0, 1\}^k \times \{0, 1\}^* \rightarrow \{0, 1\}^k$ and a symmetric encryption scheme $\text{SKE} = (\text{Gen}, \text{Enc}, \text{Dec})$. Due to space limitations, the details of the scheme are in the full version. At a high level, it works as follows.

Init. The init protocol is executed between a trusted party \mathbf{T} , n clients $\mathbf{C}_1, \dots, \mathbf{C}_n$ and the server \mathbf{S} . All parties input the security parameter k . First, the trusted party \mathbf{T} samples two keys $K_e, K_s \xleftarrow{\$} \{0, 1\}^k$ and sends them to all clients. For all

³ This issue is very similar to the one discussed in Sect. 5.5.2 so we do not expand on it in more detail.

$i \in [n]$, client \mathbf{C}_i instantiates a state dictionary $\text{sDX}_i \leftarrow \Delta_{\text{DX}}.\text{Init}(\theta, 2k)$ that will be updated during append operations and that maps labels ℓ to a pair composed of (1) the address of the head of the linked list $\text{list}_{i,\ell}$ and (2) the key that encrypts the head. The server \mathbf{S} initializes a data dictionary $\text{dDX} \leftarrow \Delta_{\text{DX}}.\text{Init}(k, \theta + \lambda + 3k)$. For all $i \in [n]$, \mathbf{S} initializes a checkpoint dictionary $\text{cDX}_i \leftarrow \Delta_{\text{DX}}.\text{Init}(k, k)$, a skip dictionary $\text{skDX}_i \leftarrow \Delta_{\text{DX}}.\text{Init}(k, k)$, a range dictionary $\text{RDX}_i \leftarrow \Delta_{\text{RDX}}.\text{Init}(k, k)$, and a range lock $\text{rLock}_i := \Lambda.\text{Init}(\cdot)$. The server \mathbf{S} also initializes a counter lock $\text{cLock} := \Lambda.\text{Init}(\cdot)$. Finally, it outputs the encrypted multi-map

$$\text{EMM} := (\text{dDX}, (\text{cDX}_i)_{i \in [n]}, (\text{RDX}_i)_{i \in [n]}, (\text{skDX}_i)_{i \in [n]}, \text{count}_{\mathbf{g}}, \text{cLock}, (\text{rLock}_i)_{i \in [n]}).$$

Append. The append protocol is executed between a client \mathbf{C}_i and the server \mathbf{S} . It takes as input a key K , a state st , a label ℓ and a value v from the client and the encrypted multi-map EMM from the server. It is a single-round protocol that works as follows:

- (*client*) The client first retrieves from sDX_i the address dtag_{ℓ^-} of the head of $\text{list}_{i,\ell}$ and its corresponding key K_{ℓ^-} . If this is the first time \mathbf{C}_i appends a value for ℓ , the entry in the state dictionary will be empty and the client sets both dtag_{ℓ^-} and K_{ℓ^-} to \perp . The client then samples a new data tag $\text{dtag}_{\ell} \xleftarrow{\$} \{0, 1\}^k$ and a new key $K_{\ell} \xleftarrow{\$} \{0, 1\}^k$ and creates a new node composed of five ciphertexts

$$(\text{ct}_{\ell}, \text{ct}_v, \text{ct}_{\ell^-}, \text{ct}_{K_{\ell^-}}, \text{ct}_{K_{\ell}}),$$

where $\text{ct}_{\ell} := \text{Enc}_{K_e}(\ell)$ is an encryption of the label with key K_e , $\text{ct}_v := \text{Enc}_{K_e}(v)$ is an encryption of the value under key K_e , $\text{ct}_{\ell^-} := \text{Enc}_{K_{\ell^-}}(\text{dtag}_{\ell^-})$ is an encryption of the previous data tag under the new key K_{ℓ} , $\text{ct}_{K_{\ell^-}} := \text{Enc}_{K_{\ell}}(K_{\ell^-})$ is an encryption of previous key under the new key K_{ℓ} , and $\text{ct}_{K_{\ell}} := \text{Enc}_{K_e}(K_{\ell})$ is an encryption of the new key under key K_e . The client then updates the state dictionary with the new data tag and key and finally sends to the server an append token $\text{atk} := (\text{dtag}_{\ell}, \text{node}_{i,\ell})$ composed of the new data tag along with the encrypted node.

- (*server*) Once the server receives the append token atk , it first retrieves and increments the global counter $\text{count} \leftarrow \Delta_{\text{CTR}}.\text{FetchAndInc}(\text{count}_{\mathbf{g}})$. It then appends the counter value to the encrypted node by setting $\text{node}_{i,\ell} := (\text{node}_{i,\ell}, \text{count})$. The server then inserts the encrypted node $\text{node}_{i,\ell}$ in the data dictionary at the address provided by the client by computing $\text{dDX} \leftarrow \Delta_{\text{DX}}.\text{Put}(\text{dDX}, \text{dtag}, \text{node})$. It also inserts the pair $(\text{count}, \text{dtag}_{\ell})$ in the range dictionary by computing $\text{RDX}_i \leftarrow \Delta_{\text{RDX}}.\text{Put}(\text{RDX}_i, \text{count}, \text{dtag}_{\ell})$. Finally, the server outputs the updated encrypted multi-map.

Get. The get protocol is executed between a client \mathbf{C}_i and the server \mathbf{S} . The protocol takes as input a key K , a state st , a label ℓ from \mathbf{C}_i and the encrypted multi-map from \mathbf{S} . The get protocol is a three-round protocol that works as follows:

- (*client round 1*) The client first generates the checkpoint tag $\text{ctag}_{\ell,j}$ for label ℓ and all $j \in [n]$ by computing $\text{ctag}_{\ell,j} := F[K_s, \ell, j, 1]$. The client then sends to the server the get token $\text{gtk}_1 := (\text{ctag}_{\ell,j})_{j \in [n]}$ composed of all checkpoint tags;
- (*server round 1*) Once the server receives the first get token gtk_1 , it initializes an empty set \mathbf{R} and retrieves the data tag dtag_j from the checkpoint dictionary by computing, for all $j \in [n]$, $\text{dtag}_j \leftarrow \Delta_{\text{DX}}.\text{Get}(\text{cDX}_j, \text{ctag}_j)$. If $\text{dtag}_j \neq \perp$, the server retrieves the corresponding node node_j from the data dictionary by computing $\text{node}_j \leftarrow \Delta_{\text{DX}}.\text{Get}(\text{dDX}, \text{dtag}_j)$, and appends it to \mathbf{R} . Note that the absence of a tag dtag_j simply means that either the j th client never appended a value for ℓ or that the ongoing get is the first get initiated by any client. The server sends the set \mathbf{R} to \mathbf{C}_i .
- (*client round 2*) For all $j \in [n]$, the i th client parses node_j in \mathbf{R} and decrypts the ciphertext of the previous key by computing $K_{\ell-,j} := \text{Dec}_{K_e}(\text{ct}_{K_{\ell-}})$. The client also computes the checkpoint tag ctag_j as well as the skip tag sktag_j by computing

$$\text{ctag}_{\ell,j} := F[K_s, \ell, j, 1] \quad \text{and} \quad \text{sktag}_{\ell,j} := F[K_s, \ell, j, 2].$$

The client sends to the server the second get token gtk_2 composed of the old key $K_{\ell-,j}$, the checkpoint tag $\text{ctag}_{\ell,j}$ and the skip tag $\text{sktag}_{\ell,j}$ for all $j \in [n]$.

- (*server round 2*) Once the server receives the second get token gtk_2 , it initializes n empty sets $(\mathbf{R}_j)_{j \in [n]}$, retrieves the counter from the skip dictionaries and the data tag from the checkpoint dictionary by computing

$$\text{skcount}_j \leftarrow \Delta_{\text{DX}}.\text{Get}(\text{skDX}_j, \text{sktag}_{\ell,j}) \quad \text{and} \quad \text{dtag}_j \leftarrow \Delta_{\text{DX}}.\text{Get}(\text{cDX}_j, \text{ctag}_{\ell,j}).$$

The server then traverses $\text{list}_{j,\ell}$ starting from the head node located at dtag_j and populates the result sets \mathbf{R}_j , for $j \in [n]$, as follows. It first retrieves the node from the data dictionary by computing $\text{node}_j \leftarrow \Delta_{\text{DX}}.\text{Get}(\text{dDX}, \text{dtag}_j)$, adds the pair $(\text{dtag}_j, \text{node}_j)$ to \mathbf{R}_j , parses the node as $(\text{ct}_\ell, \text{ct}_v, \text{ct}_{\ell-}, \text{ct}_{K_{\ell-}}, \text{ct}_{K_\ell}, \text{count})$, decrypts the ciphertext of the previous data tag by computing $\text{dtag}_j^- := \text{Dec}_{K_{\ell-,j}}(\text{ct}_{\ell-})$ (which becomes the new head), and decrypts the ciphertext of the previous key $K_{\ell-,j} := \text{Dec}_{K_{\ell-,j}}(\text{ct}_{K_{\ell-}})$ (which becomes the new key). The server reiterates this process until it reaches a data tag dtag_j^- equal to \perp .

The server also accesses the range dictionary to retrieve the data tags. In particular, it first waits and acquires the counter lock cLock , retrieves and increments the counter $\text{count}_{\text{Get}} \leftarrow \Delta_{\text{CTR}}.\text{FetchAndInc}(\text{count}_{\text{g}})$ and then unlocks cLock . For all $j \in [n]$, the server then waits and acquires the range lock of the range dictionary rLock_j , retrieves all the data tags that have a counter larger than skcount_j such that $\mathbf{r}_j \leftarrow \Delta_{\text{RDX}}.\text{GetGreater}(\text{RDX}_j, \text{skcount}_j)$, releases the lock, and then retrieves all the nodes from the data dictionary located at the corresponding data tags. Note that the acquisition and release of the lock follows a hand-over-hand locking mechanism (refer to Sect. 5.5.2

- for more details). In particular, for all $\mathbf{dtag} \in \mathbf{r}_j$, the server computes $\mathbf{node} \leftarrow \Delta_{\text{DX}}.\text{Get}(\mathbf{dDX}, \mathbf{dtag})$ and adds $(\mathbf{dtag}, \mathbf{node})$ to \mathbf{R}_j . Finally, the server sends to the client $(\mathbf{R}_j)_{j \in [n]}$ along with the get counter $\text{count}_{\text{Get}}$.
- (*client round 3*) In this round, the client filters the nodes and only keeps the ones that need to be part of final response. The client also computes, for all $j \in [n]$, the new head \mathbf{dtag}_j^* of the linked list to be stored in the checkpoint dictionary and the most recent counter skcount_j to be stored in the skip dictionary. To filter the nodes, it initializes a set \mathbf{v} and for all $j \in [n]$, performs the following steps. For all $(\mathbf{dtag}_z, \mathbf{node}_z) \in \mathbf{R}_j$, it parses the node as $(\text{ct}_{\ell,z}, \text{ct}_{v,z}, \text{ct}_{\ell^-,z}, \text{ct}_{K_{\ell^-,z}}, \text{ct}_{K_{\ell,z}}, \text{count}_z)$ and decrypts the ciphertext of the label by computing $\ell_z := \text{Dec}_{K_e}(\text{ct}_{\ell,z})$. If $\ell_z = \ell$ and count_z is smaller than $\text{count}_{\text{Get}}$, the client computes $v := \text{Dec}_{K_e}(\text{ct}_{\ell,z})$ and adds it to \mathbf{v} . Note that the second condition is necessary to synchronize between the get and the append operations which is crucial for linearizability as discussed in Sect. 5.5.3. As a second step, client computes the new head of the linked list, \mathbf{dtag}_j^* , by first identifying the node for label ℓ with the largest counter and then setting $\mathbf{dtag}_j^* := \mathbf{dtag}_{z^*}$, $\text{chkcount}_j := \text{count}_{z^*}$, where

$$z^* := \arg \max_{z \in \mathbf{Z}} (\text{count}_z) \quad \text{and} \quad \mathbf{Z} = \{z \in [|\mathbf{R}_j|] : \ell_z = \ell\}.$$

For the skip counter, the client needs to identify the node with the largest counter irrespective of the underlying label, i.e., $\text{skcount}_j := \max_z (\text{count}_z)$. The client also computes the checkpoint tag as well as the skip tag

$$\text{ctag}_{\ell,j} := F[K_s, \ell, j, 1] \quad \text{and} \quad \text{shtag}_{\ell,j} := F[K_s, \ell, j, 2],$$

which are necessary to update the j th checkpoint dictionary and the j th skip dictionary. Finally, the client sends the third get token \mathbf{gtk}_3 which is composed of the checkpoint tag $\text{ctag}_{\ell,j}$, the checkpoint counter chkcount_j and the new data tag \mathbf{dtag}_j^* which is the address of the new head of the linked list. The token also includes of the skip counter skcount_j as well as the skip tag $\text{shtag}_{\ell,j}$, for all $j \in [n]$.

- (*server round 3*) Once the server receives the third get token, it updates the checkpoint dictionary as well as the skip dictionary. In particular, for all $j \in [n]$, the server first retrieves the old data tag by computing $\mathbf{dtag}_j^{\times} \leftarrow \Delta_{\text{DX}}.\text{Get}(\mathbf{cDX}_j, \text{ctag}_j)$. It then retrieves the corresponding node from the data dictionary from which it extracts the counter count_j . The server only updates the checkpoint dictionary if the old counter count_j is strictly smaller than the new checkpoint counter chkcount_j . For this, it makes use of the compare and swap atomic instruction $\text{CompareAndSwap}(\mathbf{cDX}_j, \text{ctag}_j, \mathbf{dtag}_j^{\times}, \mathbf{dtag}_j^*)$, so that $\mathbf{cDX}_j[\text{ctag}_j]$ is updated to \mathbf{dtag}_j^* if and only if $\mathbf{cDX}_j[\text{ctag}_j] = \mathbf{dtag}_j^{\times}$. If the CompareAndSwap fails, the server performs the same steps as above until $\text{count}_j \geq \text{chkcount}_j$. It performs the same steps to update the skip dictionary; i.e., for all $j \in [n]$, it computes $\text{CompareAndSwap}(\mathbf{skDX}_j, \text{shtag}_j, \text{count}_j^{\times}, \text{skcount}_j)$, where count_j^{\times} is the old counter in the skip dictionary such that $\text{count}_j^{\times} \leftarrow \Delta_{\text{DX}}.\text{Get}(\mathbf{skDX}_j, \text{shtag}_j)$.

The client finally outputs the final response \mathbf{v} whereas the server outputs the updated encrypted multi-map EMM.

Making TST Fully-Dynamic. TST can be extended to support delete operations using *lazy deletion* which has been used in many dynamic multi-map encryption constructions [8, 9, 15, 28, 59]. The lazy deletion of a label/value pair (ℓ, v) is implemented using an append of (ℓ, v) with an additional delete marker. During gets, the client retrieves both pairs and uses the delete markers to filter out the deleted values. In the context of TST, the client retrieves the nodes of the appended and deleted pairs and filters out the deleted values as follows. For each append node with value v , if there is a delete node with value v but a greater counter, then it removes v from the response; otherwise, it keeps v . The construction can be made linearizable with a minor change to the append protocol on the server side. When a client C_i sends a new node to the server, it uses hand-over-hand locking over cLock and rLock_i to increment count_g and to update RDX_i . Unfortunately, this makes appends and deletes deadlock-free instead of lock-free. All the operations linearize at the time they lock cLock . Due to space constraints, the proof will appear in the full version of this work. We leave it as future work to design a linearizable lock-free fully-dynamic encrypted multi-map.

7 Efficiency, Linearizability and Security of TST

Efficiency Analysis. Due to space constraints, we defer the efficiency analysis of TST (including time, storage, round complexity, and progress guarantees) to the full version. We first analyze the asymptotic behavior of TST in a black-box manner, and then, examine its concrete efficiency by considering specific instantiations of the underlying plaintext data structures. Additionally, we investigate both the worst-case and best-case scenarios for the get complexity.

Linearizability. We show that TST is linearizable. In particular, we first introduce a linearizable procedure LZP (in the full version) which defines the linearization points for all possible execution histories H in TST. As a second step, given the output of the linearizable procedure we prove that TST verifies both the *span membership* and the *correctness* conditions described in Definition 2. While proving span membership is relatively straightforward, proving correctness is more challenging as it requires showing that for all append operations append_ℓ and get operations get_ℓ for label ℓ in the history H , the inequality, $\text{linp}(\text{append}_\ell) < \text{linp}(\text{get}_\ell)$ holds if and only if the appended value in append_ℓ is part of the output of the get operation get_ℓ . Conversely, if the append operation is not part of the output of the get operation, then append_ℓ should be linearized after get_ℓ . Due to space constraints, we defer the details to the full version, and only state the main result here.

Corollary 1. *If Δ_{DX} , $\Delta_{\overline{\text{DX}}}$, Δ_{RDX} and Δ_{CTR} are linearizable, then all execution histories H are linearizable and, therefore, TST is linearizable.*

Security. We describe TST’s instruction-level leakage \mathcal{L} . During an append operation, there is no leakage. During get operations, TST leaks the operation equality pattern, i.e., correlations between operations on the same label. In the first round of server communication, when the server receives the first get token gtk_1 , it infers correlations with get operations that have the same label and for which the server has also received the first get token. This is because the first get token for two gets that query the same label will be the same. Next, during the third round, when the server receives new checkpoint addresses in gtk_3 , it learns new correlations between the current get and append operations. These correlations are with the appends that added their counters to the range dictionary before the get operation read that range dictionary. This is because when the server sees the new checkpoint address, it learns that all these nodes have the same label as the current get operation. Due to space constraints, a formal and detailed description of the leakage profile \mathcal{L} and proof of the theorem are in the full version.

Theorem 1. *If F is pseudo-random, SKE is a CPA-secure, and Δ_{DX} , $\Delta_{\overline{\text{DX}}}$, Δ_{RDX} , and Δ_{CTR} are all linearizable, then TST is (χ_{12}, \mathcal{L}) -secure in the random oracle model.*

Note. In the sequential setting, our construction achieves *forward privacy* which has been extensively studied in the STE literature and is the standard security goal for dynamic encrypted multi-maps. As pointed out in [63], forward privacy can protect against certain injection attacks but recent work [4] has shown that forward privacy has some limitations. In the concurrent setting, there are no standard security notions for leakage profiles and, as far as we know, there are no concurrent-specific leakage attacks (e.g., that also exploit adversarial scheduling), therefore we do not have a baseline for comparison other than the one in the sequential setting.

References

1. Agarwal, A., Kamara, S.: Encrypted distributed hash tables. Tech. Rep. 2019/1126, IACR ePrint Cryptography Archive (2019), <https://eprint.iacr.org/2019/1126.pdf>
2. Agarwal, A., Kamara, S.: Encrypted key-value stores. In: Progress in Cryptology–INDOCRYPT 2020: 21st International Conference on Cryptology in India, Bangalore, India, December 13–16, 2020, Proceedings 21. pp. 62–85. Springer (2020)
3. Amjad, G., Kamara, S., Moataz, T.: Structured encryption secure against file injection attacks (2021), (under submission at CRYPTO ’21)
4. Amjad, G., Kamara, S., Moataz, T.: Injection-secure structured and searchable symmetric encryption. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 232–262. Springer (2023)
5. Asharov, G., Naor, M., Segev, G., Shahaf, I.: Searchable symmetric encryption: Optimal locality in linear space via two-dimensional balanced allocations. In: ACM Symposium on Theory of Computing (STOC ’16). pp. 1101–1114. STOC ’16, ACM, New York, NY, USA (2016). <https://doi.org/10.1145/2897518.2897562>, <http://doi.acm.org/10.1145/2897518.2897562>

6. Asharov, G., Komargodski, I., Lin, W.K., Peserico, E., Shi, E.: Optimal oblivious parallel RAM. In: Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA). pp. 2459–2521. SIAM (2022)
7. Asharov, G., Segev, G., Shahaf, I.: Tight tradeoffs in searchable symmetric encryption. In: Annual International Cryptology Conference. pp. 407–436. Springer (2018)
8. Bost, R.: Sophos - forward secure searchable encryption. In: ACM Conference on Computer and Communications Security (CCS '16) (2016)
9. Bost, R., Minaud, B., Ohrimenko, O.: Forward and backward private searchable encryption from constrained cryptographic primitives. In: ACM Conference on Computer and Communications Security (CCS '17) (2017)
10. Boyle, E., Chung, K.M., Pass, R.: Oblivious parallel RAM and applications. In: Theory of Cryptography Conference. pp. 175–204. Springer (2015)
11. Bronson, N.G., Casper, J., Chafi, H., Olukotun, K.: A practical concurrent binary search tree. *ACM Sigplan Notices* **45**(5), 257–268 (2010)
12. Canetti, R.: Security and composition of multi-party cryptographic protocols. *Journal of Cryptology* **13**(1) (2000)
13. Cash, D., Jarecki, S., Jutla, C., Krawczyk, H., Rosu, M., Steiner, M.: Highly-scalable searchable symmetric encryption with support for boolean queries. In: Advances in Cryptology - CRYPTO '13. Springer (2013)
14. Cash, D., Tessaro, S.: The locality of searchable symmetric encryption. In: Advances in Cryptology - EUROCRYPT 2014 (2014)
15. Cash, D., Jaeger, J., Jarecki, S., Jutla, C., Krawczyk, H., Rosu, M., Steiner, M.: Dynamic searchable encryption in very-large databases: Data structures and implementation. In: Network and Distributed System Security Symposium (NDSS '14) (2014)
16. Cash, D., Ng, R., Rivkin, A.: Improved structured encryption for sql databases via hybrid indexing. In: Applied Cryptography and Network Security: 19th International Conference, ACNS 2021, Kamakura, Japan, June 21–24, 2021, Proceedings, Part II. pp. 480–510. Springer (2021)
17. Chan, T.H.H., Chung, K.M., Shi, E.: On the depth of oblivious parallel RAM. In: Advances in Cryptology—ASIACRYPT 2017: 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I 23. pp. 567–597. Springer (2017)
18. Chan, T.H.H., Guo, Y., Lin, W.K., Shi, E.: Oblivious hashing revisited, and applications to asymptotically efficient ORAM and OPRAM. In: Advances in Cryptology—ASIACRYPT 2017: 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I 23. pp. 660–690. Springer (2017)
19. Chan, T.H.H., Nayak, K., Shi, E.: Perfectly secure oblivious parallel RAM. In: Theory of Cryptography Conference. pp. 636–668. Springer (2018)
20. Chase, M., Kamara, S.: Structured encryption and controlled disclosure. In: Advances in Cryptology - ASIACRYPT '10. Lecture Notes in Computer Science, vol. 6477, pp. 577–594. Springer (2010)
21. Chase, M., Kamara, S.: Structured encryption and controlled disclosure. Tech. Rep. 2011/010.pdf, IACR Cryptology ePrint Archive (2010)
22. Chen, B., Lin, H., Tessaro, S.: Oblivious parallel RAM: improved efficiency and generic constructions. In: Theory of Cryptography: 13th International Conference, TCC 2016-A, Tel Aviv, Israel, January 10-13, 2016, Proceedings, Part II 13. pp. 205–234. Springer (2016)

23. Curtmola, R., Garay, J., Kamara, S., Ostrovsky, R.: Searchable symmetric encryption: Improved definitions and efficient constructions. In: ACM Conference on Computer and Communications Security (CCS '06). pp. 79–88. ACM (2006)
24. Demertzis, I., Papamanthou, C.: Fast searchable encryption with tunable locality. In: ACM International Conference on Management of Data (SIGMOD '17). pp. 1053–1067. SIGMOD '17, ACM, New York, NY, USA (2017). <https://doi.org/10.1145/3035918.3064057>, <http://doi.acm.org/10.1145/3035918.3064057>
25. Demertzis, I., Papadopoulos, D., Papamanthou, C.: Searchable encryption with optimal locality: Achieving sublogarithmic read efficiency. In: Advances in Cryptology - CRYPTO '18. pp. 371–406. Springer (2018)
26. Ellen, F., Fatourou, P., Ruppert, E., van Breugel, F.: Non-blocking binary search trees. In: Proceedings of the 29th ACM SIGACT-SIGOPS symposium on Principles of distributed computing. pp. 131–140 (2010)
27. Ellis, C.S.: Concurrency in linear hashing. ACM Transactions on Database Systems (TODS) **12**(2), 195–217 (1987)
28. Etemad, M., Küpccü, A., Papamanthou, C., Evans, D.: Efficient dynamic searchable encryption with forward privacy. PoPETs **2018**(1), 5–20 (2018). <https://doi.org/10.1515/popets-2018-0002>, <https://doi.org/10.1515/popets-2018-0002>
29. Faber, S., Jarecki, S., Krawczyk, H., Nguyen, Q., Rosu, M., Steiner, M.: Rich queries on encrypted data: Beyond exact matches. In: European Symposium on Research in Computer Security (ESORICS '15). Lecture Notes in Computer Science. vol. 9327, pp. 123–145 (2015)
30. Garg, S., Mohassel, P., Papamanthou, C.: TWORAM: efficient oblivious RAM in two rounds with applications to searchable encryption. In: Advances in Cryptology - CRYPTO 2016. pp. 563–592 (2016). https://doi.org/10.1007/978-3-662-53015-3_20, https://doi.org/10.1007/978-3-662-53015-3_20
31. George, M., Kamra, S., Moataz, T.: Structured encryption and dynamic leakage suppression. In: Advances in Cryptology - EUROCRYPT 2021 (2021)
32. Greenwald, M.: Two-handed emulation: how to build non-blocking implementations of complex data-structures using dcas. In: Proceedings of the twenty-first annual symposium on Principles of distributed computing. pp. 260–269 (2002)
33. Hahn, F., Kerschbaum, F.: Searchable encryption with secure and efficient updates. In: ACM Conference on Computer and Communications Security (CCS '14). pp. 310–320. CCS '14, ACM, New York, NY, USA (2014). <https://doi.org/10.1145/2660267.2660297>, <http://doi.acm.org/10.1145/2660267.2660297>
34. Herlihy, M., Shavit, N., Luchangco, V., Spear, M.: The art of multiprocessor programming. Newnes (2020)
35. Herlihy, M.P., Wing, J.M.: Linearizability: A correctness condition for concurrent objects. ACM Transactions on Programming Languages and Systems (TOPLAS) **12**(3), 463–492 (1990)
36. Hsu, M., Yang, W.P.: Concurrent operations in extendible hashing. In: VLDB. vol. 86, pp. 25–28 (1986)
37. Hubert Chan, T.H., Shi, E.: Circuit OPRAM: Unifying statistically and computationally secure orams and oprams. In: Theory of Cryptography Conference. pp. 72–107. Springer (2017)
38. Kamara, S., Moataz, T.: Boolean searchable symmetric encryption with worst-case sub-linear complexity. In: Advances in Cryptology - EUROCRYPT '17 (2017)
39. Kamara, S., Moataz, T.: SQL on Structurally-Encrypted Data. In: Asiacrypt (2018)
40. Kamara, S., Papamanthou, C.: Parallel and dynamic searchable symmetric encryption. In: Financial Cryptography and Data Security (FC '13) (2013)

41. Kamara, S., Papamanthou, C., Roeder, T.: Dynamic searchable symmetric encryption. In: ACM Conference on Computer and Communications Security (CCS '12). ACM Press (2012)
42. Kamara, S., Moataz, T.: Design and analysis of ost. Tech. rep., MongoDB (2022)
43. Kamara, S., Moataz, T., Ohrimenko, O.: Structured encryption and leakage suppression. In: Advances in Cryptology - CRYPTO '18 (2018)
44. Katz, J., Lindell, Y.: Introduction to Modern Cryptography. Chapman & Hall/CRC (2008)
45. Kim, K.S., Kim, M., Lee, D., Park, J.H., Kim, W.H.: Forward secure dynamic searchable symmetric encryption with efficient updates. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. pp. 1449–1463 (2017)
46. Korenfeld, B.: CBTree: a Practical Concurrent Self-adjusting Search Tree. University of Tel-Aviv (2012)
47. Kumar, V.: Concurrent operations on extendible hashing and its performance. Communications of the ACM **33**(6), 681–694 (1990)
48. Lamport: How to make a multiprocessor computer that correctly executes multiprocess programs. IEEE transactions on computers **100**(9), 690–691 (1979)
49. Lea, D.: Hash table util. concurrent. concurrenthashmap, revision 1.3. JSR-166, the proposed Java Concurrency Package (2003)
50. Michael, M.M.: High performance dynamic lock-free hash tables and list-based sets. In: Proceedings of the fourteenth annual ACM symposium on Parallel algorithms and architectures. pp. 73–82 (2002)
51. Microsoft: ConcurrentDictionary.TryUpdate Method. <https://learn.microsoft.com/en-us/dotnet/api/system.collections.concurrent.concurrentdictionary-2.tryupdate?view=net-8.0>, accessed: September 16, 2024
52. Miers, I., Mohassel, P.: Io-dsse: Scaling dynamic searchable encryption to millions of indexes by improving locality. Cryptology ePrint Archive, Report 2016/830 (2016), <http://eprint.iacr.org/2016/830>
53. Nayak, K., Katz, J.: An oblivious parallel RAM with $O(\log^2 N)$ parallel runtime
54. Oracle: ConcurrentMap.computeIfPresent Method. <https://docs.oracle.com/javase/8/docs/api/java/util/concurrent/ConcurrentMap.html#computeIfPresent-K-java.util.function.BiFunction->, accessed: September 16, 2024
55. Pappas, V., Krell, F., Vo, B., Kolesnikov, V., Malkin, T., Choi, S.G., George, W., Keromytis, A., Bellovin, S.: Blind seer: A scalable private dbms. In: Security and Privacy (SP), 2014 IEEE Symposium on. pp. 359–374. IEEE (2014)
56. Popovitch, G.: parallel-hashmap: modify_if() function. <https://github.com/greg7mdp/parallel-hashmap/tree/8a889d3699b3c09ade435641fb034427f3fd12b6>, accessed: September 16, 2024
57. Shalev, O., Shavit, N.: Split-ordered lists: Lock-free extensible hash tables. Journal of the ACM (JACM) **53**(3), 379–405 (2006)
58. Song, D., Wagner, D., Perrig, A.: Practical techniques for searching on encrypted data. In: IEEE Symposium on Research in Security and Privacy. pp. 44–55. IEEE Computer Society (2000)
59. Song, X., Dong, C., Yuan, D., Xu, Q., Zhao, M.: Forward private searchable symmetric encryption with optimized i/o efficiency. IEEE Transactions on Dependable and Secure Computing **17**(5), 912–927 (2018)
60. Stefanov, E., Papamanthou, C., Shi, E.: Practical dynamic searchable encryption with small leakage. In: Network and Distributed System Security Symposium (NDSS '14) (2014)

61. Triplett, J., McKenney, P.E., Walpole, J.: Resizable, scalable, concurrent hash tables via relativistic programming. In: 2011 USENIX Annual Technical Conference (USENIX ATC 11) (2011)
62. Valois, J.D.: Lock-free linked lists using compare-and-swap. In: Proceedings of the fourteenth annual ACM symposium on Principles of distributed computing. pp. 214–222 (1995)
63. Zhang, Y., Katz, J., Papamanthou, C.: All your queries are belong to us: The power of file-injection attacks on searchable encryption. In: USENIX Security Symposium (2016)

Lattice-based Cryptography



Verifiable Oblivious Pseudorandom Functions from Lattices: Practical-Ish and Thresholdisable

Martin R. Albrecht^{1,2} and Kamil Doruk Gur^{3(✉)}

¹ King's College London, London, UK
`martin.albrecht@kcl.ac.uk`

² SandboxAQ, Palo Alto, USA
`sandboxaq.com`

³ Department of Computer Science, University of Maryland, College Park, USA
`dgur1@cs.umd.edu`

Abstract. We revisit the lattice-based verifiable oblivious PRF construction from PKC'21 and remove or mitigate its central three sources of inefficiency. First, applying Rényi divergence arguments, we eliminate one superpolynomial factor from the ciphertext modulus q , allowing us to reduce the overall bandwidth consumed by RLWE samples by about a factor of four. This necessitates us introducing intermediate unpredictability notions to argue PRF security of the final output in the Random Oracle model. Second, we remove the reliance on the 1D-SIS assumption, which reduces another superpolynomial factor, albeit to a factor that is still superpolynomial. Third, by applying the state-of-the-art in zero-knowledge proofs for lattice statements, we achieve a reduction in bandwidth of several orders of magnitude for this material. Finally, we give a t -out-of- n threshold variant of the VOPRF for constant t and with trusted setup, based on a n -out-of- n distributed variant of the VOPRF (and without trusted setup).

1 Introduction

An oblivious pseudorandom function (OPRF) is a two-party protocol between a client and a server allowing the client to derive a pseudorandom output based on their input. In particular, an OPRF allows a client to receive a pseudorandom function (PRF) evaluation on an input x from a server with key k . The security of the protocol then refers to (i) the server not learning anything about the input x and (ii) the client not learning anything besides the PRF evaluation of x under k . An OPRF is additionally *verifiable* if the client is guaranteed that the output received is indeed evaluated under a committed key. (V)OPRFs recently gained considerable popularity with important applications including but not limited to secure keyword search [23], private set intersection [31], secure data

K. D. Gur—Work partially done while working for SandboxAQ.

© International Association for Cryptologic Research 2025

K.-M. Chung and Y. Sasaki (Eds.): ASIACRYPT 2024, LNCS 15487, pp. 205–237, 2025.

https://doi.org/10.1007/978-981-96-0894-2_7

de-duplication [34], password-protected secret sharing [27,28], and more popularly password-authenticated key exchange(PAKE) [30] and private lightweight authentication mechanisms [18]. A systematisation of knowledge of OPRFs is given in [14].

Unfortunately, while VOPRFs have useful practical applications and an abundant number of constructions, these are insecure in the post-quantum setting, i.e. in the presence of a quantum adversary, since they rely on classical assumptions. Hence, it is required to design VOPRFs relying on plausibly quantum-resistant assumptions so that the world of functionalities afforded by VOPRFs can also be realised in this new era. Yet, like many other functionalities in a post-quantum setting, post-quantum secure VOPRFs are scarce. We give an overview of the current state of the art in Table 1, extending a table from [3].

In particular, VOPRFs based on lattices have only a limited number of constructions. The first known round-optimal post-quantum VOPRF in [4] is more of a feasibility result rather than a practical proposal due to the required zero-knowledge proofs causing the communication to be in GBs. Even ignoring this cost, bandwidth per query was estimated at about 2MB in the semi-honest set-

Table 1. Post-quantum (V)OPRF candidates in the literature

work	assumption	r	communication	model
plain				
[4]	R(LWE), SIS	2	≈ 2 MB	semi-honest, QROM
[41]	Legendre PRF	3	$\approx \lambda \cdot 13$ K	semi-honest, <i>pp</i> , ROM
[12]	CSIDH	3	424 KB	malicious client
[25]	CSIDH	2	21 KB	semi-honest, <i>ts</i>
[25]	CSIDH	4	35 KB	malicious client, <i>ts</i>
[25]	CSIDH	258	25 KB	semi-honest
[19]	[11]	2	80 B	semi-honest, <i>pp</i>
[22]	AES	?	4746 KB	malicious client, <i>pp</i>
[3]	lattices, [11]	2	2.5 MB + 10 KB	malicious client, ROM
[3]	lattices, [11]	2	15 MB + 5.3 KB	malicious client, ROM
[2]	[2]	2	4.8 B + 114.5 B	semi-honest, <i>pp</i> , wPRF
verifiable				
[4]	R(LWE), SIS	2	> 128 GB	malicious, QROM
[3]	lattices, [11]	2	2.8 MB + 110 KB	malicious, ROM
[3]	lattices, [11]	2	15 MB + 60 KB	malicious, ROM
[9]	Legendre PRF	9	911 KB	malicious, ROM
[8]	[8]	2	28.9 KB	malicious, ROM
this work, $Q = 2^{16}$	R(LWE), SIS	2	108.3kB + 188.6kB	malicious, ROM
this work, $Q = 2^{64}$	R(LWE), SIS	2	221.5kB + 315.9kB	malicious, ROM

The column “r” gives the number of rounds. ROM is the random oracle model, QROM the quantum random oracle model, “pp” stands for “preprocessing”, “ts” for “trusted setup”, “wPRF” for PRFs that accept random inputs. When bandwidth is reported as a sum, this is for a one-time offline cost and online costs per query respectively, where online costs are amortised over 64 queries for [3]. See Table 2 for details on our parameters.

Table 2. Example parameters

$\log Q$	λ	$(d, \log q, \log \sigma')$	$ \mathbf{c} $	$ \mathbf{c}_x $	$ \mathbf{d}_x $	size
4	100	(4096, 137, 35)	(68.5, 36.1)	(68.5, 73.2)	(1.8, 38.8)	(104.6, 182.3)
16	95	(4096, 143, 41)	(71.5, 36.8)	(71.5, 75.5)	(1.8, 39.8)	(108.3, 188.6)
32	90	(4096, 151, 49)	(75.5, 38.0)	(75.5, 79.4)	(1.8, 41.0)	(113.5, 197.7)
64	167	(8192, 169, 67)	(169.0, 52.5)	(169.0, 88.6)	(1.8, 56.5)	(221.5, 315.9)

Q is the number of queries supported, λ the dRLWE security level, d, q are dRLWE dimension and modulus and σ' the size of the drowning noise. All sizes are in kB, (x, y) means size of value and proof except in the the last column which gives the offline and online sizes. We always target a correctness error of 2^{-100} .

ting. More recently [3] adapted the ‘‘Crypto Dark Matter’’ PRF [11] to the lattice setting using fully-homomorphic encryption [15]. The resulting construction achieves practical sizes but is slow to evaluate for the server, and rather complex to implement by relying on the full machinery of fully homomorphic encryption. Moreover, in addition to the PRF assumption in [11], the construction relies on a heuristic argument for verifiability.

For the threshold setting, the concurrent work of [32] is most similar to ours. There, the authors propose four different distributed OPRFs based on the Legendre PRF in a setting where client-server communication is round optimal but servers communicate between client evaluations. Our construction does not have this requirement after the initial setup. Similar to our construction, the construction in [32] is only efficient for small (t, n) pairs. Unlike our constructions, however, [32] requires n servers to be available for evaluation for different settings whereas we only require t for security with aborts. The constructions also have weaker security guarantees compared to ours. Out of the four, only the last constructions promises security against malicious servers with a dishonest majority, which relies on the security guarantees of the underlying MPC operations. This causes issues with some security assumptions, as neither client privacy nor verifiability is provided against $\geq t$ or n corrupted servers respectively. Our construction provides these guarantees also when n servers are corrupted. On other hand, [32] achieves low bandwidth between clients and servers of roughly $n \cdot \lambda^2$ bits, which is much smaller than our construction.

1.1 Technical Overview

To present our contributions, we begin with a high-level overview of the construction from [4] and highlight its main bottlenecks. The VOPRF construction is based on the ring instantiation of the PRF by Banerjee and Peikert [7]

$$F_k(x) = \left\lfloor \frac{p}{q} \cdot \mathbf{a}^F(x) \cdot k \right\rfloor \tag{1}$$

where $k \in \mathcal{R}_q$ is the key with small coefficients represented in $\{-q/2, \dots, q/2\}$ and $\mathbf{a}^F(x)$ is essentially a hash function processing the client input x . Security of the construction can be reduced to the hardness of RLWE. The construction in [4] instantiates this framework with uniformly random public vectors $\mathbf{a}_0, \mathbf{a}_1 \in \mathcal{R}_q^{1 \times \ell}$ and a bit decomposition function G^{-1} . Given a public $\mathbf{a} \in \mathcal{R}_q^{1 \times \ell}$ the high-level protocol is then:

1. The server publishes a commitment $\mathbf{c} := \mathbf{a} \cdot k + \mathbf{e}$ to a small key $k \in \mathcal{R}$.
2. For input x , the client chooses a small $s \in \mathcal{R}$ and $\mathbf{e}_C \in \mathcal{R}^{1 \times \ell}$, and computes $\mathbf{c}_x := \mathbf{a} \cdot s + \mathbf{e}_C + \mathbf{a}^F(x) \bmod q$.
3. Using k , the server sends $\mathbf{d}_x := \mathbf{c}_x \cdot k + \mathbf{e}_S \bmod q$ for small $\mathbf{e}_S \in \mathcal{R}^{1 \times \ell}$.
4. The client finally outputs $\mathbf{y} = \left\lfloor \frac{p}{q} \cdot (\mathbf{d}_x - \mathbf{c} \cdot s) \right\rfloor$.

Since $\mathbf{d}_x = \mathbf{a} \cdot s \cdot k + \mathbf{a}^F(x) \cdot k + \mathbf{e}_C \cdot k + \mathbf{e}_S$, if \mathbf{e}_S is chosen from a distribution that hides the presence of additive terms $\mathbf{e}_C \cdot k$, $\mathbf{e} \cdot s$ and the absence of the additive term \mathbf{e}_x (which follow some narrow distribution $\mathcal{E}_{\mathbf{a}_0, \mathbf{a}_1, x, \sigma}$) then it is indistinguishable from $\mathbf{d}'_x = (\mathbf{a} \cdot k + \mathbf{e}) \cdot s + \mathbf{e}_S + (\mathbf{a}^F(x) \cdot k + \mathbf{e}_x) = \mathbf{c} \cdot s + (\mathbf{a}^F(x) \cdot k + \mathbf{e}_x) + \mathbf{e}_S$. Then if \mathbf{e}_x is chosen from a proper distribution [7], $\mathbf{a}^F(x) \cdot k + \mathbf{e}_x$ and consequently \mathbf{d}_x leaks nothing about k by the RLWE assumption. Similarly, if s chosen from a proper RLWE secret distribution and \mathbf{e} is from a discrete Gaussian, the client message $\mathbf{c}_x = \mathbf{a} \cdot s + \mathbf{e} + \mathbf{a}^F(x)$ is also indistinguishable from uniform by RLWE.

Correctness is satisfied with high probability regardless of the choice of k by the one-dimensional short integer solution (1D-SIS) assumption [13]. Verifiability is then achieved with the help of non-interactive zero-knowledge arguments of knowledge showing \mathbf{c}, \mathbf{c}_x , and \mathbf{d}_x are computed correctly.

The above construction is intuitive in following well-established pre-quantum Diffie-Hellmann blueprints. Moreover, its simple algebraic nature (and instantiation in the standard model, except potentially for zero-knowledge proofs) allows for extensions such as threshold variants.

However, the concrete instantiation is highly inefficient due to three reasons.

First, the correctness of the PRF adds a superpolynomial factor to the modulus q to ensure correct rounding which in the end results in large parameters. Indeed, to thwart adversaries that maliciously sample k such that $\mathbf{a}^F(x) \cdot k$ produces a rounding error for a target value x , [4] relies on the 1D-SIS assumption as just mentioned. This assumption requires $q \gg 2^{2\lambda}$, i.e. more than what we would naively expect to have correct rounding with overwhelming probability.¹

Second, to hide the additive terms $\mathbf{e}_C \cdot k$, $\mathbf{e} \cdot s$ and \mathbf{e}_x , the \mathbf{e}_S has to have superpolynomial size in the norm of these terms. This allows for an argument based on statistical distance to go through.

¹ Concretely, [4] picks $\log(q) \approx 256$ and a ring dimension of 2^{14} for the semi-honest setting, i.e. without considering maliciously chosen k . This leads to a communication cost of 2MB already; relying on the 1D-SIS assumption would require $\log(q) \approx 2048$ based on SIS estimates provided by the lattice estimator [5].

Third, the NIZKAoKs required for verifiability and to protect against malicious clients add further overheads as these relations require non-trivial statements. In particular, the proof that \mathbf{c}_x is correctly computed has to show \mathbf{c}_x indeed contains $\mathbf{a}^F(x)$ without revealing the secrets x , s , or \mathbf{e}_C . Since $\mathbf{a}^F(x)$ is highly irregular with calls to bit decompositions and two different public vectors, [4] used the NIZKAoK construction from [43] which proves *rank-1 constraints* (R1CS) over \mathbb{Z}_q , breaking the native structure of the protocol. Combined with large parameters this causes bandwidth in the GBs.

1.2 Contributions

In this work, we resolve or reduce the above-mentioned sources of inefficiency.

First, we avoid relying on the 1D-SIS assumption, by borrowing a trick from the non-interactive key exchange in [24]. Instead of defining the PRF output as $\lfloor \frac{p}{q} \cdot (\mathbf{a}^F(x) \cdot k) \rfloor$, we define it as $\lfloor \frac{p}{q} \cdot (\mathbf{a}^F(x) \cdot k + \mathbf{r}) \rfloor$ where \mathbf{r} is the output of some Random Oracle called on x and \mathbf{c} : $\mathbf{r} := \mathbf{H}_r(x, \mathbf{c})$. In the Random Oracle model, \mathbf{r} is independent of k and thus $\lfloor \frac{p}{q} \cdot (\mathbf{a}^F(x) \cdot k + \mathbf{r} + \mathbf{e}_C \cdot k + \mathbf{e}_S) \rfloor$ will round to the correct value $\lfloor \frac{p}{q} \cdot (\mathbf{a}^F(x) \cdot k + \mathbf{r}) \rfloor$ with a probability to $\approx 1 - \|\mathbf{e}_C \cdot k + \mathbf{e}_S\|_\infty / (q/p)$. This still requires a superpolynomial gap between q and $\|\mathbf{e}_C \cdot k + \mathbf{e}_S\|_\infty$ but this gap is comparable to that in the semi-honest setting of [4].

Second, we change the way how we analyse \mathbf{e}_S and remove the superpolynomial dependency on the norm of additive terms. To achieve this, we use a Rényi divergence based approach instead of the statistical distance. However, for this, we have to replace the simulation-based security in the standard model in [4] with a game-based notion in the Random Oracle model. In more detail, except in rather particular circumstances, we cannot apply Rényi divergence arguments to decision problems [6]. To work around this, we first show that our construction based on [4] achieves the notion of unpredictability, which we then upgrade to PRF security. Overall, this leads to a bandwidth improvement of roughly an order of magnitude when compared with the semi-honest parameters of [4] (and without NIZKAoK).

Third, we replace the NIZKAoK [43] with that from [38] compressed with LaBRADOR [10] and also work in larger rings \mathcal{R}_q with lattice statements. This improves bandwidth by several orders of magnitude.

Overall, we obtain the sizes reported in Table 2. Compared with [4], our work allows for practical-*ish* parameters. Compared with [3], our bandwidth requirements are smaller if few evaluations are required. In terms of computational burden, we note that [3] has an expensive computation on the server side (TFHE bootstrapping) whereas we have an expensive computation on the client side (proving well-formedness with a complex statement).

Finally, we extend the functionality of the VOPRF and build multiparty protocols. We use n -out-of- n and t -out-of- n *threshold VOPRFs* which consist of n servers jointly evaluating the input x and n (respectively t) servers are required to generate the output. The n -out-of- n construction is immediate from the key-homomorphic properties of the VOPRF. To achieve the more interesting

t -out-of- n setting, we exploit that in the VOPRF setting, we expect t to be quite small, i.e. constant. Moreover, we assume a trusted setup. While this is a significant limitation of this work, we think this assumption is justified in the VOPRF setting, where one entity may aim to avoid single points of failure, rather than multiple parties coming together to, say, validate some statement, i.e. the threshold signature setting. In our approach, we essentially output $\binom{n}{t}$ copies of the n -out-of- n setting. We use rejection sampling to enforce that these are all well-distributed. To achieve verifiability in the t -out-of- n case we rely on an additional cut-and-choose type argument to be able to use weaker NIZKAoKs.

2 Preliminaries

For integers a and b where $a < b$ we use $[a, b]$ to represent the set $\{a, a + 1, \dots, b - 1, b\}$. If $a = 0$ and $b = n - 1$ we use the notation $[n]$ instead. For a vector b we use $b[i]$ as the indexing operator. We denote the output of probabilistic algorithms with \leftarrow and deterministic ones with $:=$. Similarly for a distribution \mathcal{D} or a bounded set S if an element x is sampled according to distribution \mathcal{D} or uniformly random from S we denote it as $x \leftarrow \mathcal{D}$ and $x \leftarrow S$ respectively. For two distributions, we use \approx_c to denote they are computationally indistinguishable. A PPT algorithm is a probabilistic algorithm with running time polynomial in the security parameter λ . We say a function is negligible in λ if $\lambda^{-\omega(1)}$ and write $r_1 \gg r_2$ as short-hand for $r_1 \geq \lambda^{\omega(1)} \cdot r_2$. We denote the ℓ_x norm of a vector with $\|\cdot\|_x$. If $x = 2$ and clear from the context, we omit the subscript. A distribution \mathcal{D} is B bounded if $\Pr[\|\mathbf{x}\| \geq B : \mathbf{x} \leftarrow \mathcal{D}] < \delta$ for a negligible δ . We also consider the rounding operation $\lfloor \cdot \rfloor$ to the nearest integer (rounding down if there is a tie) and $\lfloor x \rfloor_{q'} := \lfloor q'/q \cdot x \rfloor$ from \mathbb{Z}_q to $\mathbb{Z}_{q'}$ for $q' < q$ and $x \in \mathbb{Z}_q$. We use lowercase letters to denote ring elements and boldface lowercase letters to denote vectors.

We use power of two cyclotomic rings in this work. For a modulus $q \in \mathbb{Z}$, we consider the polynomial ring $\mathcal{R} = \mathbb{Z}[X]/\langle X^N + 1 \rangle$ and $\mathcal{R}_q := \mathcal{R}/q\mathcal{R}$ for a power-of-two N . The set $\mathcal{R}_{\leq c}$ is then the set of all elements of \mathcal{R} with coefficients that have an absolute value of at most c . Norms of ring elements are defined over the coefficient vectors of the said elements and norms of vectors of ring elements are norms of the concatenation of the coefficient vectors.

Define $\mathbf{G} : \mathcal{R}_q^{\ell \times \ell} \rightarrow \mathcal{R}_q^{1 \times \ell}$ to be the linear operation corresponding to left multiplication by $(1, 2, \dots, 2^{\ell-1})$. Further, define $G^{-1} : \mathcal{R}_q^{1 \times \ell} \rightarrow \mathcal{R}_q^{\ell \times \ell}$ to be the bit decomposition operation that essentially inverts \mathbf{G} i.e. the i^{th} column of $G^{-1}(\mathbf{a})$ is the bit decomposition of $a_i \in \mathcal{R}_q$ into binary polynomials.

For $\mathbf{a}_0, \mathbf{a}_1 \in \mathcal{R}_q^{1 \times \ell}$, $x \in \{0, 1\}^L$, and $i \in [L]$ define

$$\mathbf{a}_{x \setminus i} := G^{-1}(\mathbf{a}_{x_{i+1}} \cdot G^{-1}(\mathbf{a}_{x_{i+2}} \cdot G^{-1}(\dots(\mathbf{a}_{x_{L-2}} \cdot G^{-1}(\mathbf{a}_{x_{L-1}})) \dots))) \in \mathcal{R}_q^{\ell \times \ell}.$$

Now, for a client input $x \in \{0, 1\}^L$, let $\mathcal{E}_{\mathbf{a}_0, \mathbf{a}_1, x, \sigma}$ be the distribution of all \mathbf{e}_x computed as $\mathbf{e} = \sum_{i=0}^{L-2} \mathbf{e}_i \cdot \mathbf{a}_{x \setminus i} + \mathbf{e}_{L-1}$ where $\forall i \in [L]: \mathbf{e}_i \leftarrow \mathcal{R}_{\chi_\sigma}^{1 \times \ell}$.

2.1 Discrete Gaussian Distributions over Polynomial Rings

The discrete Gaussian distribution over \mathcal{R} is defined as follows:

Definition 1. For $\mathbf{x} \in \mathcal{R}^m$ let $\rho_{\mathbf{v},s}(\mathbf{x}) = \exp(-\pi\|\mathbf{x} - \mathbf{v}\|_2^2/s^2)$ be the Gaussian function of parameters $\mathbf{v} \in \mathcal{R}^m$ and $s \in \mathbb{R}$. Then the discrete Gaussian distribution $\mathcal{D}_{\mathbf{v},s}^m$, centered at \mathbf{v} is

$$\mathcal{D}_{\mathbf{v},s}^m = \rho_{\mathbf{v},s}(x)/\rho_{\mathbf{v},s}(\mathcal{R}^m) \text{ where } \rho_{\mathbf{v},s}(\mathcal{R}^m) = \sum_{\mathbf{x} \in \mathcal{R}^m} \rho_{\mathbf{v},s}(x).$$

When there is only a single element in a vector, we omit the superscript and if \mathbf{v} is the zero-vector, we omit the subscript \mathbf{v} . When s exceeds the smoothing parameter $\eta_\epsilon(\mathcal{R}^m) \leq \omega(\sqrt{\log(mN)})$, \mathcal{D}_s^m behaves like a continuous Gaussian of standard deviation of $\sigma = s/2\pi$. The following lemmas will be useful when we discuss our key generation algorithm for our threshold construction where we need to argue about the distribution of key shares based on properties of Gaussians.

Lemma 1 [40, Theorem 3.3]. Let s be a parameter exceeding the smoothing parameter by a factor of at least $\sqrt{2}$ and \mathbf{x}_i for $i \in [n]$ be independent samples from \mathcal{D}_s^m . Then the distribution of $\mathbf{x} := \sum_i \mathbf{x}_i$ is statistically close to $\mathcal{D}_{s\sqrt{n}}^m$.

For our threshold construction, we rely on rejection sampling [37] to guarantee each partial key adheres to a particular distribution. The following lemma shows for a vector of ring elements \mathbf{x} sampled from a Gaussian, the ℓ_2 norm is bounded for all but negligible probability:

Lemma 2 [37, Lemma 4.4 adapted]. For any $\gamma > 1$, we have

$$\Pr \left[\|\mathbf{x}\|_2 > \gamma \cdot \sigma \cdot \sqrt{m \cdot N} : \mathbf{x} \leftarrow \mathcal{D}_s^m \right] < \gamma^{mN} \cdot e^{mN \cdot (1-\gamma^2)/2}.$$

To be able to argue that our key shares in the t -out-of- n setting are no different compared with a key sampled from an independent Gaussian distribution, we have the following lemma that allows us to decide on a σ and the expected number of repetitions M in rejection sampling:

Lemma 3 [37, Lemma 4.5 adapted]. For a $V \subseteq \mathcal{R}^m$, let $T = \max_{\mathbf{v} \in V} \|\mathbf{v}\|_2$. For a fixed t with $t = \omega(\sqrt{\log(mN)})$ and $t = o(\log(mN))$ if $\sigma = \alpha \cdot T$ for any positive α then:

$$\Pr \left[M \geq \mathcal{D}_s^m(\mathbf{x})/\mathcal{D}_{\mathbf{v},s}^m(\mathbf{x}) : \mathbf{x} \leftarrow \mathcal{D}_s^m \right] \geq 1 - \epsilon$$

where $M = e^{t/\alpha + 1/(2(\alpha^2))}$ and $\epsilon = 2e^{-t^2/2}$.

For practical set of parameters, M grows slowly which is useful for arguing that rejection sampling does not need too many trials to “clean up” a small centre \mathbf{v} . For the remainder of the work we use $\mathcal{R}_{\chi_\sigma}$ to denote distribution of elements in \mathcal{R}_q which have coefficients distributed according to the a discrete Gaussian with parameter χ_σ .

2.2 Rényi Divergence

For any two discrete probability distributions ϕ and ϕ' such that $\text{Supp}(\phi) \subseteq \text{Supp}(\phi')$ and an $\alpha \in (1, +\infty)$. The Rényi divergence of order α can be defined as:

$$R_\alpha(\phi \parallel \phi') := \left(\sum_{x \in \text{Supp}(\phi')} \frac{\phi(x)^\alpha}{\phi'(x)^{\alpha-1}} \right)^{\frac{1}{\alpha-1}}.$$

The Rényi divergence R_α has the following properties for probability distributions ϕ, ϕ', ϕ'' with $\text{Supp}(\phi) \subseteq \text{Supp}(\phi') \subseteq \text{Supp}(\phi'')$:

- **Log. Positivity:** $R_\alpha(\phi \parallel \phi') \geq R_\alpha(\phi \parallel \phi) = 1$.
- **Data Processing Inequality:** $R_\alpha(\phi^f \parallel \phi'^f) \leq R_\alpha(\phi \parallel \phi')$ for any function f where ϕ^f denotes the distribution of $f(y)$ where $y \leftarrow \phi$.
- **Multiplicativity:** Assume ϕ and ϕ' are two distributions for a pair of random variables (Y_0, Y_1) . For $i \in \{0, 1\}$, let ϕ_i denote the marginal distribution of Y_i under ϕ , and let $\phi_{1|0}(\cdot|y_0)$ denote the conditional distribution of Y_1 given $Y_0 = y_0$. Then
 - $R_\alpha(\phi \parallel \phi') = R_\alpha(\phi_0 \parallel \phi'_0) \cdot R_\alpha(\phi_1 \parallel \phi'_1)$ if Y_0 and Y_1 are independent for α in given interval.
 - $R_\alpha(\phi \parallel \phi') \leq R_\infty(\phi_0 \parallel \phi'_0) \cdot \max_{y_0 \in X} R_\alpha(\phi_{1|0}(\cdot|y_0) \parallel \phi'_{1|0}(\cdot|y_0))$
- **Probability Preservation:** Let $E \subseteq \text{Supp}(\phi')$ be an arbitrary event. For given interval of α , $\phi'(E) \geq \phi(E)^{\frac{\alpha}{\alpha-1}} / R_\alpha(\phi \parallel \phi')$. Furthermore

$$\phi'(E) \geq \phi(E) / R_\infty(\phi \parallel \phi').$$

Additionally, we rely on the following lemma to argue about the Rényi divergence between Gaussians with different centers.

Lemma 4 [35]. *Let P and Q be distributions corresponding to Gaussians $\mathcal{D}_{\mathbf{c},s}^m$ and $\mathcal{D}_{\mathbf{c}',s}^m$ with centers \mathbf{c} and \mathbf{c}' , and $s \geq \eta(\mathcal{R}^m)$. Then for any $\alpha \in (1, +\infty)$:*

$$R_\alpha(P \parallel Q) \leq \exp \left(\alpha \cdot \pi \cdot \frac{\|\mathbf{c} - \mathbf{c}'\|_2^2}{s^2} \right)$$

Remark 1. Note that the Rényi divergence grows exponentially with m , since $\|\mathbf{c} - \mathbf{c}'\|_2^2$ grows linearly with m . Similarly, the Rényi divergence for z samples grows exponentially in z by the multiplicative property.

2.3 Hardness Assumptions

We rely on the standard decisional RLWE problem for the security of the VOPRF function.

Definition 2 ([39, 42]). Let $\mathcal{R}_q, m, \sigma_s, \sigma_e > 0$ depend on security parameter λ for integers q, N and m and $\mathcal{R}_q := \mathbb{Z}_q[x]/(X^N + 1)$. The decision Ring Learning with Errors (dRLWE $_{\mathcal{R}_q, m, \sigma_s, \sigma_e}$ for short) problem is to distinguish between

$$(a_i, a_i \cdot s + e_i)_{i \in [m]} \in (\mathcal{R}_q)^2 \text{ and } (a_i, u_i)_{i \in [m]} \in (\mathcal{R}_q)^2$$

for $a_i, u_i \leftarrow \mathcal{R}_q$; $s, \leftarrow \mathcal{R}_{\chi_{\sigma_s}}$ $e_i \leftarrow \mathcal{R}_{\chi_{\sigma_e}}$.

When the number of samples m is implicit, we omit the subscript.

Remark 2. We note the trivial hierarchy that dRLWE $_{\mathcal{R}_q, m, \sigma_0, \sigma}$ is at least as hard as dRLWE $_{\mathcal{R}_q, m, \sigma_1, \sigma}$ when $\sigma_0 = \sqrt{k} \cdot \sigma_1$ for any integer $k > 1$ and $\sigma_1 \geq \sqrt{2} \cdot \eta_\epsilon(\mathcal{R})$. Given samples from the latter (a_i, b_i) submit $(a_i, b_i + a_i \cdot \delta)$ to the distinguisher for the former, where $\delta \leftarrow \mathcal{R}_{\chi_{\sqrt{k-1}\sigma_0}}$. By a simple corollary of Lemma 1, we have that $\delta + s$ is correctly distributed if $s \leftarrow \mathcal{R}_{\chi_{\sigma_0}}$. Since $\sum_{i=0}^{k-1} \chi_{\sigma_0} \approx_s \chi_{\sigma_1}$ and $\sum_{i=0}^{k-2} \chi_{\sigma_0} \approx_s \chi_{\sqrt{k-1}\sigma_0}$, we have $\chi_0 + \chi_{\sqrt{k-1}\sigma_0} \approx_s \chi_{\sigma_1}$. Here, “ \approx_s ” indicates that two distributions are statistically close.

2.4 Non-interactive Zero-Knowledge Arguments of Knowledge (NIZKAoK)

We use the standard definitions regarding zero-knowledge (ZK) proof systems and arguments of knowledge (AoK). Informally, a ZK proof system for a language \mathcal{L} allows a prover \mathbb{P} to convince a verifier \mathbb{V} some x is in \mathcal{L} and not reveal anything else. A ZKAoK then provides a stronger guarantee where \mathbb{P} also convinces \mathbb{V} that they hold a witness w attesting to the fact. Formally the definition is as follows:

Definition 3. For a prover \mathbb{P} , a verifier \mathbb{V} , a language \mathcal{L} with accompanying predicate $P_{\mathcal{L}}(\cdot, \cdot)$, a witness set $\mathcal{W}_{\mathcal{L}}(\cdot)$ such that for all $x \in \mathcal{L}$ and $w \in \mathcal{W}_{\mathcal{L}}(x)$ $P_{\mathcal{L}}(x, w) = 1$, a NIZKAoK is a tuple of algorithms $(\text{Setup}, \mathbb{P}, \mathbb{V})$ such that:

- $\text{Setup}(1^\lambda)$: On input 1^λ outputs a common reference string crs .
- $\mathbb{P}(\text{crs}, x, w)$: On input of a common reference string crs , a statement $x \in \mathcal{L}$, and a witness $w \in \mathcal{W}_{\mathcal{L}}(x)$ outputs a proof $\pi \in \{0, 1\}^*$ polynomial in λ .
- $\mathbb{V}(\text{crs}, x, \pi)$: On input of a common reference string crs , a statement x , and a proof $\pi \in \{0, 1\}^*$ outputs $b \in \{0, 1\}$.

The security of a NIZKAoK holds as long as the following definitions hold

Definition 4 (Completeness). For $x \in \mathcal{L}$, $w \in \mathcal{W}_{\mathcal{L}}(x)$ with $P_{\mathcal{L}}(x, w) = 1$:

$$\Pr \left[1 \leftarrow \mathbb{V}(\text{crs}, x, \pi) : \begin{array}{l} \text{crs} \leftarrow \text{Setup}(1^\lambda) \\ \pi \leftarrow \mathbb{P}(\text{crs}, x, w) \end{array} \right] \geq 1 - \epsilon$$

where ϵ is negligible in λ .

Definition 5 (Computational Knowledge Extraction (Extractability)).

The NIZKAoK is said to have computational knowledge extraction (or is extractable) with knowledge error ϵ_{extr} if for any malicious prover \mathbb{P}^* with auxiliary information aux there exists an extraction algorithm Extract and a polynomial p such that for any x :

$$\Pr [1 \leftarrow P_{\mathcal{L}}(x, w') : w' \leftarrow \text{Extract}(\mathbb{P}^*(\text{crs}, x, \text{aux}))] \geq \frac{\epsilon_{\text{Vfy}} - \epsilon_{\text{extr}}}{p(|x|)}$$

where ϵ_{Vfy} is the probability that $\mathbb{V}(\text{crs}, x, \mathbb{P}^*(\text{crs}, x, \text{aux}))$ outputs 1.

Definition 6 (Computational zero-knowledge). There exists a simulated setup algorithm SimSetup which on input 1^λ outputs crs_{Sim} and a trapdoor \mathcal{T} along with a PPT simulator Sim where for all $x \in \mathcal{L}$ and $w \in \mathcal{W}_{\mathcal{L}}(x)$:

$$\left\{ \begin{array}{l} \text{crs} \leftarrow \text{Setup}(1^\lambda) \\ \pi \leftarrow \mathbb{P}(\text{crs}, x, w) \end{array} \right\} \approx_c \left\{ \begin{array}{l} \text{crs}' \\ \pi_{\text{Sim}} \leftarrow \text{Sim}(\text{crs}', \mathcal{T}, x) : (\text{crs}', \mathcal{T}) \leftarrow \text{SimSetup}(1^\lambda) \end{array} \right\}$$

2.5 Verifiable Oblivious Pseudorandom Functions

A verifiable oblivious pseudorandom function (VOPRF) for a keyed function F is a two-party protocol between a client \mathbb{C} and a server \mathbb{S} consisting of the following algorithms:

- $\text{Init}_{\mathbb{S}}$ is a protocol run by \mathbb{S} which on input 1^λ outputs a secret key sk and its public commitment pk .
- $\text{Init}_{\mathbb{C}}$ is a protocol run by \mathbb{C} which on input pk outputs a *state* indicating acceptance/rejection of public commitment.
- $\text{Query}_{\mathbb{C}}$ is a protocol run by \mathbb{C} which on input client input x and *state* outputs a blinded message \bar{x} and a state ρ .
- $\text{Query}_{\mathbb{S}}$ is a protocol run by \mathbb{S} which on input of client’s blinded message \bar{x} and a secret key sk outputs a blinded evaluation y_x .
- Finalize is a protocol run by \mathbb{C} which on input server’s blinded evaluation y_x , public commitment pk and a state ρ outputs the PRF output y .

We define security of VOPRF based on corresponding games. This is not common for OPRF protocols and only a handful instantiations in literature use game-based notion of security. A protocol $\text{PRF} = (\text{Init}_{\mathbb{S}}, \text{Init}_{\mathbb{C}}, \text{Query}_{\mathbb{C}}, \text{Query}_{\mathbb{S}}, \text{Finalize})$ with inputs $x \in \{0, 1\}^*$ and $sk \in \mathcal{K}$ is a VOPRF protocol corresponding to a keyed function F if the following hold:

Definition 7 (Correctness). For every pair of inputs x, sk :

$$\Pr [\text{PRF}(x, sk) \neq F_{sk}(x)] \leq \epsilon$$

where ϵ is negligible in security parameter λ .

Definition 8 (Obliviousness [36]). PRF is said to be oblivious if for any PPT adversary \mathcal{A} the probability of the obliviousness experiment depicted in Fig. 1a outputting 1 is $1/2 + \epsilon$ where ϵ negligible in λ .

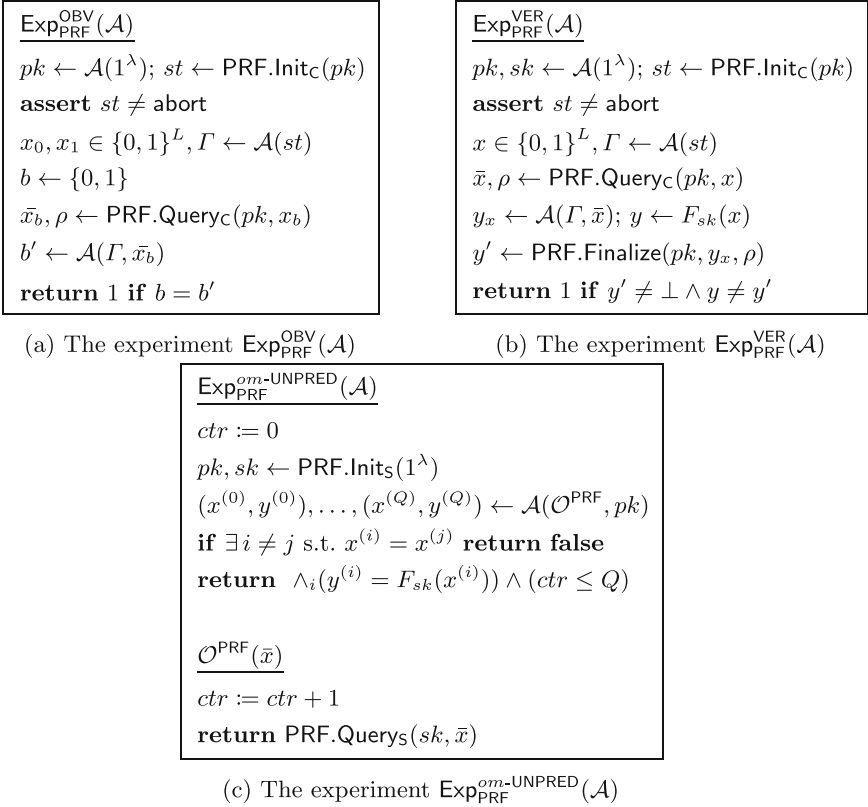


Fig. 1. Obliviousness, Verifiability and Unpredictability

In the obliviousness game of Definition 8 security is still based on an indistinguishability notion. For malicious \mathbb{C} , since we will rely on R enyi based argument, we cannot use such an indistinguishability-based notion for security unless a specific set of conditions are met [6]. We instead define a search-based security game and upgrade this to indistinguishability of the output in the Random Oracle model (ROM).

Definition 9 (One-more unpredictability [21]). *PRF is said to be one-more unpredictable if for any PPT adversary \mathcal{A} the probability of the one-more unpredictability experiment depicted in Fig. 1c outputting 1 is negligible in λ .*

Note that the queries to the oracle \mathcal{O}^{PRF} include receiving blinded inputs from \mathcal{A} as it is not guaranteed that \mathcal{A} outputs a correctly computed \bar{x} . The definition is similar to unforgeability definitions for signature schemes. The intuition here is once we can argue that the interaction between \mathbb{C} and \mathbb{S} has an unpredictable output, the security of the VOPRF can be shown in the ROM. To achieve this we use a different notion of security called *one-more PRF security*.

$\text{Exp}_{\text{PRF,H}}^{\text{om-PRF}}(\mathcal{A})$	$\mathcal{O}^{\text{RoR}}(x_i)$
$ctr, q := 0, 0$	$q := q + 1, b[q] \leftarrow \{0, 1\}$
$pk, sk \leftarrow \text{PRF.Init}_S(1^\lambda)$	$y_0 \leftarrow \{0, 1\}^*$
$(i_1, \dots, i_Q, b') \leftarrow \mathcal{A}(\text{H}, \mathcal{O}^{\text{PRF}}, \mathcal{O}^{\text{RoR}})$	$y_1 \leftarrow \text{H}(x_i, F_{sk}(x_i))$
if $\exists \alpha$ s.t. $i_\alpha \notin [Q]$ return false	return $y_{b[q]}$
if $Q > q$ or $ctr \geq Q$ return false	
if $\exists \alpha \neq \beta$ s.t. $i_\alpha = i_\beta$ return false	$\mathcal{O}^{\text{PRF}}(\bar{x})$
return $b' := \bigoplus_{\alpha=1}^Q b[i_\alpha]$	$ctr := ctr + 1$
	return $\text{PRF.Query}_S(sk, \bar{x})$

Fig. 2. The experiment $\text{Exp}_{\text{PRF,H}}^{\text{om-PRF}}(\mathcal{A})$

Definition 10 (One-more PRF security [21]). A PRF is said to be one-more pseudorandom if for any PPT adversary \mathcal{A} the probability of the one-more pseudorandomness depicted in Fig. 2 outputting 1 is negligible in λ .

It is easy to see that unpredictability implies one-more PRF security in the ROM when Finalize includes y_x as an input to an oracle H.

Lemma 5. Let F be a keyed one-more unpredictable function and H be a hash function modeled as a random oracle. Then the function PRF corresponding to F has one-more PRF security.

Proof. We can construct an adversary \mathcal{A} against unpredictability using an adversary \mathcal{B} against one-more PRF security as a subroutine. Let \mathcal{B} be an adversary who outputs a tuple (i_1, \dots, i_Q) and a bit b' . If \mathcal{B} wins the one-more PRF security game then \mathcal{B} distinguished Q real or random PRF executions with $ctr < Q$ queries to the PRF oracle and $q \geq Q$ queries to the real or random oracle for the specific input. Even if \mathcal{B} submits all of its PRF oracle queries to the real or random oracle, since $q > ctr$ this means there exists at least one oracle answer $x_{i'}, y_{i'}$ that has not been queried to the PRF oracle. The adversary \mathcal{A} then can use $x_{i'}$ to find $F_{sk}(x_{i'})$ such that $y_{i'} = \text{H}(x_{i'}, F_{sk}(x_{i'}))$. The adversary \mathcal{A} then submits $x_{i'}, F_{sk}(x_{i'})$ along with queried x_i as the answer to the unpredictability game. Since the queried x_i have been generated by the PRF oracle, the answers are valid. On the other hand, $x_{i'}$ was never queried to the PRF oracle yet is a valid answer which means \mathcal{A} will win the unpredictability game.

It also has been shown in prior work that one-more PRF security is the strict strengthening of the traditional PRF security [21].

Finally, we define the notion of *verifiability* which assures \mathbb{C} the output y is indeed $F_{sk}(x)$.

Definition 11 (Verifiability [3]). PRF is said to be verifiable if for any PPT adversary \mathcal{A} the probability of the verifiability experiment depicted in Fig. 1b outputting 1 is negligible in λ .

2.6 Lattice (VO)PRFs

We use the construction from [4] which adapts the lattice PRF $F_k(x) = \lfloor \frac{p}{q} \cdot \mathbf{a}^F(x) \cdot k \rfloor$ from [7] into ring setting. The construction can be thought of an instantiation of Fig. 3 where $n = 1$, $H_r(\cdot, \cdot) := 0$ and $H(\cdot, y_x) := y_x$.²

Remark 3. We note that there is a generic transformation upgrading any VOPRF to a *partial* VOPRF, where part of the client's input is in the clear. The server computes $\mathbf{sk}_t := H_K(\mathbf{sk}, t)$ for its master secret key \mathbf{sk} and public client input or tag t and then proceeds with \mathbf{sk}_t [14, 29]. In the verifiable case, the server is also required to output a commitment to \mathbf{sk}_t , to have something to verify against. We forego discussing partial variants of (V)OPRFs for the remainder of this work.

3 Construction

We first define the languages $\mathcal{L}_0, \mathcal{L}_1, \mathcal{L}_2$ with corresponding predicates $P_{\mathcal{L},0}, P_{\mathcal{L},1}, P_{\mathcal{L},2}$ for our NIZKAoKs:

$$\mathcal{L}_0 := \left\{ P_{\mathcal{L},0}(x, w) = 1 \left| \begin{array}{l} x := (c_i) \wedge w := (k_i, e_i): \\ \|k_i\|_2, \|e_i\|_2 \leq B_0 \wedge c_i = a \cdot k_i + e_i \pmod q \end{array} \right. \right\}$$

$$\mathcal{L}_1 := \left\{ P_{\mathcal{L},1}(x, w) = 1 \left| \begin{array}{l} x := (c_x) \wedge w := (x = (x_0, \dots, x_{L-1}), s, e_{\mathbb{C}}): \\ \|s\|_2, \|e_{\mathbb{C}}\| \leq B_1 \\ \wedge \mathbf{a}_x = \mathbf{a}_{x_0} \cdot G^{-1}(\dots(\mathbf{a}_{x_{L-2}} \cdot G^{-1}(\mathbf{a}_{x_{L-1}})) \dots) \\ \wedge c_x = a \cdot s + e_{\mathbb{C}} + \mathbf{a}_x[0] \pmod q \end{array} \right. \right\}$$

$$\mathcal{L}_2 := \left\{ P_{\mathcal{L},2}(x, w) = 1 \left| \begin{array}{l} x := (c_i, c_x, d_{x,i}) \wedge w := (k_i, e_i, e_{\mathbb{S},i}): \\ \|k_i\|_2, \|e_i\|_2 \leq B_0 \wedge \|e_{\mathbb{S},i}\|_2 \leq B_2 \\ \wedge c_i = a \cdot k_i + e_i \pmod q \\ \wedge d_{x,i} = c_x \cdot k_i + e_{\mathbb{S},i} \pmod q \end{array} \right. \right\}$$

for reference strings $\text{crs}_0 = (a, B_0)$, $\text{crs}_1 = (a, \mathbf{a}_0, \mathbf{a}_1, B_1)$, and $\text{crs}_2 = (a, B_2)$. Our construction, which is a mild variant of the construction given in [4], is given in Fig. 3 when $n = 1$. For the rest of this work, we set $B_0 = \sigma \cdot \sqrt{N}$, $B_1 = \sigma \cdot \sqrt{N}$, and $B_2 = \sigma' \cdot \sqrt{N}$.

We start by proving that the protocol is secure against a malicious \mathbb{S} i.e. is oblivious and verifiable. The first proof is almost exactly the same as the malicious server proof of [4] and given for completeness; the second proof is similar but drops the need for invoking the 1D-SIS assumption.

² In [4] the protocol is defined for vectors of ring elements of length ℓ rather than single ring elements. We give the variant here, already mentioned in [4], that only considers a single ring element, for performance.

CRS SetUp:

- $\mathbf{a}_0, \mathbf{a}_1 \leftarrow R_q^{1 \times \ell}$.
- $a \leftarrow R_q$, sample crs_0 for \mathbb{P}_0 , $\text{crs}_0 := (\text{crs}_0, a)$.
- Sample $\text{crs}_1, \text{crs}_2$ for $\mathbb{P}_1, \mathbb{P}_2$.

Initialisation:

- Init_S : Server \mathbb{S}_i for $i \in [n]$ executes
 - Choose σ_i such that $\sigma_i \leq \sigma$
 - $k_i \leftarrow R_{\chi_{\sigma_i}}, e_i \leftarrow R_{\chi_\sigma}$.
 - $c_i \leftarrow a \cdot k_i + e_i \pmod q$.
 - $\pi_{0,i} \leftarrow \mathbb{P}_0(k_i, e_i : \text{crs}_0, c_i)$
 and broadcasts $c_i, \pi_{0,i}$.
- Init_C : \mathbb{C} on input of $\{(c_i, \pi_{0,i})\}_{i \in [n]}$ executes
 - $b_i \leftarrow \mathbb{V}_0(\text{crs}_0, c_i, \pi_{0,i})$.
 - Output **abort** with i if $b_i = 0$, otherwise store c_i .

Query:

1. Query_C : \mathbb{C} executes the following with the input $(x \in \{0, 1\}^L, \text{crs}_1, \text{crs}_2)$
 - $s \leftarrow R_{\chi_\sigma}, e_C \leftarrow R_{\chi_\sigma}$.
 - $\mathbf{a}_x := \mathbf{a}_{x_0} \cdot G^{-1}(\dots(\mathbf{a}_{x_{L-2}} \cdot G^{-1}(\mathbf{a}_{x_{L-1}}))\dots) \pmod q$.
 - $c_x \leftarrow a \cdot s + e_C + \mathbf{a}_x[0] \pmod q$.
 - $\pi_1 \leftarrow \mathbb{P}_1(x, s, e_C : \text{crs}_1, c_x, a, \mathbf{a}_0, \mathbf{a}_1)$.
 and broadcasts (c_x, π_1) to all \mathbb{S}_i .
2. Query_S : \mathbb{S}_i executes the following after receiving (c_x, π_1)
 - $b \leftarrow \mathbb{V}_1(\text{crs}_1, c_x, \mathbf{a}_0, \mathbf{a}_1, \pi_1)$, output **abort** if $b = 0$.
 - $e_{S,i} \leftarrow R_{\chi_{\sigma_i}}$.
 - $d_{x,i} := c_x \cdot k_i + e_{S,i} \pmod q$.
 - $\pi_{2,i} \leftarrow \mathbb{P}_2(k_i, e_{S,i}, e_i : \text{crs}_2, c_i, d_{x,i}, c_x, a)$.
 and sends $(d_{x,i}, \pi_{2,i})$ to \mathbb{C} and outputs \perp .
3. Finalize_C : \mathbb{C} finally executes the following after receiving $(d_{x,i}, \pi_{2,i})$ from all \mathbb{S}_i .
 - $b_i \leftarrow \mathbb{V}_2(\text{crs}_0, \text{crs}_2, c_i, d_{x,i}, c_x, \pi_{2,i})$, output **abort** with i if $b_i = 0$.
 - $d_x := \sum_{i \in [n]} d_{x,i}, c := \sum_{i \in [n]} c_i$.
 - $r \leftarrow H_r(x, c) \in \mathcal{R}_q // H_r(\cdot, \cdot) := 0$ in [4]
 - $y_x := \lfloor d_x + r - c \cdot s \rfloor_p$.
 - $y \leftarrow H(x, y_x) // H(\cdot, y_x) := y_x$ in [4]
 and outputs y .

Fig. 3. n -out-of- n VOPRF Construction.

Theorem 1. *Let σ and N be poly(λ). Let $\text{dRLWE}_{q,N,\sigma,\sigma}$ be hard. Let $(\mathbb{P}_0, \mathbb{V}_0)$, $(\mathbb{P}_1, \mathbb{V}_1)$ be NIZKAoKs for languages $\mathcal{L}_0, \mathcal{L}_1$, then the protocol in Fig. 3 with $n = 1$ is oblivious against any PPT adversary \mathcal{A} controlling \mathbb{S} .*

We give the proof in the full version of this work.

Theorem 2. *Let σ and N be $\text{poly}(\lambda)$. Let $\beta = 2\sigma^2 \cdot N + \sigma' \cdot \sqrt{N}$ and $q/p \gg \beta$. Let $(\mathbb{P}_0, \mathbb{V}_0), (\mathbb{P}_2, \mathbb{V}_2)$ be NIZKAoKs for languages $\mathcal{L}_0, \mathcal{L}_2$, H_r be a random oracle, and Q_H be number of queries made to such oracle. Then the protocol in Fig. 3 with $n = 1$ is verifiable against any PPT adversary \mathcal{A} controlling \mathbb{S} in the ROM.*

Proof. We show \mathcal{A} corrupting a server \mathbb{S}^* cannot have a significant advantage by biasing the output derived by \mathbb{C} and force an incorrect evaluation.

If \mathbb{S}^* 's reply does not have a valid proof for the setup or the final round then the client will abort. If not, we extract a key k^* from π_0 with $\|k^*\|_\infty \leq \sigma \cdot \sqrt{N}$. Let s and e_C be sampled as it is in the protocol for an honest client therefore $\|s\|_\infty \leq \|s\|_2 \leq \sigma \cdot \sqrt{N}$ and $\|e_C\|_\infty \leq \|e_C\|_2 \leq \sigma \cdot \sqrt{N}$. Observe that an honest client has

$$\frac{p}{q} \cdot \left(d_x + r - c \cdot s \right) = \frac{p}{q} \cdot \mathbf{a}_x[0] \cdot k^* + \frac{p}{q} \cdot r + \frac{p}{q} \cdot \left(e_C \cdot k^* - e \cdot s + e_S \right)$$

Each k^*, e_S are correctly computed according to the protocol hence $\|k^*\|_\infty \leq \|k^*\|_2 \leq \sigma \cdot \sqrt{N}$ and $\|e_S\|_\infty \leq \|e_S\|_2 \leq \sigma' \cdot \sqrt{N}$.

If every coefficient of $\frac{p}{q} \cdot \mathbf{a}_x[0] \cdot k^* + \frac{p}{q} \cdot r$ is further away from $\mathbb{Z} + 1/2$ than $\left\| \frac{p}{q} \cdot (e_C \cdot k^* - e \cdot s + e_S) \right\|_\infty$, the evaluation is correct. The adversary can query H_r to find r^* such that evaluation would be incorrect. Note that in the Random Oracle Model r^* is independent of k^* , since H_r takes a commitment to k^* as one of its inputs, therefore the only way \mathcal{A} can find a satisfying r^* is by finding an x^* such that $r^* := H_r(x^*, c)$. This probability is negligible in the Random Oracle Model as long as $2^\lambda \gg Q_H$. It follows that \mathcal{A} can only force an incorrect evaluation with probability proportional to $\|e_C \cdot k^* - e \cdot s + e_S\|_\infty / (q/p)$. We then have

$$\begin{aligned} \|e_C \cdot k^* - e \cdot s + e_S\|_\infty / (q/p) &\leq (\|e_C\|_\infty \cdot \|k^*\|_\infty + \|e\|_\infty \cdot \|s\|_\infty + \|e_S\|_\infty) / (q/p) \\ &\leq (2\sigma^2 \cdot N + \sigma' \sqrt{N}) / (q/p). \end{aligned}$$

Since $q/p \gg \beta$ with $\beta = 2\sigma^2 \cdot N + \sigma' \cdot \sqrt{N}$ the probability above is negligible. \square

4 Using Rènyi Divergence for Smaller Parameters

The security of [4] relies on \mathbf{e}_S chosen from a distribution $\mathcal{R}_{\chi_{\sigma'}}^{1 \times \ell}$ with $\sigma' \gg \max(L \cdot \ell \cdot \sigma \cdot N^{3/2}, \sigma^2 \cdot N^2)$. This superpolynomial gap has a significant impact on communication costs. In the malicious client proof of [4], the security argument relies on the statistical distance between

$$\mathbf{c} \cdot s + \mathbf{e}_C \cdot k + \mathbf{a}_x \cdot k + \mathbf{e}_S \text{ and } \mathbf{c} \cdot s + \hat{\mathbf{e}}_S + (\mathbf{a}_x \cdot k + \mathbf{e}_x)$$

where $\mathbf{e}_C \leftarrow \mathcal{R}_{\chi_\sigma}^{1 \times \ell}$, $\mathbf{e}_S, \hat{\mathbf{e}}_S \leftarrow \mathcal{R}_{\chi_{\sigma'}}^{1 \times \ell}$, and $\mathbf{e}_x \leftarrow \mathcal{E}_{\mathbf{a}_0, \mathbf{a}_1, x, \sigma}$.

Here, instead, we use a Rènyi divergence based argument to prove that the protocol given in Fig. 3 with $n = 1$, $H_r(\cdot, \cdot) := 0$, and $H(x, y_x) := y_x$ is unpredictable. We note, though, that Theorem 3 requires a bound Q on the permitted number of queries, this explains the four different rows in Table 2.

Theorem 3. *Assume that σ and N are $\text{poly}(\lambda)$, and $p|q$. Let $\text{dRLWE}_{q,N,\sigma,\sigma}$ hard and $\frac{q}{2p} \gg \sigma' \geq (L \cdot \sqrt{N} + 2 \cdot \sigma) \cdot \sigma \cdot N \cdot \sqrt{Q \cdot N}$ for a number of queries made Q . Let $(\mathbb{P}_0, \mathbb{V}_0), (\mathbb{P}_1, \mathbb{V}_1), (\mathbb{P}_2, \mathbb{V}_2)$ be NIZKAoKs for languages $\mathcal{L}_0, \mathcal{L}_1, \mathcal{L}_2$, then the VOPRF protocol defined in Fig. 3 with $n = 1$, $H_r(\cdot, \cdot) := 0$, and $H(x, y_x) := y_x$ is unpredictable against any PPT adversary \mathcal{A} controlling \mathcal{C} .*

Proof. We show \mathcal{A} controlling \mathcal{C}^* cannot find an unqueried request-response pair $(x^{(\tau)}, y_x^{(\tau)})$ with all but negligible probability in λ .

Hybrid₀: This is the real execution of the protocol where \mathcal{A} makes Q queries to \mathbb{S} . The server \mathbb{S} samples a key k and outputs a commitment c . For $\tau \in [Q]$, \mathcal{A} sends a query $(c_x^{(\tau)}, \pi_1^{(\tau)})$ based on x for which \mathbb{S} computes $(d_x^{(\tau)}, \pi_2^{(\tau)})$ if $\pi_1^{(\tau)}$ verifies and aborts otherwise. The adversary \mathcal{A} then computes $y_x^{(\tau)}$ based on $d_x^{(\tau)}$ and $x^{(\tau)}$ (resp. $y_x^{(\tau)}$) is added to the set \mathcal{X} (resp. \mathcal{Y}). At the end, \mathcal{A} outputs (x^*, y_x^*) and wins the game if $x^* \notin \mathcal{X}$ and c_x^* generated on x^* evaluates to y_x^* . The advantage of \mathcal{A} is the probability of \mathcal{A} winning in the unpredictability game.

Hybrid₁: Hybrid₁ is the same as Hybrid₀ except how the proofs by the server are computed. Instead of honestly generating crs_0 and crs_2 , and computing π_0 and $\pi_2^{(\tau)}$, \mathbb{S} calls the simulator for the relative proof systems. Hybrid₁ is then indistinguishable from Hybrid₀ by the ZK property of the underlying ZKAoKs.

Hybrid₂: Hybrid₂ is the same as Hybrid₁ except that in the **Query** phase we have that after \mathbb{S} receives $(c_x^{(\tau)}, \pi_1^{(\tau)})$, it calls the extractor for the underlying ZKAoK to obtain $(x^{(\tau)}, e_C^{(\tau)}, s^{(\tau)})$ and aborts if it fails to do so. By the extractability of the underlying ZKAoK Hybrid₁ is exactly like Hybrid₂ as long as the extraction does not fail. Then Hybrid₁ and Hybrid₂ are indistinguishable.

Hybrid₃: \mathbb{S} changes how $d_x^{(\tau)}$ are computed. Upon receiving $c_x^{(\tau)}$, \mathbb{S} samples $e'_S \leftarrow \mathcal{R}_{\chi_{\sigma'}}$ and $e_x^{(\tau)} \leftarrow \mathcal{E}_{\mathbf{a}_0, \mathbf{a}_1, x, \sigma}$ based on the extracted $x^{(\tau)}$. The server \mathbb{S} then sends $d_x^{(\tau)} := c_x^{(\tau)} \cdot k + e'_S + e_x^{(\tau)} - e_C^{(\tau)} \cdot k + e \cdot s^{(\tau)}$.

Since $(s^{(\tau)}, e_C^{(\tau)}, x^{(\tau)})$ were extracted from $\pi_1^{(\tau)}$, it is possible for \mathbb{S} to sample $e_x^{(\tau)}$ based on $x^{(\tau)}$ and compute $e_C^{(\tau)} \cdot k$ and $e \cdot s^{(\tau)}$. To bound the probability of the adversary winning when going from Hybrid₂ to Hybrid₃, we will show that if \mathcal{A} making Q queries can win the game in Hybrid₂ with probability ρ then its probability of winning in Hybrid₃ is also polynomial in ρ . Let $\mathcal{D}_2^{(\tau)}$ and $\mathcal{D}_3^{(\tau)}$ denote the distributions of $(d_x^{(\tau)}, \pi_2^{(\tau)})$ in Hybrid₂ and Hybrid₃ respectively.

In both Hybrid₂ and Hybrid₃, $(\pi_2^{(\tau)})$ are simulated and hence are distributed exactly the same. In Hybrid₂, $d_x^{(\tau)}$ is computed as $d_x^{(\tau)} = c_x^{(\tau)} \cdot k + e_S$ whereas in Hybrid₃ we have $d_x^{(\tau)} = c_x^{(\tau)} \cdot k + e'_S + e_x^{(\tau)} - e_C^{(\tau)} \cdot k + e \cdot s^{(\tau)}$ for $e_S, e'_S \in \mathcal{R}_{\chi_{\sigma'}}$. The distribution of $d_x^{(\tau)}$ in two hybrids can then be considered as two Gaussians with different centres. Outside $\{(d_x^{(\tau)}, \pi_2^{(\tau)})\}_{\tau \in Q}$, \mathcal{A} 's view in both hybrids consists of

$$\text{crs}_0, \text{crs}_1, \text{crs}_2, a, \mathbf{a}_1, \mathbf{a}_0, c_x^{(\tau)}$$

and consequently $c_x^{(\tau)}$. Here, we have that $\text{crs}_0, \text{crs}_1, \text{crs}_2, a, \mathbf{a}_1, \mathbf{a}_0$ and c are sampled independently from \mathcal{A} , therefore are fixed in both views. $x^{(\tau)}$ and consequently $c_x^{(\tau)}$ however are chosen by adversary which means $x^{(\tau)}$ (consequently $\mathbf{a}_x^{(\tau)}$), $s^{(\tau)}$, $e_{\mathbb{C}}^{(\tau)}$ and therefore $c_x^{(\tau)}$ can be adaptively chosen. However, note that each $c_x^{(\tau)}$ is associated with a proof $\pi_1^{(\tau)}$ proving $c_x^{(\tau)}$ is correctly computed. Since \mathbb{S} does not abort, each $\pi_1^{(\tau)}$ has to verify therefore each $c_x^{(\tau)}$ corresponds to the same distribution.

The RD between $\mathcal{D}_2^{(\tau)}$ and $\mathcal{D}_3^{(\tau)}$ is then by Lemma 4:

$$\begin{aligned} R_\alpha(\mathcal{D}_2^{(\tau)} \parallel \mathcal{D}_3^{(\tau)}) &\leq 1 \cdot \exp\left(\frac{\alpha \cdot \pi \cdot \|e_x - e_{\mathbb{C}} \cdot k + e \cdot s\|_2^2}{\sigma'^2}\right) \\ &\leq \exp\left(\frac{\alpha \cdot \pi \cdot \left(\sqrt{N} \|e_{\mathbb{C}}\|_2 \cdot \|k\|_2 + \sqrt{N} \|e\|_2 \cdot \|s\|_2 + \|e_x\|_2\right)^2}{\sigma'^2}\right) \end{aligned}$$

Since $\tau \in Q$, we can define the distributions \mathcal{D}_2 and \mathcal{D}_3 for the distribution of $(d_x^{(\tau)}, \pi_2^{(\tau)})$ for the entire hybrids. We have:

$$R_\alpha(\mathcal{D}_2 \parallel \mathcal{D}_3) \leq \exp\left(\frac{\alpha \cdot \pi \cdot Q \cdot \left(\sqrt{N} \|e_{\mathbb{C}}\|_2 \cdot \|k\|_2 + \sqrt{N} \|e\|_2 \cdot \|s\|_2 + \|e_x\|_2\right)^2}{\sigma'^2}\right)$$

Let ψ, ψ' denote the views of \mathcal{A} in Hybrid_2 and Hybrid_3 respectively. By data processing inequality of RD we then have:

$$R_\alpha(\psi \parallel \psi') \leq R_\alpha(\mathcal{D}_2 \parallel \mathcal{D}_3)$$

Let E be the event that \mathcal{A} outputs a successful prediction. By our assumption we then have $\mathcal{D}_2(E) = \rho$. Following the probability preservation property of RD:

$$\psi'(E) \geq \frac{\rho^{\frac{\alpha}{\alpha-1}}}{R_\alpha(\psi \parallel \psi')}$$

By assumption $\sigma' \geq (L \cdot \sqrt{N} + 2 \cdot \sigma) \cdot \sigma \cdot N \cdot \sqrt{Q \cdot N}$, $\|e\|_2, \|e_{\mathbb{C}}\|_2 \leq \sigma\sqrt{N}$, $\|k\|_2, \|s\|_2 \leq \sigma\sqrt{N}$, and $\|e_x\|_\infty \leq L \cdot \sigma \cdot N^{3/2}$ [4, Lemma 4]. This means $R_\alpha(\psi \parallel \psi') \leq \exp(\pi \cdot \alpha)$ and consequently $\psi'(E) \geq \rho^{\frac{\alpha}{\alpha-1}} \cdot \exp(-\alpha\pi)$ which is non-negligible if and only if ρ is non-negligible.

Hybrid₄: \mathbb{S} stops using key material k for replies to \mathbb{C}^* . The server \mathbb{S} maintains a received list for $(x^{(\tau)}, y_q)$. After receiving and verifying $c_x^{(\tau)}$, it checks if the extracted $x^{(\tau)}$ has been queried before. If $(x^{(\tau)}, y_q)$ exists in **received**, \mathbb{S} retrieves y_q from the list, samples $\bar{e}_{\mathbb{S}}^{(\tau)} \leftarrow \mathcal{R}_{\chi_{\sigma'}}$ and returns $\bar{d}_x^{(\tau)} = c \cdot s^{(\tau)} + \bar{e}_{\mathbb{S}}^{(\tau)} + y_q$. If $x^{(\tau)}$ is queried for the first time, \mathbb{S} first samples a PRF output y and then

uniformly samples a y_q such that $y_q \leftarrow \mathcal{R}_q \cap (q/p \cdot y + \mathcal{R}_{\leq q/2p})$. \mathbb{S} records $(x^{(\tau)}, y_q)$ and computes $\bar{d}_x^{(\tau)}$ the same. First, note that we can rewrite $d_x^{(\tau)}$ in Hybrid_3 as

$$\begin{aligned} d_x^{(\tau)} &= c_x^{(\tau)} \cdot k + e'_S + e_x^{(\tau)} - e_C^{(\tau)} \cdot k + e \cdot s^{(\tau)} \\ &= a \cdot s^{(\tau)} \cdot k + e_C^{(\tau)} \cdot k + \mathbf{a}_x^{(\tau)}[0] \cdot k + e'_S + e_x^{(\tau)} - e_C^{(\tau)} \cdot k + e \cdot s^{(\tau)} \\ &= a \cdot s^{(\tau)} \cdot k + e \cdot s^{(\tau)} + e_C^{(\tau)} \cdot k + \mathbf{a}_x^{(\tau)}[0] \cdot k + e'_S + e_x^{(\tau)} - e_C^{(\tau)} \cdot k \\ &= c \cdot s^{(\tau)} + e_x^{(\tau)} + \left(\mathbf{a}_x^{(\tau)}[0] \cdot k + e'_S \right) \end{aligned}$$

We have that $\mathbf{a}_x^{(\tau)}[0] \cdot k + e_x^{(\tau)}$ is indistinguishable from some uniform $u_x^{(\tau)}$ by opening up the proof of [4, Lemma 3].³ Said lemma holds under the hardness of $\text{dRLWE}_{q,N,\sigma,\sigma}$ where $\mathbf{a}_x^{(\tau)}[0] \cdot k + e_x^{(\tau)}$ can be decomposed as multiple samples of the form $a_i \cdot k + e_i$ for uniform $a_i \in \mathcal{R}_q$ and small $e_i \in \mathcal{R}_q$. Multiple queries for $\mathbf{a}_x^{(\tau)}[0] \cdot k + e_x^{(\tau)}$ can then be considered as increased number of samples for $a_i \cdot k + e_i$. Hence, by the hardness of $\text{dRLWE}_{q,N,\sigma,\sigma}$, $c \cdot s^{(\tau)} + \mathbf{a}_x^{(\tau)}[0] \cdot k + e_x^{(\tau)} + e'_S$ is indistinguishable from $d_x^{(\tau)} = c \cdot s^{(\tau)} + u_x^{(\tau)} + e'_S$ for some uniform $u_x^{(\tau)}$. Since y is a PRF output, y_q is a uniformly chosen element of a uniformly chosen interval, it is also indistinguishable from $u_x^{(\tau)}$. Finally $\bar{e}_S^{(\tau)}$ is sampled from the same distribution as e'_S , $d_x^{(\tau)}$ and $\bar{d}_x^{(\tau)}$ therefore Hybrid_3 and Hybrid_4 are indistinguishable.

Hybrid_5 : Now that the VOPRF answer does not rely on k , \mathbb{S} stops sampling a k altogether and samples a uniformly random $c \leftarrow \mathcal{R}_q$ instead. By the hardness of $\text{dRLWE}_{q,N,\sigma,\sigma}$, c in Hybrid_4 and Hybrid_5 are indistinguishable.

Now that every reply to \mathcal{A} is freshly generated and independent from any secret material, they are unpredictable. This concludes the proof. \square

From the unpredictable function, we can define a VOPRF. We first define random oracles $\text{H}: \{0, 1\}^L \times \mathcal{R}_q \rightarrow \{0, 1\}^\lambda$ and $\text{H}_r: \{0, 1\}^L \times \mathcal{R}_q \rightarrow \mathcal{R}_q$ which then are used to generate the VOPRF output on the client side. The new VOPRF protocol is depicted in Fig. 3 with $n = 1$, and new definitions of H and H_r .

The transformation is rather standard and we immediately follow that the VOPRF protocol has one-more PRF security.

Corollary 1. *Assume that σ and N are $\text{poly}(\lambda)$, and $p|q$. Let H, H_r be hash functions modeled as random oracles, and Q denote the number of queries made to the VOPRF. Let $\text{dRLWE}_{q,N,\sigma,\sigma}$ hard and $\frac{q}{2p} \gg \sigma' \geq (L \cdot \sqrt{N} + 2 \cdot \sigma) \cdot \sigma \cdot N \cdot \sqrt{Q \cdot N}$. Let $(\mathbb{P}_0, \mathbb{V}_0), (\mathbb{P}_1, \mathbb{V}_1), (\mathbb{P}_2, \mathbb{V}_2)$ be NIZKAoKs for languages $\mathcal{L}_0, \mathcal{L}_1, \mathcal{L}_2$. Then if the VOPRF protocol defined in Fig. 3 for $n = 1$ is unpredictable, it also has one-more PRF security in random oracle model against any PPT adversary \mathcal{A} controlling \mathcal{C} .*

³ In [4], it is argued that $\mathbf{a}_x^{(\tau)}[0] \cdot k + e_x^{(\tau)}$ and $u_x^{(\tau)}$ are indistinguishable directly by a lemma implicit in the underlying PRF [7]. This is incorrect as is because the lemma does not consider multiple queries.

5 Bandwidth Estimate

We give rough bandwidth estimates in Table 2. We adapt the estimation scripts from [3] and give our adapted scripts in the full version of this work. Our size estimates for ring elements are simply $d \cdot \log q$ bits, except for d_x where we use (a) that we can drop lower order bits since those are drowned by σ' anyway and (b) that we only use d_x additively when computing $d_x + r - c \cdot s$, allowing us to drop many coefficients of d_x since we only want to extract from λ many. It would be possible to amortise the zero-knowledge proofs as in [3], but we forego this optimisation here.

While these are somewhat rough estimates, they suffice to make good on our claim that the parameters we obtain are practical-*ish*.

6 Threshold Lattice (V)OPRF

As a result of the nice homomorphic properties of [4], we give lattice-based threshold/distributed VOPRFs for both n -out-of- n and t -out-of- n thresholds, when t is constant.

6.1 Threshold Verifiable Oblivious Pseudorandom Functions

A (t, n) threshold VOPRF is an extension to VOPRFs where instead of having a single server \mathbb{S} , there are n servers $\mathbb{S}_0, \dots, \mathbb{S}_{n-1}$ where any $t \leq n$ servers can collectively generate the PRF output. If $t = n$ i.e. the threshold scheme is n -out-of- n we call it a *distributed* scheme, but we may also call it *full threshold* so that we can discuss n -out-of- n and t -out-of- n together as “threshold”. Based on the setting, the initialisation phase can be done by either each \mathbb{S}_i individually or by an outside trusted authority. A threshold verifiable oblivious pseudorandom function (VOPRF) for a keyed function F is then an $n+1$ party protocol between a client \mathbb{C} and n servers $\mathbb{S}_0, \dots, \mathbb{S}_{n-1}$ consisting of following algorithms:

- $\text{Init}_{\mathbb{S}}$ is a protocol run by each \mathbb{S}_i (or a trusted authority), which on input 1^λ outputs a partial secret key sk_i and its public commitment pk_i (or the combined commitment pk).
- $\text{Init}_{\mathbb{C}}$ is a protocol run by \mathbb{C} , which on input $\{pk\}_{i \in [n]}$ (respectively pk) outputs a *state* indicating acceptance/rejection of the public commitment.
- $\text{Query}_{\mathbb{C}}$ is a protocol run by \mathbb{C} , which on input client input x and *state* outputs a blinded message \bar{x} and a state ρ .
- $\text{Query}_{\mathbb{S}}$ is a protocol run by \mathbb{S}_i which on input of client’s blinded message \bar{x} , a subset of participating users \mathcal{U} and a partial key sk_i outputs a blinded partial evaluation $y_{x,i}$.
- Finalize is a protocol run by \mathbb{C} which on input server’s blinded evaluation y_x and public commitments $\{pk_i\}_{i \in \mathcal{U}}$ (or a single pk) and a state ρ outputs the PRF output y .

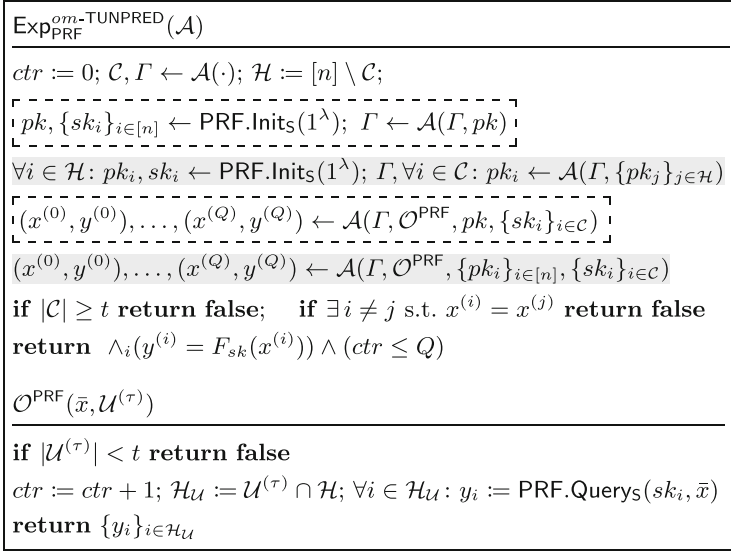


Fig. 4. The experiment $\text{Exp}_{\text{PRF}}^{\text{om-TUNPRED}}(\mathcal{A})$. Lines in grey are executed if each \mathbb{S}_i generates their own key. Lines in dashed boxes are executed if there is a trusted authority setting up the keys.

Some definitions of VOPRF security are not sufficient for the threshold case as \mathcal{A} can corrupt a subset of servers \mathcal{C} along with the client. Similarly \mathcal{A} can engage in concurrent queries for the same input. So we extend unpredictability and one-more PRF security to accommodate a set of corrupted servers and concurrent $\text{Query}_{\mathcal{C}}$ executions and again define the corresponding games. We also introduce a new algorithm **Comb** that takes a set of partial output shares and outputs a single combined output.

Definition 12 (Threshold unpredictability). A threshold PRF is said to be unpredictable if for any PPT adversary \mathcal{A} the probability of the one-more unpredictability depicted in Fig. 4 outputting 1 is negligible in λ .

Definition 13 (Threshold one-more TPRF security). A threshold PRF is said to be pseudorandom if for any PPT adversary \mathcal{A} the probability of the one-more pseudorandomness depicted in Fig. 5 outputting 1 is negligible in λ .

For both unpredictability and pseudorandomness, we assume a rushing adversary where honest parties send their messages first. The relationship between unpredictability and one-more PRF security also translates into the threshold setting where the unpredictable \mathcal{A} controls the same parties as the PRF adversary \mathcal{B} . However, we must take that that unlike in the plain one-more PRF game, here both adversaries have partial inside access to evaluating parties. However,

$\text{Exp}_{\text{PRF,H}}^{\text{om-TPRF}}(\mathcal{A})$	$\mathcal{O}^{\text{RoR}}(x^{(\tau)}, \mathcal{U}^{(\tau)})$
$ctr, q := 0; \mathcal{C}, \Gamma \leftarrow \mathcal{A}(\cdot); \mathcal{H} \leftarrow [n] \setminus \mathcal{C}$	if $ \mathcal{U}^{(\tau)} < t$ return false
$pk, \{sk_i\}_{i \in [n]} \leftarrow \text{PRF.Init}_S(1^\lambda); \Gamma \leftarrow \mathcal{A}(\Gamma, pk)$	$q := q + 1, b[q] \leftarrow \{0, 1\}$
$\forall i \in \mathcal{H}: pk_i, sk_i \leftarrow \text{PRF.Init}_S(1^\lambda)$	$sk := \text{Comb}(\mathcal{U}^{(\tau)}, \{sk_i\}_{i \in \mathcal{U}^{(\tau)}})$
$\Gamma, \forall i \in \mathcal{C}: pk_i \leftarrow \mathcal{A}(\Gamma, \{pk_j\}_{j \in \mathcal{H}})$	$y_0 \leftarrow \{0, 1\}^*$
$(i_1, \dots, i_Q, b') \leftarrow$	$y_1 \leftarrow \text{H}(x^{(i)}, F_{sk}(x^{(i)}))$
$\mathcal{A}(\Gamma, pk, \{sk_i\}_{i \in \mathcal{C}}, \text{H}, \mathcal{O}^{\text{PRF}}, \mathcal{O}^{\text{RoR}})$	return $y_{b[q]}$
$\mathcal{A}(\Gamma, \{pk_i\}_{i \in [n]}, \{sk_i\}_{i \in \mathcal{C}}, \text{H}, \mathcal{O}^{\text{PRF}}, \mathcal{O}^{\text{RoR}})$	$\mathcal{O}^{\text{PRF}}(\bar{x}, \mathcal{U}^{(\tau)})$
if $ \mathcal{C} \geq t$ return false	if $ \mathcal{U}^{(\tau)} < t$ return false
if $\exists \alpha$ s.t. $i_\alpha \notin [Q]$ return false	$ctr := ctr + 1; \mathcal{H}_U := \mathcal{U}^{(\tau)} \cap \mathcal{H}$
if $Q > q$ or $ctr \geq Q$ return false	$\forall i \in \mathcal{H}_U: y_i := \text{PRF.Query}_S(sk_i, \bar{x})$
if $\exists \alpha \neq \beta$ s.t. $i_\alpha = i_\beta$ return false	return $\{y_i\}_{i \in \mathcal{H}_U}$
return $b' := \bigoplus_{\alpha=1}^Q b[i_\alpha]$	

Fig. 5. The experiment $\text{Exp}_{\text{PRF,H}}^{\text{om-TPRF}}(\mathcal{A})$

note that \mathcal{A} is queried for its corrupt shares independent of the oracles decision regarding real or random value which prevents \mathcal{A} from trivially winning the game.

Since we are in a multiparty setting, we need NIZKAoKs that provide concurrent security as well as prevent exponential tightness loss as part of the extractability. For which, we rely on NIZKAoKs that are *straight-line extractable* i.e. can extract the witness without rewinding. For some systems, this can be achieved by using the generic transform by Katsumata [33]. However, while the transform is straightforward to use for proofs of \mathcal{L}_0 and \mathcal{L}_2 , this is not the case for \mathcal{L}_1 since we here we rely on LaBRADOR [10]. Thus, straight-line extractability for client messages is open. We note that it seems plausible we can apply the “encryption-to-the-sky” paradigm here, see e.g. [1], encrypting only that small part of the witness that we need extract to avoid blowing up bandwidth costs.

6.2 Case 1: n -out-of- n

We start with the easy case where all n participants are required to generate the pseudorandom output. The n -out-of- n distributed setting can be considered as a *multikey* application of the base scheme. This is also the reason why we used the n -out-of- n protocol with $n = 1$ in the previous sections. Instead of setting a single σ however, we allow each party to choose $\sigma_i \leq \sigma$ for publicly known σ .⁴ We have multiple servers $\mathbb{S}_0, \dots, \mathbb{S}_{n-1}$ interacting with single client \mathbb{C} , Fig. 3 depicts the protocol. For the sake of simplicity of exposure, though, we

⁴ We will expand on this idea when we discuss our t -out-of- n construction.

assume $\sigma_i = \sigma$ for all i for the rest of this section. Correctness follows from the underlying protocol where $\lfloor r + d_x - c \cdot s \rfloor_p$

$$\begin{aligned}
 &= \left\lfloor r + \sum d_{x,i} - \sum c_i \cdot s \right\rfloor_p \\
 &= \left\lfloor r + c_x \cdot \sum k_i + \sum e_{S,i} - s \cdot \sum (a \cdot k_i + e_i) \right\rfloor_p \\
 &= \left\lfloor r + a \cdot s \sum k_i + e_C \sum k_i + \mathbf{a}_x[0] \sum k_i + \sum e_{S,i} - a \cdot s \sum k_i - s \cdot \sum e_i \right\rfloor_p \\
 &= \left\lfloor r + \mathbf{a}_x[0] \sum k_i + e_C \sum k_i + \sum e_{S,i} - s \sum e_i \right\rfloor_p \\
 &= \left\lfloor r + \frac{p}{q} \cdot \mathbf{a}_x[0] \cdot k \right\rfloor_p
 \end{aligned}$$

with $k = \sum_{i \in [n]} k_i$. Correctness then follows from Theorem 2. Note however, since $k = \sum_{i \in [n]} k_i$ for $k_i \in \mathcal{R}_{\chi_{\sigma_i}}$ we have to scale the combined parameter by a factor of \sqrt{n} . We then have the following lemma.

Lemma 6. *Let $k_i \leftarrow \mathcal{R}_{\chi_{\sigma_i}}$, $k = \sum_{i \in [n]} k_i$, and $q \gg p \cdot \sigma \cdot \sqrt{L \cdot n} \cdot N$. Then the function $F_k(x) := \lfloor \mathbf{a}_x[0] \cdot k + r \rfloor_p$ is a PRF under $\text{dRLWE}_{q,N,\sigma\sqrt{n},\sigma\sqrt{n}}$ assumption.*

Security against malicious servers i.e. obliviousness and verifiability is immediate. Each S_i receives the same client input, which is the same as in single party case. For completeness we state them here:

Theorem 4. *Let σ and N be $\text{poly}(\lambda)$. Let $\text{dRLWE}_{q,N,\sigma,\sigma}$ be hard. Let $(\mathbb{P}_0, \mathbb{V}_0)$, $(\mathbb{P}_1, \mathbb{V}_1)$ be straight-line extractable NIZKAoKs for languages $\mathcal{L}_0, \mathcal{L}_1$, then the protocol in Fig. 3 is oblivious against any PPT adversary \mathcal{A} controlling all S_i .*

Theorem 5. *Let σ and N be $\text{poly}(\lambda)$. Let $\beta = 2n \cdot \sigma^2 \cdot N + \sigma' \cdot \sqrt{n \cdot N}$ and $q/p \gg \beta$. Let $(\mathbb{P}_0, \mathbb{V}_0)$, $(\mathbb{P}_2, \mathbb{V}_2)$ be straight-line extractable NIZKAoKs for languages $\mathcal{L}_0, \mathcal{L}_2$, H_r be a random oracle, and Q_H be number of queries made to such oracle. Then the protocol in Fig. 3 is verifiable against any PPT adversary \mathcal{A} controlling all S_i .*

The interesting security goal is threshold unpredictability when there is a collusion between the malicious client and some subset of servers. We show how the protocol given in Fig. 3 is unpredictable. We note that we implicitly assume that a malicious C sends the same message to honest servers. This assumption can be removed with an initial round of consistency check among the servers which we omit here.

Theorem 6. *Let σ and N be $\text{poly}(\lambda)$. Let $\text{dRLWE}_{q,N,\sigma,\sigma}$ be hard, and $\frac{q}{2p} \gg \sigma' \geq (L \cdot \sqrt{N} + (\sqrt{n} + 1) \cdot \sigma) \cdot \sigma \cdot N \cdot \sqrt{Q \cdot N}$ for a number of queries made Q . Let $(\mathbb{P}_0, \mathbb{V}_0)$, $(\mathbb{P}_1, \mathbb{V}_1)$, $(\mathbb{P}_2, \mathbb{V}_2)$ be straight-line extractable NIZKAoKs for languages $\mathcal{L}_0, \mathcal{L}_1, \mathcal{L}_2$, then the distributed VOPRF protocol defined in Fig. 3 is threshold unpredictable against malicious clients controlling up to $n - 1$ servers.*

Proof. In the Random Oracle model, if the input of H at the end of the protocol is unpredictable, then so is the output of the protocol. Hence, we show that the client-derived input to H is unpredictable. Similar to the proof of Theorem 3, we show \mathcal{A} controlling \mathbb{C}^* and a subset of servers \mathcal{C} cannot find an unqueried request-response pair $(x^{(\tau)}, y_x^{(\tau)})$ with all but negligible probability in λ .

Hybrid₀: This is the real execution of the protocol where the \mathcal{A} makes Q queries to servers. The adversary \mathcal{A} first corrupts a set of servers \mathcal{C} with fewer than $t = n$ elements and the rest of the servers denoted with \mathcal{H} behave honestly. The honest parties sample a key share k_i and each party outputs a commitment c_i alongside a proof of correct computation. For $\tau \in [Q]$, \mathcal{A} sends a query $(c_x^{(\tau)}, \pi_1^{(\tau)})$ based on some x for which honest servers compute $(d_x^{(\tau)}, \pi_2^{(\tau)})$ if $\pi_1^{(\tau)}$ verifies and aborts otherwise. For corrupted servers, \mathcal{A} can send arbitrary shares as long as $\pi_2^{(\tau)}$ verifies for each of them. The adversary \mathcal{A} then computes $y_x^{(\tau)}$ based on $d_x^{(\tau)}$ and $x^{(\tau)}$ (resp. $y_x^{(\tau)}$) is added to the set \mathcal{X} (resp. \mathcal{Y}). At the end, \mathcal{A} outputs (x^*, y_x^*) and wins the game if $x^* \notin \mathcal{X}$ and c_x^* generated on x^* evaluates to y_x^* . The advantage of \mathcal{A} is the probability of \mathcal{A} winning in the threshold unpredictability game where \mathcal{U} are all n servers.

Hybrid₁: Hybrid₁ is exactly like Hybrid₀ except how proofs by the server computed. Instead of honestly generating $\text{crs}_{0,j}, \text{crs}_{2,j}$ for $j \in \mathcal{H}$, and computing and $\pi_{0,j}, \pi_{2,j}^{(\tau)}$ each honest server calls the simulator for relative proof systems. Hybrid Hybrid₁ is then indistinguishable from Hybrid₀ by the ZK property of the underlying ZKAoKs.

Hybrid₂: Hybrid₂ is exactly like Hybrid₁ except the honest parties try to extract a witness for the corrupted parties. During **Init_S**, after the honest parties extract $\{k_i, e_i\}_{i \in \mathcal{C}}$ from $\pi_{0,i}$ using the extractor for the underlying ZKAoK, aborts if the extraction fails. Similarly during the **Query** phase after honest servers receive $c_x^{(\tau)}, \pi_1^{(\tau)}$, it calls the extractor to obtain $(x^{(\tau)}, e_C^{(\tau)}, s^{(\tau)})$. By the extractability of the underlying ZKAoKs Hybrid₁ is exactly like Hybrid₂ unless the extraction fails, and Hybrid₁ and Hybrid₂ are indistinguishable.

Hybrid₃: $\mathbb{S}_{i'}$ changes how $d_{x,i'}^{(\tau)}$ is computed for $i' \in \mathcal{H}$. Upon receiving $c_x^{(\tau)}$, fix an index i' . For every other honest party, the computation continues as before. The server $\mathbb{S}_{i'}$ then derives the combined key $k = \sum_{i \in [n]} k_i$ and error $e = \sum_{i \in [n]} e_i$ and samples $e_x^{(\tau)} \leftarrow \mathcal{E}_{\mathbf{a}_0, \mathbf{a}_1, x, \sigma}$ based on the extracted $x^{(\tau)}$. The server $\mathbb{S}_{i'}$ finally computes $\bar{d}_{x,i'}^{(\tau)} = c_x^{(\tau)} \cdot k_{i'} + e_{\mathbb{S}_{i'}}^{(\tau)} + e_x^{(\tau)} - e_C^{(\tau)} \cdot k + e \cdot s^{(\tau)}$ sends $\bar{d}_{x,i'}^{(\tau)}$ as its share. The rest follows as before.

The difference between Hybrid₂ and Hybrid₃ is in the error term of $\mathbb{S}_{i'}$'s share where there is an added term of $e_x^{(\tau)} - e_C^{(\tau)} \cdot k + e \cdot s^{(\tau)}$. Using the same R enyi argument in Hybrid₃ of Theorem 3, we conclude if \mathcal{A} has a winning probability in Hybrid₂, it also does a winning probability polynomial of said probability in

Hybrid₃. Since $\|k\|_2 \leq \sigma\sqrt{n}$ however, σ' has an increased factor compared to the single party case.⁵

Hybrid₄: We stop using combined key k for deriving $\bar{d}_x^{(\tau)}$. Each honest server maintains a received list for $(x^{(\tau)}, y_q)$. After receiving and verifying $c_x^{(\tau)}$ checks if the extracted $x^{(\tau)}$ has been queried before. If $(x^{(\tau)}, y_q)$ exists in received, $\mathbb{S}_{i'}$ retrieves y_q from the list and samples $\bar{e}_{\mathbb{S},i'}^{(\tau)} \leftarrow \mathcal{R}_{\chi_{\sigma'}}$ and returns $\bar{d}_{x,i'}^{(\tau)} = c \cdot s^{(\tau)} - c_x^{(\tau)} \cdot \sum_{j \neq i'} k_j + \bar{e}'^{(\tau)} + y_q$. If $x^{(\tau)}$ is queried for the first time, $\mathbb{S}_{i'}$ first samples an output y and then uniformly samples a y_q such that $y_q \leftarrow \mathcal{R}_q \cap (q/p \cdot y + \mathcal{R}_{\leq q/2p})$. Each server records $(x^{(\tau)}, y_q)$ and computes $\bar{d}_{x,i}^{(\tau)}$ the same. In Hybrid₃ $\bar{d}_{x,i}^{(\tau)}$ can be rewritten as:

$$\begin{aligned} \bar{d}_{x,i'}^{(\tau)} &= c_x^{(\tau)} \cdot k_{i'} + e_{\mathbb{S},i'}^{(\tau)} + e_x^{(\tau)} - e_C^{(\tau)} \cdot k + e \cdot s^{(\tau)} \\ &= c_x^{(\tau)} \cdot k - c_x^{(\tau)} \cdot \sum_{j \neq i'} k_j + e_{\mathbb{S},i'}^{(\tau)} + e_x^{(\tau)} - e_C^{(\tau)} \cdot k + e \cdot s^{(\tau)} \\ &= a \cdot s^{(\tau)} \cdot k + e_C^{(\tau)} \cdot k + \mathbf{a}_x^{(\tau)}[0] \cdot k - c_x^{(\tau)} \cdot \sum_{j \neq i'} k_j + e_{\mathbb{S},i'}^{(\tau)} + e_x^{(\tau)} - e_C^{(\tau)} \cdot k + e \cdot s^{(\tau)} \\ &= c \cdot s^{(\tau)} - c_x^{(\tau)} \cdot \sum_{j \neq i'} k_j + \mathbf{a}_x^{(\tau)}[0] \cdot k + e_x^{(\tau)} + e_{\mathbb{S},i'}^{(\tau)} \end{aligned}$$

Using the same argument in Hybrid₄ for Theorem 3, Hybrid₃ and Hybrid₄ are indistinguishable.

Hybrid₅: We modify honest parties' shares so that each of them includes additional error terms $e_i'^{(\tau)} \leftarrow \mathcal{R}_{\chi_{\sigma'}}$ and $e_{x,i}'^{(\tau)} \leftarrow \mathcal{E}_{\mathbf{a}_0, \mathbf{a}_1, x, \sigma}$, and the adjusted share $\bar{d}_{x,i'}^{(\tau)}$ includes the subtraction of these shares $-\sum_{i \neq i' \in \mathcal{H}} (e_i'^{(\tau)} + e_{x,i}'^{(\tau)})$. The rest follows as before. In Hybrid₄ each honest party outside i' computes their share as $d_{x,i}^{(\tau)} = c_x^{(\tau)} \cdot k_i + e_{\mathbb{S},i}^{(\tau)}$ whereas in Hybrid₅ $d_{x,i}^{(\tau)} = c_x^{(\tau)} \cdot k_i + e_{\mathbb{S},i}^{(\tau)} + e_i'^{(\tau)} + e_{x,i}'^{(\tau)}$. The difference is then how error terms are distributed for two Gaussians of parameter σ' with two different centers. Using a similar argument to Hybrid₃,⁶ we conclude if \mathcal{A} can win in Hybrid₄ with some probability, it also has a winning probability polynomial in the said probability in Hybrid₅.

Hybrid₆: We remove the dependency on partial key shares for honest parties. Except i' , each honest server samples a uniformly random $u_i^{(\tau)} \leftarrow \mathcal{R}_q$ and computes their share as $\bar{d}_{x,i}^{(\tau)} := u_i^{(\tau)} + e_{\mathbb{S},i}^{(\tau)}$ instead. Similarly, $\mathbb{S}_{i'}$ defines its share as $\bar{d}_{x,i'}^{(\tau)} = c \cdot s^{(\tau)} - c_x^{(\tau)} \cdot \sum_{j \in \mathcal{C}} k_j + \bar{e}_{\mathbb{S},i'}^{(\tau)} + y_q - \sum_{i \neq i' \in \mathcal{H}} u_i^{(\tau)}$. The rest proceeds

⁵ Note that this hybrid also changes the combined $d_x^{(\tau)}$ since the additional error term carries over. By Lemma 1 the error term is statistically close to a Gaussian with parameter $\sigma'\sqrt{n}$ using a similar R enyi argument as above, but with easier to satisfy parameters, already satisfied by the parameters considered in the main text.

⁶ Note that we do not need to consider the combined $d_x^{(\tau)}$ as $\mathbb{S}_{i'}$ adjusts its share based on the added error terms.

as before. In **Hybrid₅** after the addition of noise terms, each partial evaluation is $d_{x,i}^{(\tau)} = c_x^{(\tau)} \cdot k_i + e_{\mathbb{S},i}^{(\tau)} + e_i'^{(\tau)} + e_{x,i}'^{(\tau)} = (a \cdot s^{(\tau)} + e_C^{(\tau)}) \cdot k_i + e_i'^{(\tau)} + \mathbf{a}_x^{(\tau)}[0] \cdot k_i + e_{x,i}'^{(\tau)} + e_{\mathbb{S},i}^{(\tau)}$. Using the same argument to **Hybrid₄** of Theorem 3, $\mathbf{a}_x^{(\tau)}[0] \cdot k_i + e_{x,i}'^{(\tau)}$ is indistinguishable from uniform. Replacing these terms with uniform ones, $d_{x,i}^{(\tau)}$ in **Hybrid₅** and $u_i^{(\tau)} + e_{\mathbb{S},i}^{(\tau)}$, consequently **Hybrid₅** and **Hybrid₆** are indistinguishable.

Hybrid₇: Now that the function evaluation does not rely on the combined key honest servers stop deriving key shares k_j altogether. During initialization each server samples random $c_i \leftarrow \mathcal{R}_q$. Then by the hardness of $\text{dRLWE}_{q,N,\sigma\sqrt{n},\sigma\sqrt{n}}$, c in **Hybrid₆** and **Hybrid₇** are indistinguishable.

Since every reply to \mathcal{A} is freshly generated and independent from the secret combined key k and the honest key shares, they are unpredictable. Thus we conclude the proof. \square

Now that the protocol is threshold unpredictable, we can also argue it has threshold one-more PRF security.

Corollary 2. *Let σ and N be $\text{poly}(\lambda)$. Let $\text{dRLWE}_{q,N,\sigma,\sigma}$ be hard and $\frac{q}{2p} \gg \sigma' \geq (L \cdot \sqrt{N} + (\sqrt{n} + 1) \cdot \sigma) \cdot \sigma \cdot N \cdot \sqrt{Q \cdot N}$ for a number of queries made Q . Let $(\mathbb{P}_0, \mathbb{V}_0), (\mathbb{P}_1, \mathbb{V}_1), (\mathbb{P}_2, \mathbb{V}_2)$ be straight-line extractable NIZKAoKs for languages $\mathcal{L}_0, \mathcal{L}_1, \mathcal{L}_2$ and H, H_r be hash functions modeled as random oracles, then if the distributed VOPRF protocol defined in Fig. 6 is threshold unpredictable, it also has threshold one-more PRF security against any PPT adversary \mathcal{A} controlling \mathcal{C} and a subset of servers \mathcal{C} of size at most $t - 1$.*

6.3 Case 2: t -out-of- n

We now switch to the more interesting case of arbitrary thresholds i.e. t -out-of- n with $t \leq n$. We cannot directly use the additive homomorphism of the underlying operation but tweak it to our setting. Dealing with t -out-of- n shares in a lattice setting is not trivial against malicious adversaries, and we here work around known issues by assuming a trusted setup. As mentioned above, we consider this a realistic assumption for OPRFs as most use cases of threshold OPRFs utilise the functionality to prevent a single point of failure during execution rather than to achieve execution among untrusted parties. Still, this is a limitation of this work.

Similarly, we assume that in the context of distributed OPRFs, neither n nor $\binom{n}{t}$ is large. Hence we can consider a separate set of keys for different thresholds of servers since we only have $\binom{n}{t}$ of such sets. However, trivially combining all partial keys will result in $\binom{n}{t}$ different combined keys and consequently public commitments which now (i) requires the client to know which servers are replying for correctness (ii) the client will receive different PRF outputs for the same input for different threshold sets.

Instead, we make use of the setting we are in and consider a different way of representing these $\binom{n}{t}$ sets. We delegate key generation to a trusted authority

which in return allows us to use different *additive shares* of the same combined key. On a high level, key generation proceeds as follows: The trusted authority first samples a combined key from the combined distribution of individual keys. For every threshold set \mathcal{T} , the authority fixes indices i_a and i_b . For every server i outside i_a and i_b in the set, it samples partial keys from a smaller distribution. For i_b , it samples a partial key from a wider distribution. The trusted authority then computes the key of i_a as the difference of the combined key and $t-1$ partial keys and rejects the key with a certain probability. If rejected, the process starts again for the threshold set. If not the trusted authority proceeds to the next set.

The first key idea is that we can choose distributions of keys for differing parameters as long as each of them are bounded from below for RLWE security and above for correctness and noise drowning. Hence, choosing a key with some $\sigma_{\mathcal{L}} > \sigma$ allows us to control the rejection probability. The second key idea is that while the last share is not exactly the same as the sampled keys, if rejected correctly it is statistically indistinguishable from the distribution of i_b 's key and thus still secure. This rejection is similar to the inefficient variant Dilithium-G signature discussed in [17]. Since the last key share is computed as $k - \sum_{i \neq i_a} k_{i,\mathcal{T}} = k - \sum_{i \neq i_a, i_b} k_{i,\mathcal{T}} - k_{i_b,\mathcal{T}}$, we can treat it as a Gaussian centered around $k - \sum_{i \neq i_a, i_b} k_{i,\mathcal{T}}$ and use Lemmas 1 to 3 to find correct M, t, T to make the last share within a negligible statistical distance of a sampled Gaussian with parameter $\sigma_{\mathcal{L}}$ by Lemma 3. For $\sigma = 3.2$ and $N = 4096$, a threshold of 5 parties can have $\sigma_{\mathcal{L}} = 1024$ with $M = 131$ repetitions per key with all but negligible probability 2^{-102} . We emphasise that for large sizes of $\binom{n}{t}$ this can result in long key generation times but does not affect the actual PRF evaluation. For security argument, we will assume $\binom{n}{t}$ is $\text{poly}(\lambda)$.

Remark 4. Our approach can be considered as a variant of *replicated secret sharing* [26] using qualified sets. Hence, we could consider algorithms such as in [16] for key generation instead. However, we highlight some key differences: (a) Secret sharing is done for long term keys rather than online randomness which renders the $\binom{n}{t}$ overhead more acceptable. (b) Our final shares are with overwhelming probability from specific Gaussian distributions rather than uniformly random in order to preserve the structure of the protocol. (c) Since shares of Shamir secret sharing are arbitrarily large any advantage regarding easy conversion into Shamir secret sharing is not relevant in our context.

One downside to this approach, we cannot show verifiability for individual partial evaluations as public commitments $k_{i,\mathcal{T}}$ for all subsets \mathcal{T} of size t do not exist. Hence it is not possible for \mathbb{S}_i to prove $d_{x,i}$ is computed correctly with respect to a partial key $k_{i,\mathcal{U}}$. One solution to this for the trusted authority to publish public commitments for each $k_{i,\mathcal{T}}$ which however would require $t \cdot \binom{n}{t}$ commitments to be published and for \mathbb{C} to know which subset of users are participating in PRF execution. This is worse than the trivial construction of having $\binom{n}{t}$ different combined keys.

Instead, we combine the *cut-and-choose* type of approach in [4] with a weaker proof system. The intuition is while we cannot prove that correct $k_{i,\mathcal{U}}$ is used

for generating $d_{x,i}$, we can verify that a small $k_{i,\mathcal{U}}$ is used consistently across multiple evaluations. If one of these evaluations points can be checked with respect to a publicly known value, we can argue that every evaluation used the correct combined key. This does not guarantee each individual partial evaluation is done correctly but assures \mathbb{C} the final output is correct.

We change the protocol as follows. During setup there is a public fixed input x' and its evaluation $y_{x'}$ (under the key k) known both to the client and the servers. During $\text{Query}_{\mathbb{C}}$, instead of a single input x , \mathbb{C} blinds two inputs x_0, x_1 . To do that, the client first decides on a random bit b' . For $i \neq b'$, the client uses the private input $x_i = x$ for some $x \in \{0, 1\}^L$ and for $i = b'$, $x_i = x'$. The client \mathbb{C} then runs the computation for two c_x values and sends $c_{x,i}, \pi_{1,i}$ pairs. Each server \mathbb{S}_j then runs partial evaluations on each of them and sends a proof $\pi_{2,j}$ to prove that the same short $k_{j,\mathcal{U}}$ have been used for computing all $c_{x,i}$ values. During Finalize , \mathbb{C} first verifies $\pi_{2,i}$ and then computes two different y_x values and checks if $y_{x,b'} = y_{x'}$. If everything verifies, \mathbb{C} uses $y_{x,i}$ for $i \neq b'$ as its output.

We depict this t -out-of- n VOPRF construction with a trusted setup in Fig. 6. Note that $\pi_{2,i}$ are computed with a different proof system $\mathbb{P}'_2, \mathbb{V}'_2$ for language \mathcal{L}'_2 since it's slightly different from $\mathbb{P}_2, \mathbb{V}_2$. It can however still be initiated with the same proof systems discussed in the full version of this work.

Correctness follows from the linearity of the additive secret sharing. Obliviousness is once again immediate as the client's input to the t servers do not change. We first show that the protocol described has verifiability:

Theorem 7. *Let σ, N , and $\binom{n}{t}$ be poly(λ). Let $\beta = 2t \cdot \sigma^2 \cdot N + \sigma' \cdot \sqrt{t \cdot N}$ and $q/p \gg \beta$. Let $(\mathbb{P}'_2, \mathbb{V}'_2)$ be straight-line extractable NIZKAoK for language \mathcal{L}'_2 , H_r be a random oracle, and Q_H be number of queries made to such oracle. Let*

$$N \cdot \left(\log q - \log \left(\sigma \cdot \sqrt{(t+1) \cdot N} \right) - \log \left(\frac{q}{2p} \right) \right) > \lambda.$$

Then the protocol in Fig. 6 is verifiable against any PPT adversary \mathcal{A} controlling a subset of servers \mathcal{C} of size at most $t - 1$.

Proof. In verifiability game, the challenger will abort and \mathcal{A} will trivially lose if the checks during Finalize fail. For \mathcal{A} to win, $y_{x,i}$ for $i \neq b'$ derived by \mathbb{C} must be different from the actual PRF. Similar to Theorem 2, \mathcal{A} can only find an x^* that would cause r^* to force an incorrect evaluation only with negligible probability. If $\pi_{2,j}$ verifies for each $j \in \mathcal{U}$, then there exists short $\{k_{j,\mathcal{U}}\}_{j \in \mathcal{U}}$ used in each of the $\{d_{x,j,i}\}_{i \in \{0,1\}, j \in \mathcal{U}}$. Then if $y_{x,b'} = y_{x'}$ each server \mathbb{S}_j knows $k_{j,\mathcal{U}}^*$ for some k^* where $k^* = \sum_{j \in \mathcal{U}} k_{j,\mathcal{U}}^*$.

Since $\pi_{2,j}$ verifies, we have $\|k_{j,\mathcal{U}}^*\|_{\infty} \leq \|k_{j,\mathcal{U}}^*\|_2 \leq \sigma \cdot \sqrt{N}$ consequently $\|k^*\|_{\infty} \leq \sigma \cdot \sqrt{t \cdot N}$, and $\|e_{\mathbb{S},j,i}\|_{\infty} \leq \|e_{\mathbb{S},j,i}\|_2 \leq \sigma' \cdot \sqrt{N}$. Since $q/p \gg 2t \cdot \sigma^2 \cdot N + \sigma' \cdot \sqrt{t \cdot N}$, we have:

CRS Setup:

- $\mathbf{a}_0, \mathbf{a}_1 \leftarrow \mathcal{R}_q^{1 \times \ell}$.
- $a \leftarrow \mathcal{R}_q^{1 \times \ell}$.
- Sample $\text{crs}_1, \text{crs}'_2$ for $\mathbb{P}_1, \mathbb{P}'_2$.
- Fix an input $x' \in \{0, 1\}^L$.

Initialisation:

- Init_5 : A trusted authority executes:
 - $k \leftarrow \mathcal{R}_{\chi_\sigma}, e \leftarrow \mathcal{R}_{\chi_\sigma}$.
 - $c \leftarrow a \cdot k + e \pmod q, r' \leftarrow \text{H}_r(x', c) \in \mathcal{R}_q^{1 \times \ell}$.
 - $\mathbf{a}_{x'} := \mathbf{a}_{x_0} \cdot G^{-1}(\dots(\mathbf{a}_{x_{L-2}} \cdot G^{-1}(\mathbf{a}_{x_{L-1}}))\dots) \pmod q$.
 - $y_{x'} := \mathbf{a}_{x'}[0] \cdot k + r'$.
 - For every threshold set \mathcal{T} of size t :
 - * Fix indices $i_a, i_b \in \mathcal{T}$.
 - * For $i = i_b, k_{i_b, \mathcal{T}} \leftarrow \mathcal{R}_{\chi_{\sigma_L}}$.
 - * For $i \in \mathcal{T} \setminus \{i_a, i_b\}, k_{i, \mathcal{T}} \leftarrow \mathcal{R}_{\chi_\sigma}$.
 - * $k_{i_a, \mathcal{T}} := k - \sum_{i \neq i_a \in \mathcal{T}} k_{i, \mathcal{T}}$
 - * With a probability $1 - \min(1, D_{\sigma_L}(k_{i_a, \mathcal{T}})) / D_{k - \sum k_{i, \mathcal{T}, \sigma_L}(k_{i_a, \mathcal{T}})$ repeat the process for \mathcal{T} .
- Send $k_{i, \mathcal{T}}$ to server \mathbb{S}_i for every \mathcal{T} that \mathbb{S}_i is part of, broadcast $c, y_{x'}$.

Query:

1. **Query $_C$** : \mathbb{C} executes the following with the input $(x \in \{0, 1\}^L, \text{crs}_1, \text{crs}'_2)$
 - $b' \leftarrow \{0, 1\}$.
 - For each index $i \in \{0, 1\}$:
 - If $i = b'$, $x_i = x'$ otherwise $x_i = x$
 - $s_i \leftarrow \mathcal{R}_{\chi_\sigma}, e_{\mathbb{C}, i} \leftarrow \mathcal{R}_{\chi_\sigma}^{1 \times \ell}$.
 - $\mathbf{a}_{x, i} := \mathbf{a}_{x_0} \cdot G^{-1}(\dots(\mathbf{a}_{x_{L-2}} \cdot G^{-1}(\mathbf{a}_{x_{L-1}}))\dots) \pmod q$.
 - $c_{x, i} \leftarrow a \cdot s_i + e_{\mathbb{C}, i} + \mathbf{a}_{x, i}[0] \pmod q$.
 - $\pi_{1, i} \leftarrow \mathbb{P}_1(x_i, s_i, e_{\mathbb{C}, i} : \text{crs}_1, c_{x, i}, a, \mathbf{a}_0, \mathbf{a}_1)$.

and broadcasts $\{(c_{x, i}, \pi_{1, i})\}_{i \in \{0, 1\}}$ to every \mathbb{S}_j .
2. **Query $_S$** : $\mathbb{S}_j \in \mathcal{U}$ executes the following for each index i after receiving $\{(c_{x, i}, \pi_{1, i})\}_{i \in \{0, 1\}}$
 - $b \leftarrow \mathbb{V}_1(\text{crs}_1, c_{x, i}, \mathbf{a}_0, \mathbf{a}_1, \pi_{1, i})$, output abort if $b = 0$.
 - $e_{\mathbb{S}, j, i} \leftarrow \mathcal{R}_{\chi_{\sigma'}}$, $d_{x, j, i} := c_{x, i} \cdot k_{j, \mathcal{U}} + e_{\mathbb{S}, j, i} \pmod q$.
 - $\pi_{2, j} \leftarrow \mathbb{P}'_2(k_{j, \mathcal{U}}, e_{\mathbb{S}, j, i} : \text{crs}'_2, \{d_{x, j, i}, c_{x, i}\}_{i \in M}, a)$.

and sends $(\{(d_{x, j, i})\}_{i \in \{0, 1\}}, \pi_{2, j})$ to \mathbb{C} and outputs \perp .
3. **Finalize**: \mathbb{C} finally executes the following after receiving $(\{(d_{x, j, i})\}_{i \in \{0, 1\}})$ from $\mathbb{S}_j \in \mathcal{U}$.
 - $b_j \leftarrow \mathbb{V}'_2(\text{crs}_2, \{d_{x, j, i}, c_{x, i}\}_{i \in \{0, 1\}}, \pi_{2, j})$, output abort with i if $b_j = 0$.
 - $d_{x, i} := \sum_j d_{x, j, i}, r_i \leftarrow \text{H}_r(x_i, c) \in \mathcal{R}_q^{1 \times \ell}, y_{x, i} := \lfloor d_{x, i} + r_i - c \cdot s_i \rfloor_p$.
 - If $y_{x, i} \neq y_{x'}$ for $i = b'$ abort.
 - $y \leftarrow \text{H}(x_i, y_{x, i})$ for $i \neq b'$.

and outputs y .

Fig. 6. t -out-of- n VOPRF Construction with Trusted Setup

$$\begin{aligned} \left[\sum_j d_{x,j,b'} + r' - c \cdot s \right]_p &= \left[\mathbf{a}_x[0] \cdot k^* + r' + \left(e_{\mathbb{C}} \cdot k^* - e \cdot s + \sum_j e_{\mathbb{S},j,b'} \right) \right]_p \\ &= \lfloor \mathbf{a}_x[0] \cdot k^* + r' \rfloor_p \end{aligned}$$

with overwhelming probability. Then, \mathcal{A} can only win if it can find $k^* \neq k$ such that $\lfloor \mathbf{a}_{x'}[0] \cdot k^* + r' \rfloor_p = \lfloor \mathbf{a}_{x'}[0] \cdot k + r' \rfloor_p$. Rearranging the terms we get $\lfloor \mathbf{a}_{x'}[0] \mid 1 \rfloor \cdot \left[\frac{k^* - k}{e'} \right] = \mathbf{0} \pmod q$ for some $e' \in \mathcal{R}_q$, $\|e'\|_\infty \leq q/(2p)$. By our assumption we have that there are $(2\sigma \cdot \sqrt{t \cdot N})^N \cdot (q/(2p))^N$ possible choices for $(k^* - k, e')$ but over the randomness of $\mathbf{a}_0[0], \mathbf{a}_1[0]$, the probability of obtaining 0 is $1/q^N$. Thus, with high probability such a k^* does not exist. Hence if the evaluation for $i = b'$ is correct, $k^* = k$.

Since the same $k_{j,\mathcal{U}}^*$ and consequently the same $k^* = k$ are used for computing $y_{x,i}, i \neq b'$; the evaluation must also be correct if $y_{x,b'}$ is correct. This concludes the proof. \square

Remark 5. Our proof above relies on the absences of any SIS solution to $\lfloor \mathbf{a}_{x'} \mid 1 \rfloor$. First, our bound is rather loose, by first extracting a worst-case ℓ_∞ bound from the ℓ_2 bound established by the NIZKAoK and then constructing a box of solutions with this ℓ_∞ bound. A tighter approximation would be accomplished by bounding the number of integer points inside the ℓ_2 ball established by the NIZKAoK directly. Moreover, an alternative approach, giving smaller parameters, is to instead rely on a computational SIS assumption wrt the infinity norm and with unbalanced entries. This problem was considered in [20,44]. Indeed, even assuming an infinity norm bound of $q/4$ for all components, the difficulty of the resulting SIS instance is comparable to λ as given in Table 2 according to the lattice estimator [5].

We now show how the protocol has threshold unpredictability.

Theorem 8. *Let σ, N , and $\binom{n}{t}$ be $\text{poly}(\lambda)$. Let $t' = o(\log(N))$, $T \leq \sigma\sqrt{(t-1)N}$ and $\alpha = T/\sigma_{\perp}$. Let $M = \exp(t'/\alpha + 1/2 \cdot \alpha^{-2})$. Let $\text{dRLWE}_{q,N,\sigma,\sigma}$ be hard, and $\frac{q}{2p} \gg \sigma' \geq (L \cdot \sqrt{N} + 2 \cdot \sigma) \cdot \sigma \cdot N \cdot \sqrt{Q \cdot N}$ for the number of queries made Q . Let $(\mathbb{P}_1, \mathbb{V}_1), (\mathbb{P}'_2, \mathbb{V}'_2)$ be straight-line extractable NIZKAoKs for language $\mathcal{L}_1, \mathcal{L}'_2$. Then the (t, n) threshold OPRF protocol defined in Fig. 6 is threshold unpredictable against malicious clients controlling up to $t - 1$ servers.*

We give the proof in the full version of this work. Finally, since the protocol is threshold unpredictable, we can also argue it has threshold one-more PRF security.

Corollary 3. *Let σ, N , and $\binom{n}{t}$ be $\text{poly}(\lambda)$. Let $t' = o(\log(N))$, $T \leq \sigma\sqrt{(t-1)N}$ and $\alpha = T/\sigma_{\perp}$. Let $M = \exp(t'/\alpha + 1/2 \cdot \alpha^{-2})$. Let $\text{dRLWE}_{q,N,\sigma,\sigma}$ be hard, and $\frac{q}{2p} \gg \sigma' \geq (L \cdot \sqrt{N} + 2 \cdot \sigma) \cdot \sigma \cdot N \cdot \sqrt{Q \cdot N}$ for the number of queries made Q . Let $(\mathbb{P}_1, \mathbb{V}_1), (\mathbb{P}'_2, \mathbb{V}'_2)$ be straight-line extractable NIZKAoKs for language $\mathcal{L}_1, \mathcal{L}'_2$ and \mathbf{H}, \mathbf{H}_r be hash functions modeled as random oracles then if the*

(t, n) threshold OPRF protocol defined in Fig. 6 is threshold unpredictable, it also has threshold one-more PRF security against any PPT adversary \mathcal{A} controlling \mathcal{C} and a subset of servers \mathcal{C} of size at most $t - 1$.

Acknowledgements. We thank reviewers for their valuable comments and pointing out the issue with both [4] and our unpredictability proofs. This work was supported in part by UKRI grant EP/Y02432X/1. Part of this work was done while Kamil Doruk Gur was at SandboxAQ and supported in part by NSF award CNS-2154705.

References

1. Agrawal, S., Kirshanova, E., Stehlé, D., Yadav, A.: Practical, round-optimal lattice-based blind signatures. In: Yin, H., Stavrou, A., Cremers, C., Shi, E. (eds.) ACM CCS 2022. pp. 39–53. ACM Press (Nov 2022). <https://doi.org/10.1145/3548606.3560650>
2. Alamati, N., Policharla, G.V., Raghuraman, S., Rindal, P.: Improved alternating-moduli PRFs and post-quantum signatures. Cryptology ePrint Archive, Report 2024/582 (2024), <https://eprint.iacr.org/2024/582>
3. Albrecht, M.R., Davidson, A., Deo, A., Gardham, D.: Crypto dark matter on the torus - oblivious PRFs from shallow PRFs and TFHE. In: Joye, M., Leander, G. (eds.) EUROCRYPT 2024, Part VI. LNCS, vol. 14656, pp. 447–476. Springer, Cham (May 2024). https://doi.org/10.1007/978-3-031-58751-1_16
4. Albrecht, M.R., Davidson, A., Deo, A., Smart, N.P.: Round-optimal verifiable oblivious pseudorandom functions from ideal lattices. In: Garay, J. (ed.) PKC 2021, Part II. LNCS, vol. 12711, pp. 261–289. Springer, Cham (May 2021). https://doi.org/10.1007/978-3-030-75248-4_10
5. Albrecht, M.R., Player, R., Scott, S.: On the concrete hardness of Learning with Errors. Journal of Mathematical Cryptology **9**(3), 169–203 (2015)
6. Bai, S., Lepoint, T., Roux-Langlois, A., Sakzad, A., Stehlé, D., Steinfeld, R.: Improved security proofs in lattice-based cryptography: Using the Rényi divergence rather than the statistical distance. Journal of Cryptology **31**(2), 610–640 (Apr 2018). <https://doi.org/10.1007/s00145-017-9265-9>
7. Banerjee, A., Peikert, C.: New and improved key-homomorphic pseudorandom functions. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part I. LNCS, vol. 8616, pp. 353–370. Springer, Berlin, Heidelberg (Aug 2014). https://doi.org/10.1007/978-3-662-44371-2_20
8. Basso, A.: POKE: A framework for efficient PKEs, split KEMs, and OPRFs from higher-dimensional isogenies. Cryptology ePrint Archive, Report 2024/624 (2024), <https://eprint.iacr.org/2024/624>
9. Beullens, W., Dodgson, L., Faller, S., Hesse, J.: The 2Hash OPRF framework and efficient post-quantum instantiations. Cryptology ePrint Archive, Report 2024/450 (2024), <https://eprint.iacr.org/2024/450>
10. Beullens, W., Seiler, G.: LaBRADOR: Compact proofs for R1CS from module-SIS. In: Handschuh, H., Lysyanskaya, A. (eds.) CRYPTO 2023, Part V. LNCS, vol. 14085, pp. 518–548. Springer, Cham (Aug 2023). https://doi.org/10.1007/978-3-031-38554-4_17
11. Boneh, D., Ishai, Y., Passelègue, A., Sahai, A., Wu, D.J.: Exploring crypto dark matter: New simple PRF candidates and their applications. In: Beimel, A., Dziembowski, S. (eds.) TCC 2018, Part II. LNCS, vol. 11240, pp. 699–729. Springer, Cham (Nov 2018). https://doi.org/10.1007/978-3-030-03810-6_25

12. Boneh, D., Kogan, D., Woo, K.: Oblivious pseudorandom functions from isogenies. In: Moriai, S., Wang, H. (eds.) ASIACRYPT 2020, Part II. LNCS, vol. 12492, pp. 520–550. Springer, Cham (Dec 2020). https://doi.org/10.1007/978-3-030-64834-3_18
13. Brakerski, Z., Vaikuntanathan, V.: Constrained key-homomorphic PRFs from standard lattice assumptions - or: How to secretly embed a circuit in your PRF. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015, Part II. LNCS, vol. 9015, pp. 1–30. Springer, Berlin, Heidelberg (Mar 2015). https://doi.org/10.1007/978-3-662-46497-7_1
14. Casacuberta, S., Hesse, J., Lehmann, A.: Sok: Oblivious pseudorandom functions. In: 7th IEEE European Symposium on Security and Privacy, EuroS&P 2022. pp. 625–646. IEEE (2022). <https://doi.org/10.1109/EuroSP53844.2022.00045>, <https://doi.org/10.1109/EuroSP53844.2022.00045>
15. Chillotti, I., Gama, N., Georgieva, M., Izabachène, M.: TFHE: Fast fully homomorphic encryption over the torus. *Journal of Cryptology* **33**(1), 34–91 (Jan 2020). <https://doi.org/10.1007/s00145-019-09319-x>
16. Cramer, R., Damgård, I., Ishai, Y.: Share conversion, pseudorandom secret-sharing and applications to secure computation. In: Kilian, J. (ed.) TCC 2005. LNCS, vol. 3378, pp. 342–362. Springer, Berlin, Heidelberg (Feb 2005). https://doi.org/10.1007/978-3-540-30576-7_19
17. Damgård, I., Orlandi, C., Takahashi, A., Tibouchi, M.: Two-round n-out-of-n and multi-signatures and trapdoor commitment from lattices. *Journal of Cryptology* **35**(2), 14 (Apr 2022). <https://doi.org/10.1007/s00145-022-09425-3>
18. Davidson, A., Goldberg, I., Sullivan, N., Tankersley, G., Valsorda, F.: Privacy pass: Bypassing internet challenges anonymously. *PoPETs* **2018**(3), 164–180 (Jul 2018). <https://doi.org/10.1515/popets-2018-0026>
19. Dinur, I., Goldfeder, S., Halevi, T., Ishai, Y., Kelkar, M., Sharma, V., Zaverucha, G.: MPC-friendly symmetric cryptography from alternating moduli: Candidates, protocols, and applications. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021, Part IV. LNCS, vol. 12828, pp. 517–547. Springer, Cham, Virtual Event (Aug 2021). https://doi.org/10.1007/978-3-030-84259-8_18
20. Esgin, M.F., Steinfeld, R., Zhao, R.K.: MatRiCT⁺: More efficient post-quantum private blockchain payments. In: 2022 IEEE Symposium on Security and Privacy. pp. 1281–1298. IEEE Computer Society Press (May 2022). <https://doi.org/10.1109/SP46214.2022.9833655>
21. Everspaugh, A., Chatterjee, R., Scott, S., Juels, A., Ristenpart, T.: The pythia PRF service. In: Jung, J., Holz, T. (eds.) USENIX Security 2015. pp. 547–562. USENIX Association (Aug 2015)
22. Faller, S., Ottenhues, A., Ottenhues, J.: Composable oblivious pseudo-random functions via garbled circuits. *Cryptology ePrint Archive*, Report 2023/1176 (2023), <https://eprint.iacr.org/2023/1176>
23. Freedman, M.J., Ishai, Y., Pinkas, B., Reingold, O.: Keyword search and oblivious pseudorandom functions. In: Kilian, J. (ed.) TCC 2005. LNCS, vol. 3378, pp. 303–324. Springer, Berlin, Heidelberg (Feb 2005). https://doi.org/10.1007/978-3-540-30576-7_17
24. Gajland, P., de Kock, B., Quaresma, M., Malavolta, G., Schwabe, P.: Swoosh: Practical lattice-based non-interactive key exchange. *Cryptology ePrint Archive*, Report 2023/271 (2023), <https://eprint.iacr.org/2023/271>
25. Heimberger, L., Meisingseth, F., Rechberger, C.: Oprfs from isogenies: Designs and analysis. *Cryptology ePrint Archive*, Paper 2023/639 (2023), <https://eprint.iacr.org/2023/639>, <https://eprint.iacr.org/2023/639>

26. Ito, M., Saito, A., Nishizeki, T.: Secret sharing scheme realizing general access structure. *Electronics and Communications in Japan (Part III: Fundamental Electronic Science)* **72**(9), 56–64 (1989)
27. Jarecki, S., Kiayias, A., Krawczyk, H.: Round-optimal password-protected secret sharing and T-PAKE in the password-only model. In: Sarkar, P., Iwata, T. (eds.) *ASIACRYPT 2014, Part II*. LNCS, vol. 8874, pp. 233–253. Springer, Berlin, Heidelberg (Dec 2014). https://doi.org/10.1007/978-3-662-45608-8_13
28. Jarecki, S., Kiayias, A., Krawczyk, H., Xu, J.: Highly-efficient and composable password-protected secret sharing (or: How to protect your bitcoin wallet online). In: 2016 IEEE European Symposium on Security and Privacy (EuroS&P). pp. 276–291 (2016). <https://doi.org/10.1109/EuroSP.2016.30>
29. Jarecki, S., Krawczyk, H., Resch, J.: Threshold partially-oblivious PRFs with applications to key management. *Cryptology ePrint Archive, Report 2018/733* (2018), <https://eprint.iacr.org/2018/733>
30. Jarecki, S., Krawczyk, H., Xu, J.: OPAQUE: An asymmetric PAKE protocol secure against pre-computation attacks. In: Nielsen, J.B., Rijmen, V. (eds.) *EUROCRYPT 2018, Part III*. LNCS, vol. 10822, pp. 456–486. Springer, Cham (Apr / May 2018). https://doi.org/10.1007/978-3-319-78372-7_15
31. Jarecki, S., Liu, X.: Efficient oblivious pseudorandom function with applications to adaptive OT and secure computation of set intersection. In: Reingold, O. (ed.) *TCC 2009*. LNCS, vol. 5444, pp. 577–594. Springer, Berlin, Heidelberg (Mar 2009). https://doi.org/10.1007/978-3-642-00457-5_34
32. Kaluderovic, N., Cheng, N., Mitrokovtsa, K.: A post-quantum distributed OPRF from the legendre PRF. *Cryptology ePrint Archive, Report 2024/544* (2024), <https://eprint.iacr.org/2024/544>
33. Katsumata, S.: A new simple technique to bootstrap various lattice zero-knowledge proofs to QROM secure NIZKs. In: Malkin, T., Peikert, C. (eds.) *CRYPTO 2021, Part II*. LNCS, vol. 12826, pp. 580–610. Springer, Cham, Virtual Event (Aug 2021). https://doi.org/10.1007/978-3-030-84245-1_20
34. Keelveedhi, S., Bellare, M., Ristenpart, T.: DupLESS: Server-aided encryption for deduplicated storage. In: King, S.T. (ed.) *USENIX Security 2013*. pp. 179–194. USENIX Association (Aug 2013)
35. Langlois, A., Stehlé, D., Steinfeld, R.: GGHLite: More efficient multilinear maps from ideal lattices. In: Nguyen, P.Q., Oswald, E. (eds.) *EUROCRYPT 2014*. LNCS, vol. 8441, pp. 239–256. Springer, Berlin, Heidelberg (May 2014). https://doi.org/10.1007/978-3-642-55220-5_14
36. Lehmann, A.: ScrambleDB: Oblivious (chameleon) pseudonymization-as-a-service. *PoPETs* **2019**(3), 289–309 (Jul 2019). <https://doi.org/10.2478/popets-2019-0048>
37. Lyubashevsky, V.: Lattice signatures without trapdoors. In: Pointcheval, D., Johansson, T. (eds.) *EUROCRYPT 2012*. LNCS, vol. 7237, pp. 738–755. Springer, Berlin, Heidelberg (Apr 2012). https://doi.org/10.1007/978-3-642-29011-4_43
38. Lyubashevsky, V., Nguyen, N.K., Plancon, M.: Lattice-based zero-knowledge proofs and applications: Shorter, simpler, and more general. In: Dodis, Y., Shrimpton, T. (eds.) *CRYPTO 2022, Part II*. LNCS, vol. 13508, pp. 71–101. Springer, Cham (Aug 2022). https://doi.org/10.1007/978-3-031-15979-4_3
39. Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. In: Gilbert, H. (ed.) *EUROCRYPT 2010*. LNCS, vol. 6110, pp. 1–23. Springer, Berlin, Heidelberg (May / Jun 2010). https://doi.org/10.1007/978-3-642-13190-5_1

40. Micciancio, D., Peikert, C.: Hardness of SIS and LWE with small parameters. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part I. LNCS, vol. 8042, pp. 21–39. Springer, Berlin, Heidelberg (Aug 2013). https://doi.org/10.1007/978-3-642-40041-4_2
41. Seres, I.A., Horváth, M., Burcsi, P.: The legendre pseudorandom function as a multivariate quadratic cryptosystem: Security and applications. Cryptology ePrint Archive, Report 2021/182 (2021), <https://eprint.iacr.org/2021/182>
42. Stehlé, D., Steinfeld, R., Tanaka, K., Xagawa, K.: Efficient public key encryption based on ideal lattices. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 617–635. Springer, Berlin, Heidelberg (Dec 2009). https://doi.org/10.1007/978-3-642-10366-7_36
43. Yang, R., Au, M.H., Zhang, Z., Xu, Q., Yu, Z., Whyte, W.: Efficient lattice-based zero-knowledge arguments with standard soundness: Construction and applications. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part I. LNCS, vol. 11692, pp. 147–175. Springer, Cham (Aug 2019). https://doi.org/10.1007/978-3-030-26948-7_6
44. Zhang, J., Yu, Y., Fan, S., Zhang, Z., Yang, K.: Tweaking the asymmetry of asymmetric-key cryptography on lattices: KEMs and signatures of smaller sizes. In: Kiayias, A., Kohlweiss, M., Wallden, P., Zikas, V. (eds.) PKC 2020, Part II. LNCS, vol. 12111, pp. 37–65. Springer, Cham (May 2020). https://doi.org/10.1007/978-3-030-45388-6_2



Unbounded ABE for Circuits from LWE, Revisited

Valerio Cini^(✉) and Hoeteck Wee

NTT Research, Sunnyvale, CA, USA
cini.valerio@gmail.com

Abstract. We introduce new lattice-based techniques for building ABE for circuits with unbounded attribute length based on the LWE assumption, improving upon the previous constructions of Brakerski and Vaikuntanathan (CRYPTO 16) and Goyal, Koppula, and Waters (TCC 16). Our main result is a simple and more efficient unbounded ABE scheme for circuits where only the circuit depth is fixed at set-up; this is the first unbounded ABE scheme for circuits that rely only on black-box access to cryptographic and lattice algorithms. The scheme achieves semi-adaptive security against unbounded collusions under the LWE assumption. The encryption time and ciphertext size are roughly $3\times$ larger than the prior bounded ABE of Boneh et al. (EUROCRYPT 2014), substantially improving upon the encryption times in prior works. As a secondary contribution, we present an analogous result for unbounded inner product predicate encryption that satisfies weak attribute-hiding.

1 Introduction

Attribute-based encryption (ABE) [SW05, GPSW06] is a generalization of public-key encryption to support fine-grained access control for encrypted data. Here, ciphertexts are associated with attributes like ‘(author:Waters), (inst:UT), (topic:PK)’ and keys with access policies like ((topic:Thy) OR (topic:Qu)) AND (NOT(inst:CWI)), and decryption is possible only when the attributes satisfy the access policy. Over past decade, substantial progress has been made in the design and analysis of ABE schemes, leading to a large families of schemes that achieve various trade-offs between efficiency, security and underlying assumptions. Meanwhile, ABE has found use in a variety of settings such as electronic medical records, messaging systems and online social networks; companies like Cloudflare already use ABE to distribute private key storage across data centers [Ver23].

As institutions grow and with new emerging and more complex applications for ABE, we need ABE schemes that can readily accommodate the addition of new roles, entities, attributes and policies. This means that the ABE set-up algorithm should put no restriction on the length of the attributes or the size of the policies that will be used in the ciphertexts and keys. This requirement was introduced and first realized in the work of Lewko and Waters [LW11] under the term *unbounded ABE*; we would henceforth also refer to standard ABE as *bounded ABE*. The Lewko-Waters schemes rely on pairings without

random oracles, and have since been improved and extended in several subsequent works [Lew12, OT12, RW13, Att14, KL15, Att16, CGKW18]. All of these schemes are limited to policies described by NC^1 circuits or branching programs, as is the case with all pairing-based ABE schemes.

In 2016, Brakerski and Vaikuntanathan (BV16) gave the first construction of unbounded ABE for circuits [BV16] based on the Learning with Errors (LWE) assumption, building upon bounded ABE schemes in [BGG⁺14, GVW13]. This was followed shortly by a generalization in Goyal-Koppula-Waters (GKW16) [GKW16] showing a generic compiler of bounded ABE schemes to unbounded ones assuming additionally adaptively secure identity-based encryption (IBE). Both BV16 and GKW16 schemes also achieve *semi-adaptive security* [CW14], a slight strengthening of selective security where an adversary can choose its encryption challenge after seeing the public key. We note that both schemes do inherit the limitation from prior bounded ABE for circuits, in that the depth of the circuits needs to be fixed at set-up; nonetheless, this already capture NC^1 circuits, whose depth can be bounded by security parameter λ .

One theoretical and practical draw-back of the BV16 and GKW16 schemes is that they require non-black-box access to the underlying cryptographic building blocks and algorithms, which not only incur substantial efficiency overheads during encryption, but also make these schemes harder to implement and deploy in practice. In particular, the BV16 scheme uses homomorphic computation of a pseudorandom function, whereas the GKW16 applies circuit garbling techniques to the underlying ABE schemes. This is in contrast to the aforementioned pairing-based unbounded ABE schemes as well as prior LWE-based ABE schemes for circuits, which avoid non-black-box techniques.

1.1 Our Results

In this work, we present new LWE techniques for building simple and more efficient unbounded ABE from bounded ones that avoid non-black-box techniques, leading to substantial savings in encryption times. Our constructions are inspired in part by prior pairing-based schemes in [Lew12, OT12, CGKW18], as well as ideas from [Agr17] on how to combine inner-product functionality and BGGHNSVV14 structure.

Unbounded ABE for Circuits. Our main result is a more efficient unbounded ABE for circuits of a-priori bounded depth d based on the LWE assumption. From a feasibility stand-point, this is the first unbounded ABE scheme for circuits that rely only on black-box access to cryptographic and lattice algorithms. As with BV16 and GKW16, we achieve semi-adaptive security against unbounded collusions. For depth d circuits over ℓ -bit inputs where only d is fixed at set-up, we have

$$|\text{mpk}| = \text{poly}(d, \lambda), \quad |\text{ct}| = \ell \cdot \text{poly}(d, \lambda), \quad |\text{sk}| = \ell \cdot \text{poly}(d, \lambda)$$

Compared to the BGGHNSVV14 ABE (which only achieves selective security),

- the encryption time and the ciphertext size are roughly $3\times$ larger;
- the decryption time incurs an additive $\ell \cdot \text{poly}(d, \lambda)$ overhead; the overhead is sublinear in the BGGHNSVV14 ABE decryption time $s \cdot \text{poly}(d, \lambda)$.

The efficiency savings over prior works are as follows:

- compared to the BV16 unbounded ABE, the savings in running times and ciphertext/key sizes are two-fold: cutting down $\text{poly}(d + d_{\text{PRF}})$ dependencies to $\text{poly}(d)$ where d_{PRF} is the depth of a PRF and removing additive overheads corresponding to PRF evaluation; in particular, (i) encryption time in BV16 is mostly dominated by homomorphic evaluation of a PRF with ℓ -bit output, and our encryption time should be a $\text{poly}(\lambda)$ factor smaller, (ii) for constant-depth circuits and shallow circuits where $d \ll d_{\text{PRF}}$, our scheme is substantially more efficient for all running times and sizes.
- compared to the GKW16 unbounded ABE, our encryption time and ciphertext size are a multiplicative $O(\lambda)$ factor smaller, which corresponds to the overhead from garbling the BGGHNSVV14 ABE encryption circuit.

Decryption times in our scheme and GKW16 are comparable to that in BGGHNSVV14 ABE, and faster than that in BV16. In all three unbounded ABE schemes, the secret key has two components: a private component corresponding to a BGGHNSVV14 ABE secret key of size $\text{poly}(d, \lambda)$ as well as a public component of size $\ell \cdot \text{poly}(d, \lambda)$ that can be reused across all keys for circuits of input length ℓ . In BV16, the private component is slightly larger $\text{poly}(d + d_{\text{PRF}}, \lambda)$, but the public component is just $\ell + \text{poly}(\lambda)$ bits.

Unbounded Inner Product Predicate Encryption. Next, we turn our attention to inner product predicate encryption (IPPE) [KSW08], where ciphertexts are associated with (row) vectors $\mathbf{x} \in \mathbb{Z}_q^\ell$ and keys with vectors $\mathbf{y} \in \mathbb{Z}_q^\ell$ and decryption is possible only if their inner product $\mathbf{x}\mathbf{y}^\top$ equals 0. In addition to hiding the message as in ABE, we require attribute-hiding, namely that ciphertexts hide the attribute \mathbf{x} . Unbounded IPPE schemes can be realized from pairings [OT12] with black-box techniques, or from LWE by applying the GKW16 transformation to the bounded IPPE scheme of Agrawal, Freeman, and Vaikuntanathan (AFV11) [AFV11] with non-black-box techniques.

Our second result is a more efficient unbounded inner product predicate encryption scheme based on the LWE assumption. We achieve semi-adaptive, weak attribute-hiding security against unbounded collusions. For vectors over \mathbb{Z}_q^ℓ where only q is fixed at set-up, we have

$$|\text{mpk}| = \text{poly}(\log q, \lambda), \quad |\text{ct}| = \ell \cdot \text{poly}(\log q, \lambda), \quad |\text{sk}| = \ell \cdot \text{poly}(\log q, \lambda)$$

Compared to the scheme derived from combining GKW16 with the AFV11 scheme, our encryption time and ciphertext size are a multiplicative $O(\lambda \log q)$ factor smaller, where the $O(\lambda)$ factor comes from garbling as before, and the $O(\log q)$ comes from the fact that we can directly support attributes over \mathbb{Z}_q in our scheme. In contrast, the techniques in BV16 and GKW16 are inherently limited to attributes over a binary alphabet.

Scheme	Time(Enc)	Time(Dec)
[BGG ⁺ 14]	$T_{\text{Enc}}(\ell, d) = \ell \cdot \text{poly}(\lambda, d)$	$T_{\text{Dec}}(s, d) = s \cdot \text{poly}(\lambda, d)$
[BV16]	$O(T_{\text{Enc}}(\ell, d + d_{\text{PRF}})) + s_{\text{PRF}} \cdot \text{poly}(\lambda, d + d_{\text{PRF}})$	$T_{\text{Dec}}(s + s_{\text{PRF}}, d + d_{\text{PRF}}) + \ell \cdot \text{poly}(\lambda, d)$
[GKW16]	$\text{poly}(\lambda) \cdot T_{\text{Enc}}(\ell, d)$	$T_{\text{Dec}}(s, d) + \ell \cdot \text{poly}(\lambda, d)$
this work	$(3 + o(1)) \cdot T_{\text{Enc}}(\ell, d)$	$T_{\text{Dec}}(s, d) + \ell \cdot \text{poly}(\lambda, d)$

Fig. 1. Comparison of running times with prior KP-ABE for circuits of size s and depth d over $\{0, 1\}^\ell$. [BGG⁺14] is used as a benchmark. Here, $d_{\text{PRF}} = O(\log \lambda + \log \ell)$ and $s_{\text{PRF}} = O(\ell \cdot \lambda)$ denotes the depth and size of a PRF for ℓ -bit inputs. The ciphertext sizes satisfy an analogous relationship, where we replace T_{Enc} by S_{Enc} , namely: $S_{\text{Enc}}(\ell, d) = \ell \cdot \text{poly}(d, \lambda)$, $O(S_{\text{Enc}}(\ell, d + d_{\text{PRF}}))$, $\text{poly}(\lambda) \cdot S_{\text{Enc}}(\ell, d)$, $(3 + o(1)) \cdot S_{\text{Enc}}(\ell, d)$ respectively. The total key sizes are $S_{\text{Dec}}(d) = \text{poly}(\lambda, d)$, $S_{\text{Dec}}(d + d_{\text{PRF}}) + \ell + \text{poly}(\lambda)$, $S_{\text{Dec}}(d) + \ell \cdot \text{poly}(\lambda)$, $\ell \cdot S_{\text{Dec}}(d)$ respectively. In each row, the dominant term for Time(Dec) comes from $T_{\text{Dec}}(s, d)$.

Our Construction, in a Nutshell. The starting point, following BV16 and GKW16, is to compute/sample a BGGHNSVV14 mpk during key generation, which would be reused across all key queries; this (deceptively) simple idea buys us both short mpk and semi-adaptive security. Decryption would then first reconstruct a BGGHNSVV14 ciphertext w.r.t. mpk and then proceed as in BGGHNSVV14 decryption. The key technical novelties in this work lie in how we enable reconstruction of BGGHNSVV14 ciphertext using simple LWE algebra and techniques (instead of non-black-box techniques), along with a new simple idea for handling circuit with different input lengths in the key queries.

1.2 Technical Overview

We proceed to provide a technical overview of our constructions, focusing on the unbounded ABE.

BGGHNSVV14 ABE. We begin with an overview of the BGGHNSVV14 bounded ABE scheme for depth d circuits over $\{0, 1\}^\ell$ [BGG⁺14]. Let $\mathbf{A} \in \mathbb{Z}_q^{n \times \ell \cdot m}$ be a matrix where $q \in \mathbb{N}$ is prime and $m = O(n \log q)$. Given \mathbf{A} and a circuit $f : \{0, 1\}^\ell \rightarrow \{0, 1\}$ of depth d , we can derive [BGG⁺14, GSW13] a matrix $\mathbf{A}_f \in \mathbb{Z}_q^{n \times m}$ such that for any $\mathbf{x} \in \{0, 1\}^\ell$, we can compute a low-norm matrix $\mathbf{H}_{\mathbf{A}, f, \mathbf{x}}$ satisfying

$$(\mathbf{A} - \mathbf{x} \otimes \mathbf{G}) \cdot \mathbf{H}_{\mathbf{A}, f, \mathbf{x}} = \mathbf{A}_f - f(\mathbf{x}) \cdot \mathbf{G}, \quad (1)$$

where $\mathbf{G} \in \mathbb{Z}_q^{n \times m}$ is the gadget matrix [MP12] and $\|\mathbf{H}_{\mathbf{A},f,\mathbf{x}}\| \leq m^{O(d)}$. The ABE scheme is as follows, omitting error terms in the ciphertext:

$$\begin{aligned} \text{mpk} &= \mathbf{A}_0 \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{b} \leftarrow \mathbb{Z}_p^n, \mathbf{A} \leftarrow \mathbb{Z}_q^{n \times \ell \cdot m}. \\ \text{ct} &= \left(\overbrace{\mathbf{s} \cdot \mathbf{A}_0}^{c_0}, \overbrace{\mathbf{s} \cdot \mathbf{b}^\top + \mu \cdot \lfloor q/2 \rfloor}^{c_2}, \overbrace{\mathbf{s} \cdot (\mathbf{A} - \mathbf{x} \otimes \mathbf{G})}^{c_3} \right), \mathbf{s} \leftarrow \mathbb{Z}_q^n. \\ \text{sk} &= \mathbf{k}_f^\top \leftarrow \mathcal{D}_{\mathbb{Z}^{2m}, \tau} \text{ s.t. } [\mathbf{A}_0 \mid \mathbf{A}_f] \cdot \mathbf{k}_f^\top = \mathbf{b}^\top. \end{aligned}$$

Decryption computes an approximation to $\mu \cdot \lfloor q/2 \rfloor$ for $f(\mathbf{x}) = 0$ as follows:

$$\mathbf{c}_2 - \overbrace{[\mathbf{c}_0 \mid \mathbf{c}_3 \cdot \mathbf{H}_{\mathbf{A},f,\mathbf{x}}]}^{\approx \mathbf{s} \cdot [\mathbf{A}_0 \mid \mathbf{A}_f]} \cdot \mathbf{k}_f^\top.$$

Compressing mpk. As a warm-up, we describe an ABE for circuits over $\{0, 1\}^\ell$ where $|\text{mpk}| = \text{poly}(d, \lambda)$. It is convenient to then write \mathbf{A} in the BGGHNSVV14 ABE as $\mathbf{A}_i \in \mathbb{Z}_q^{n \times m}, i \in [\ell]$ and \mathbf{c}_3 in the ciphertext as $\mathbf{s} \cdot (\mathbf{A}_i - x_i \mathbf{G}), i \in [\ell]$. We want to sample \mathbf{A}_i during key generation (and not set-up) and then compute $\mathbf{s} \cdot (\mathbf{A}_i - x_i \mathbf{G})$ during decryption. In particular,

- mpk now contains random matrices $\mathbf{B}_0, \mathbf{W}, \mathbf{V} \leftarrow \mathbb{Z}_q^{n \times m}$ in addition to \mathbf{A}_0, \mathbf{b} , and msk contains the trapdoors for \mathbf{A}_0 and \mathbf{B}_0 ;
- the ciphertext contains

$$\mathbf{s}_i \cdot \mathbf{B}_0, \quad \{\mathbf{s}_i \cdot \mathbf{W} + \mathbf{s} \cdot \mathbf{G}, \quad \mathbf{s}_i \cdot \mathbf{V} + x_i \cdot \mathbf{s} \cdot \mathbf{G}\}_{i \in [\ell]},$$

where $\mathbf{s}, \mathbf{s}_i \leftarrow \mathbb{Z}_q^n$ are sampled during encryption;

- during decryption, we compute

$$\mathbf{s} \cdot (\mathbf{A}_i - x_i \cdot \mathbf{G}) \approx (\mathbf{s}_i \cdot \mathbf{W} + \mathbf{s} \cdot \mathbf{G}) \cdot \mathbf{G}^{-1}(\mathbf{A}_i) - (\mathbf{s}_i \cdot \mathbf{V} + x_i \cdot \mathbf{s} \cdot \mathbf{G}) + \mathbf{s}_i \cdot \mathbf{B}_0 \cdot \mathbf{Z}_i$$

where $\mathbf{Z}_i \leftarrow \mathbf{B}_0^{-1}(\mathbf{V} - \mathbf{W} \cdot \mathbf{G}^{-1}(\mathbf{A}_i))$ is provided in the secret key.

- key generation for f returns the same $(\mathbf{A}_i, \mathbf{Z}_i)$ across all secret keys—generated using a PRF key in msk so that we don't need to maintain state across key queries—as well as a BGGHNSVV14 secret key for f .

This is sufficient for functionality. However, an adversary can also compute $\mathbf{s}_i \cdot \mathbf{B}_0 \cdot \mathbf{Z}_j$ and thus $\mathbf{s}(\mathbf{A}_j - x_i \cdot \mathbf{G})$ for any $i \neq j$. To prevent this attack, we replace \mathbf{W} with $\mathbf{W} + i \cdot \mathbf{G}$ in both the ciphertext and the secret key. This yields the following ABE scheme:

$$\text{mpk} = \mathbf{A}_0, \mathbf{B}_0, \mathbf{W}, \mathbf{V} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{b} \leftarrow \mathbb{Z}_p^n.$$

$$\text{ct} = \left(\overbrace{\mathbf{s} \cdot \mathbf{A}_0}^{c_0}, \overbrace{\{\mathbf{s}_i \cdot \mathbf{B}_0\}}^{c_{1,i}}, \overbrace{\{\mathbf{s}_i \cdot (\mathbf{W} + i \cdot \mathbf{G}) + \mathbf{s} \cdot \mathbf{G}\}}^{c_{2,i}}, \overbrace{\{\mathbf{s}_i \cdot \mathbf{V} + x_i \cdot \mathbf{s} \cdot \mathbf{G}\}}_{i \in [\ell]}^{c_{3,i}}, \overbrace{\mathbf{s} \cdot \mathbf{b}^\top + \mu \cdot \lfloor q/2 \rfloor}^{c_4} \right), \mathbf{s}, \mathbf{s}_i \leftarrow \mathbb{Z}_q^n.$$

$$\text{sk} = (\{\mathbf{Z}_j, \mathbf{R}_j\}_{j \in [\ell]}, \mathbf{k}_f), \text{ where } \mathbf{Z}_j, \mathbf{R}_j \text{ are fixed across all keys}$$

$$\begin{bmatrix} \mathbf{Z}_j \\ \mathbf{R}_j \end{bmatrix} \leftarrow \mathcal{D}_{\mathbb{Z}^{2m \times m}, \tau} \text{ s.t. } [\mathbf{B}_0 \mid \mathbf{W} + j \cdot \mathbf{G}] \cdot \begin{bmatrix} \mathbf{Z}_j \\ \mathbf{R}_j \end{bmatrix} = \mathbf{V},$$

$$\mathbf{A}_j = \mathbf{G} \cdot \mathbf{R}_j$$

$$\mathbf{k}_f^\top \leftarrow \mathcal{D}_{\mathbb{Z}^{2m}, \tau} \text{ s.t. } [\mathbf{A}_0 \mid \mathbf{A}_f] \cdot \mathbf{k}_f^\top = \mathbf{b}^\top$$

Decryption first uses

$$\mathbf{c}_{1,i} \cdot \mathbf{Z}_i + \mathbf{c}_{2,i} \cdot \mathbf{R}_i - \mathbf{c}_{3,i} \approx \mathbf{s} \cdot (\mathbf{A}_i - x_i \cdot \mathbf{G}) \quad (2)$$

to recover a BGGHNSVV14 ciphertext, and then proceed as in BGGHNSVV14 decryption.

Proof Overview. The proof proceeds in two steps: *Step 1.* For $i = 1, 2, \dots, \ell$, we rely on pseudorandomness of $\mathbf{s}_i \cdot [\mathbf{B}_0 \mid \mathbf{W} + i \cdot \mathbf{G}]$ to replace $\mathbf{c}_{1,i}, \mathbf{c}_{2,i}$ with random and rewrite $\mathbf{c}_{3,i}$ in terms of $\mathbf{s} \cdot (\mathbf{A}_i - x_i \cdot \mathbf{G})$ using (2). In more detail,

- we program $\mathbf{W} + i \cdot \mathbf{G} = \mathbf{B}_0 \cdot \tilde{\mathbf{W}}$ for a random low-norm $\tilde{\mathbf{W}}$;
- we sample random Gaussian $\mathbf{Z}_i, \mathbf{R}_i$ and program \mathbf{V} accordingly;
- for all $j \neq i$, we sample $\mathbf{Z}_j, \mathbf{R}_j$ using the trapdoor for $[\mathbf{B}_0 \mid \mathbf{W} + j \cdot \mathbf{G}] = [\mathbf{B}_0 \mid \mathbf{B}_0 \cdot \tilde{\mathbf{W}} + (j - i) \cdot \mathbf{G}]$;
- use the LWE assumption to replace $\mathbf{s}_i \cdot \mathbf{B}_0$ with random.

Step 2. Run the BGGHNSVV14 security proof. This step knows the trapdoor for $\tilde{\mathbf{B}}_0$, which is used to solve for \mathbf{Z}_j .

Our construction and proof strategy achieve semi-adaptive security for the same reason as in BV16, GKW16: the matrices \mathbf{A}_i from the BGGHNSVV14 mpk are sampled after the adversary chooses its encryption challenge attribute.

Getting to an Unbounded ABE Scheme. In an unbounded ABE scheme, we need to allow both an honest party and an adversary to ask for keys corresponding to functions with different input lengths. The previous scheme already satisfies the syntax of an unbounded ABE scheme, since we can sample the \mathbf{A}_i matrices “on the fly”, while using a PRF to ensure that we use the same \mathbf{A}_i across all secret keys. However, it is insecure as an unbounded ABE: consider an attack that fixes \mathbf{x}^* for the challenge ciphertext and then query a f such that f evaluates to true on a prefix of \mathbf{x}^* . To defeat this attack, we add $\mathbf{s} \cdot (\mathbf{B}_1 - |\mathbf{x}| \cdot \mathbf{G})$ to the challenge ciphertext and modify sk for $f : \{0, 1\}^\ell \rightarrow \{0, 1\}$ to satisfy

$$[\mathbf{A}_0 \mid \mathbf{A}_f \mid \mathbf{B}_1 - \ell \cdot \mathbf{G}] \cdot \mathbf{k}_f^\top = \mathbf{b}^\top$$

To handle semi-adaptive security, we would simply guess $|\mathbf{x}^*|$ when simulating \mathbf{B}_1 in the reduction, which incurs an additional polynomial loss. This is where GKW16 uses an adaptively secure IBE (for which the known instantiations from LWE in e.g. [Yam16] are more complex than their selectively secure counterparts), since they embed $|\mathbf{x}^*|$ into the identity of the IBE ciphertext. The BV16 scheme similarly embeds $|\mathbf{x}^*|$ as part of the attribute in an “outer ABE” that plays an analogous role to the IBE ciphertext in GKW16.

Inner Product Predicate Encryption Scheme. Here, we start with the AFV11 inner product predicate encryption, where an encryption for an attribute $\mathbf{x} = (x_1, \dots, x_\ell) \in \mathbb{Z}_q^\ell$ is exactly the same as that in the BGGHNSVV14 ABE. Here, we exploit the fact that our construction directly support attributes over \mathbb{Z}_q . We can then proceed essentially as before.

1.3 Discussion

Additional Comparison with Prior Approaches. As mentioned at the beginning of Sect. 1.1, our constructions are inspired in part by prior pairing-based schemes. To better convey this, compare our ciphertext with that in the KP-ABE for arithmetic span programs in [CGKW18, Section 9.3], where an encryption of $\mathbf{x} = (x_1, \dots, x_\ell) \in \mathbb{Z}_p^\ell$ has the form:

$$\left(\overbrace{[\mathbf{s}\mathbf{B}_0]_1}^{c_0}, \overbrace{[\mathbf{s}_i\mathbf{B}_0]_1}^{c_{2,i}}, \overbrace{[\mathbf{s}'_i\mathbf{B}_0]_1}^{c'_{2,i}}, \overbrace{[\mathbf{s}_i(\mathbf{W} + i\mathbf{W}_1) + \mathbf{s}'_i(\mathbf{W}' + i\mathbf{W}'_1) + \mathbf{s}(\mathbf{V} + x_i\mathbf{V}')]_1}^{c_{1,i}}, [\mathbf{s}\mathbf{b}^\top]_{T\mu} \right), \mathbf{s}, \mathbf{s}_i, \mathbf{s}'_i \leftarrow \mathbb{Z}_p^k.$$

where $[\cdot]_1$ denotes exponentiation in the group G_1 . Our $\mathbf{s}_i(\mathbf{W} + i \cdot \mathbf{G})$ is inspired by $\mathbf{s}_i(\mathbf{W} + i \cdot \mathbf{W}_1)$ above. However, note that $\mathbf{s}, \mathbf{s}x_i$ appear together as $\mathbf{s}(\mathbf{V} + x_i\mathbf{V}')$ in $c_{1,i}$. In our scheme, $\mathbf{s}\mathbf{G}, x_i\mathbf{s}\mathbf{G}$ appear separately in $\mathbf{s}_{2,i}$ and $\mathbf{s}_{3,i}$ respectively.

In our analysis, we implicitly treat $\{\mathbf{s}_i(\mathbf{W} + i \cdot \mathbf{G})\}_{i \in [\ell]}$ as independent IBE ciphertexts (for the LWE-based IBE in [ABB10]) corresponding to the identities $1, 2, \dots, \ell$ with randomness \mathbf{s}_i . This is again inspired by the pairing-based scheme [CGKW18] which uses IBE techniques in a similar way. IBE schemes are also used in the BV16, GKW16 constructions in a generic manner, whereas our schemes exploit specific algebraic structure in the underlying IBE.

Our construction can be viewed as using a one-key secure inner product functional encryption (IPFE) scheme to compute $\mathbf{s}(\mathbf{A}_i - x_i\mathbf{G})$, where the IPFE ciphertext encrypts $(\mathbf{s}, x_i\mathbf{s})$. The IPFE approach was used in [Agr17, Section 5] to construct a “bounded” ABE for circuits with semi-adaptive security. Our construction is simpler in that we do not need to encrypt a LWE error term, but also more delicate since we want an unbounded ABE scheme.

Perspective. Apart from the landmark results of ABE for circuits from LWE about a decade ago now, research on LWE-based ABE has largely lagged behind their pairing-based counter-parts. One reason is that we have a much larger arsenal of techniques in the pairings world, which exploit the rich algebraic structure in pairing groups. We see this work as taking another step towards discovering analogues of these algebraic techniques in the LWE setting, in the specific context of realizing short \mathbf{mpk} . We stress that realizing short \mathbf{mpk} (where $|\mathbf{mpk}|$ is much shorter than the ciphertext attributes) is not only relevant for constructing unbounded ABE and IPPE schemes, but also a necessity for several outstanding open problems in the LWE-based ABE literature, notably (i) CP-ABE for unbounded size circuits (even just NC^1), (ii) ABE for DFA and Turing machines, (iii) broadcast encryption where the total parameter size $|\mathbf{mpk}| + |\text{ct}| + |\mathbf{sk}|$ is sublinear in the total number of users, all of which we have made much more substantial progress in the pairings setting. We hope that developing new algebraic techniques for short \mathbf{mpk} as well as LWE analogues of existing pairing-based techniques in this work could help facilitate progress on these open problems.

2 Preliminaries

Notations. We use boldface lower case for row vectors (e.g. \mathbf{r}) and boldface upper case for matrices (e.g. \mathbf{R}). For integral vectors and matrices (i.e., those over \mathbb{Z}), we use the notation $\|\mathbf{r}\|$, $\|\mathbf{R}\|$ to denote the maximum absolute value over all the entries. We use $v \leftarrow D$ to denote a random sample from a distribution D , as well as $v \leftarrow S$ to denote a uniformly random sample from a set S . We use \approx_s and \approx_c as the abbreviation for statistically close and computationally indistinguishable. We denote by $\mathcal{D}_{\mathbb{Z}^m, \chi}$ the (centered) discrete Gaussian distribution over \mathbb{Z}^m with parameter χ , i.e., the distribution over \mathbb{Z}^m where for all \mathbf{x} , $\Pr[\mathbf{x}] \propto e^{-\pi \cdot (x_1^2 + \dots + x_m^2) / \chi^2}$.

2.1 Pseudorandom Functions

A pseudorandom function (PRF) is a family of functions $\{F(\mathbf{k}, \cdot) : \{0, 1\}^{m(\lambda)} \rightarrow \{0, 1\}^{\ell(\lambda)}\}_{\lambda \in \mathbb{N}, \mathbf{k} \in \{0, 1\}^\lambda}$ such that:

- *efficiency*: one can compute $F(\mathbf{k}, x)$ in $\text{poly}(\lambda)$ -time given x and \mathbf{k} ,
- *security*: for any PPT adversary \mathcal{A} let

$$\text{Adv}_{\mathcal{A}, F}^{\text{PRF}}(\lambda) := \left| \Pr \left[\mathcal{A}^{F(\mathbf{k}, \cdot)}(1^\lambda) = 1 \right] - \Pr \left[\mathcal{A}^{R(\cdot)}(1^\lambda) = 1 \right] \right|,$$

where $\mathbf{k} \leftarrow \{0, 1\}^\lambda$ and $R \leftarrow \mathcal{F}(\{0, 1\}^{m(\lambda)} \rightarrow \{0, 1\}^{\ell(\lambda)})$, with $\mathcal{F}(\{0, 1\}^{m(\lambda)} \rightarrow \{0, 1\}^{\ell(\lambda)})$ denoting the set of all functions mapping $m(\lambda)$ bits to $\ell(\lambda)$ bits. A PRF F is secure if for all PPT adversary \mathcal{A} , the advantage $\text{Adv}_{\mathcal{A}, F}^{\text{PRF}}(\lambda)$ is a negligible function in λ .

2.2 Attribute-Based Encryption

Syntax. A key policy attribute-based encryption (KP-ABE) scheme Π for some class \mathcal{F} consists of four algorithms:

- $\text{Setup}(1^\lambda, \mathcal{F}) \rightarrow (\text{mpk}, \text{msk})$. The setup algorithm gets as input the security parameter 1^λ and class description \mathcal{F} . It outputs the master public key mpk and the master secret key msk .
- $\text{Enc}(\text{mpk}, \mathbf{x}, \boldsymbol{\mu}) \rightarrow \text{ct}_{\mathbf{x}}$. The encryption algorithm gets as input mpk , an input \mathbf{x} and a message $\boldsymbol{\mu} \in \{0, 1\}^\lambda$. It outputs a ciphertext $\text{ct}_{\mathbf{x}}$. Note that \mathbf{x} is public given $\text{ct}_{\mathbf{x}}$.
- $\text{KeyGen}(\text{mpk}, \text{msk}, f) \rightarrow \text{sk}_f$. The key generation algorithm gets as input mpk, msk and $f \in \mathcal{F}$. It outputs a secret key sk_f . Note that f is public given sk_f .
- $\text{Dec}(\text{mpk}, \text{sk}_f, f, \text{ct}_{\mathbf{x}}, \mathbf{x}) \rightarrow \boldsymbol{\mu}$. The decryption algorithm gets as input sk_f and $\text{ct}_{\mathbf{x}}$ along with mpk . It outputs a message $\boldsymbol{\mu}$.

Correctness. For all $\ell \in \mathbb{N}$, inputs $\mathbf{x} \in \{0, 1\}^\ell$, functions $f : \{0, 1\}^\ell \rightarrow \{0, 1\}$ with $f(\mathbf{x}) = 0$, and all $\boldsymbol{\mu} \in \{0, 1\}^\lambda$, we require

$$\Pr \left[\begin{array}{l} (\text{mpk}, \text{msk}) \leftarrow \text{Setup}(1^\lambda, \mathcal{F}) \\ \text{Dec}(\text{mpk}, \text{sk}_f, \text{ct}_{\mathbf{x}}) = \boldsymbol{\mu} : \text{sk}_f \leftarrow \text{KeyGen}(\text{mpk}, \text{msk}, f) \\ \text{ct}_{\mathbf{x}} \leftarrow \text{Enc}(\text{mpk}, \mathbf{x}, \boldsymbol{\mu}) \end{array} \right] \geq 1 - \text{negl}(\lambda).$$

Security Definition. For a stateful adversary \mathcal{A} , we define the advantage function

$$\text{Adv}_{\mathcal{A}, \Pi}^{\text{ABE}}(\lambda) := \Pr \left[\begin{array}{l} (\text{mpk}, \text{msk}) \leftarrow \text{Setup}(1^\lambda, \mathcal{F}) \\ \mathbf{x}^* \leftarrow \mathcal{A}(1^\lambda, \text{mpk}) \\ (\boldsymbol{\mu}_0, \boldsymbol{\mu}_1) \leftarrow \mathcal{A}^{\text{KeyGen}(\text{mpk}, \text{msk}, \cdot)}(\text{mpk}) \\ b \leftarrow \{0, 1\}; \text{ct}_{\mathbf{x}^*} \leftarrow \text{Enc}(\text{mpk}, \mathbf{x}^*, \boldsymbol{\mu}_b) \\ b' \leftarrow \mathcal{A}^{\text{KeyGen}(\text{mpk}, \text{msk}, \cdot)}(\text{ct}_{\mathbf{x}^*}) \end{array} \right] - \frac{1}{2},$$

with the restriction that all queries $f : \{0, 1\}^\ell \rightarrow \{0, 1\}$ that \mathcal{A} sent to $\text{KeyGen}(\text{mpk}, \text{msk}, \cdot)$ satisfy either $\ell \neq |\mathbf{x}^*|$ or $f(\mathbf{x}^*) = 1$. An ABE scheme Π is *semi-adaptively secure* if for all PPT adversaries \mathcal{A} , the advantage $\text{Adv}_{\mathcal{A}, \Pi}^{\text{ABE}}(\lambda)$ is a negligible function in λ .

2.3 Lattices Background

Learning with Errors. Given $n, m, q, \chi_e \in \mathbb{N}$, the $\text{LWE}_{n, m, q, \chi_e}$ assumption states that

$$(\mathbf{A}, \mathbf{s} \cdot \mathbf{A} + \mathbf{e}) \approx_c (\mathbf{A}, \mathbf{c}),$$

where

$$\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z}^m, \chi_e}, \mathbf{c} \leftarrow \mathbb{Z}_q^m.$$

Leftover Hash Lemma and Generalizations A result that we will use is the so-called leftover hash lemma (LHL) [HILL99], which states that for $m \geq (n+1) \cdot \log q + 2 \cdot \lambda$ the distribution of $(\mathbf{A}, \mathbf{u} = \mathbf{A} \cdot \mathbf{x})$ for uniform and independent $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}$ and $\mathbf{x} \leftarrow \{1, -1\}^m$ is statistically indistinguishable from uniformly random.

Lemma 1 (Generalized Leftover Hash Lemma [DRS04, ABB10]). *Suppose that $m > (n + 1) \log q + \omega(\log n)$ and that $q > 2$ is prime. Let \mathbf{R} be an $m \times k$ matrix chosen uniformly in $\{1, -1\}^{m \times k} \pmod q$ where $k = k(n)$ is polynomial in n . Let \mathbf{A} and \mathbf{B} be matrices chosen uniformly in $\mathbb{Z}_q^{n \times m}$ and $\mathbb{Z}_q^{n \times k}$ respectively. Then, for all vectors \mathbf{w} in \mathbb{Z}_q^m , the distribution $(\mathbf{A}, \mathbf{A} \cdot \mathbf{R}, \mathbf{w}^\top \cdot \mathbf{R})$ is statistically close to the distribution $(\mathbf{A}, \mathbf{B}, \mathbf{w}^\top \cdot \mathbf{R})$.*

Trapdoor and Preimage Sampling. Let $n, q \in \mathbb{Z}$,

$$\mathbf{g}_q = (1, 2, 4, \dots, 2^{\lceil \log q \rceil - 1}) \in \mathbb{Z}^{\lceil \log q \rceil}.$$

The gadget matrix $\mathbf{G}_{n,q}$ is defined as the diagonal concatenation of \mathbf{g}_q n times. Formally, $\mathbf{G}_{n,q} = \mathbf{g}_q \otimes \mathbf{I}_n \in \mathbb{Z}^{n \times n \cdot \lceil \log q \rceil}$. For any $t \in \mathbb{Z}$, the function $\mathbf{G}_{n,q}^{-1} : \mathbb{Z}_q^{n \times t} \rightarrow \{0, 1\}^{n \cdot \lceil \log q \rceil \times t}$ expands each entry $a \in \mathbb{Z}_q$ of the input matrix into a column of size $\lceil \log q \rceil$ consisting of the bit-representation of a . For any matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times t}$ it holds that $\mathbf{G}_{n,q} \cdot \mathbf{G}_{n,q}^{-1}(\mathbf{A}) = \mathbf{A} \bmod q$. We refer to the gadget matrix simply as \mathbf{G} when parameters n and q are clear from the context.

Let $n, m, q \in \mathbb{N}$ and consider a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$. For all $\mathbf{V} \in \mathbb{Z}_q^{n \times m'}$ we let $\mathbf{A}^{-1}(\mathbf{V}, \tau)$ denote the random variable whose distribution is the discrete Gaussian $\mathcal{D}_{\mathbb{Z}^{m \times m'}, \tau}$ conditioned on $\mathbf{A} \cdot \mathbf{A}^{-1}(\mathbf{V}, \tau) = \mathbf{V} \bmod q$. If $\mathbf{Y} \leftarrow \mathbf{A}^{-1}(\mathbf{V}, \tau)$ then $\|\mathbf{Y}\| \leq k \cdot \tau \cdot \sqrt{m \cdot m'}$ with probability at least $1 - e^{-\Omega(k^2)}$. A matrix $\mathbf{T} \in \mathbb{Z}^{m \times m}$ such that $\mathbf{A} \cdot \mathbf{T} = \mathbf{H} \cdot \mathbf{G}$, for some invertible matrix $\mathbf{H} \in \mathbb{Z}_q^{n \times n}$ is called a τ -trapdoor for \mathbf{A} , for $\tau \geq 2 \cdot m \cdot \sqrt{n \cdot \log q} \cdot \|\mathbf{T}\|$. The following properties have been established in a long sequence of works.

Lemma 2 (Trapdoor Generation and Sampling [Ajt96, GPV08, MP12]).

There exists a pair of probabilistic polynomial-time algorithms:

- $\text{TrapGen}(1^n, 1^m, q)$ that for all $m \geq m_0 = m_0(n, q) = O(n \log q)$, outputs $(\mathbf{A}, \mathbf{T}_\mathbf{A})$ s.t. $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ is within statistical distance 2^{-n} from uniform and $\mathbf{T}_\mathbf{A}$ is a τ -trapdoor for \mathbf{A} where $\tau = O(\sqrt{n} \cdot \log q \cdot \log n)$.
- $\text{SamplePre}(\mathbf{A}, \mathbf{T}, \mathbf{V}, \tau)$ that given \mathbf{A} and any τ -trapdoor \mathbf{T} of \mathbf{A} , outputs a sample from $\mathbf{A}^{-1}(\mathbf{V}, \tau)$.

Moreover

1. for $\mathbf{x} \leftarrow \mathcal{D}_{\mathbb{Z}^m, \tau}$, the marginal distribution of $\mathbf{y} = \mathbf{A} \cdot \mathbf{x} \in \mathbb{Z}_q^n$ is uniform (up to $\text{negl}(n)$ statistical distance), and the conditional distribution of \mathbf{x} given \mathbf{y} is $\mathbf{A}^{-1}(\mathbf{y}, \tau)$.

Lemma 3 (Trapdoor Extension [ABB10, CHKP10]). *Given $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, with a τ -trapdoor \mathbf{T} , it is efficient to sample from $[\mathbf{A}|\mathbf{B}]^{-1}(\cdot, \tau)$ for all $\mathbf{B} \in \mathbb{Z}_q^{n \times k}$. Moreover, for any $\mathbf{V} \in \mathbb{Z}_q^{n \times m'}$, the following two distributions are statistically close*

- $\mathbf{U} \in \mathbb{Z}^{m+k \times m'}$, where $\mathbf{U} \leftarrow [\mathbf{A}|\mathbf{B}]^{-1}(\mathbf{V}, \tau)$,
- $\begin{bmatrix} \mathbf{U}_0 \\ \mathbf{U}_1 \end{bmatrix} \in \mathbb{Z}^{m+k \times m'}$, where $\mathbf{U}_1 \leftarrow \mathcal{D}_{\mathbb{Z}^{k \times m'}, \tau}$ and $\mathbf{U}_0 \leftarrow \mathbf{A}^{-1}(\mathbf{V} - \mathbf{B}\mathbf{U}_1, \tau)$.

Another related result that we will use is the so-called leftover hash lemma (LHL) [HILL99], which states that for $m \geq (n+1) \cdot \log q + 2 \cdot \lambda$ the distribution of $(\mathbf{A}, \mathbf{u} = \mathbf{A} \cdot \mathbf{x})$ for uniform and independent $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}$ and $\mathbf{x} \leftarrow \{0, 1\}^m$ is statistically indistinguishable from uniformly random.

Homomorphic Computation on Matrices. We recall basic homomorphic computation on matrices used in BGGHNSVV14:

Theorem 1 ([BGG⁺14, GSW13]). *There exist efficient deterministic algorithms EvalF and EvalFX such that for all $n, q, \ell \in \mathbb{N}$, and for any sequence of matrices $\mathbf{A} = (\mathbf{A}_1, \dots, \mathbf{A}_\ell) \in (\mathbb{Z}^{n \times n \cdot \lceil \log q \rceil})^\ell$, for any depth- d Boolean circuit $f : \{0, 1\}^\ell \rightarrow \{0, 1\}$ and for every $\mathbf{x} = (x_1, \dots, x_\ell) \in \{0, 1\}^\ell$, the following properties hold.*

- The outputs $\mathbf{A}_f = \text{EvalF}(\mathbf{A}, f)$ and $\mathbf{H}_{\mathbf{A}, f, \mathbf{x}} = \text{EvalFX}(\mathbf{A}, f, \mathbf{x})$ are matrices in $\mathbb{Z}_q^{n \times (n \cdot \lceil \log q \rceil)}$ and $\mathbb{Z}^{(\ell \cdot n \cdot \lceil \log q \rceil) \times (n \cdot \lceil \log q \rceil)}$,
- It holds that $\|\mathbf{H}_{\mathbf{A}, f, \mathbf{x}}\| \leq (n \cdot \log q)^{O(d)}$,
- It holds that

$$(\mathbf{A} - \mathbf{x} \otimes \mathbf{G}_{n, q}) \cdot \mathbf{H}_{\mathbf{A}, f, \mathbf{x}} = \mathbf{A}_f - f(\mathbf{x}) \cdot \mathbf{G}_{n, q} \pmod{q}.$$

For a proof of this theorem, we refer the reader to [BCTW16, Fact 3.4].

3 Unbounded ABE for Circuits

We refer to Sect. 1.2 for an overview of the scheme and the security proof.

Construction. Let the ABE $\Pi = (\text{Setup}, \text{Enc}, \text{KeyGen}, \text{Dec})$ for the family \mathcal{F}_d of circuits of depth d , over ℓ -bit inputs for any $\ell \in \mathbb{N}$, be defined as follows:

- $\text{Setup}(1^\lambda, 1^d)$: Sample

$$\begin{aligned} (\mathbf{A}_0, \mathbf{T}_{\mathbf{A}_0}) &\leftarrow \text{TrapGen}(1^n, 1^m, q), (\mathbf{B}_0, \mathbf{T}_{\mathbf{B}_0}) \leftarrow \text{TrapGen}(1^n, 1^m, q), \\ \mathbf{B}_1, \mathbf{W}, \mathbf{V} &\leftarrow \mathbb{Z}_p^{n \times m}, \mathbf{D} \leftarrow \mathbb{Z}_q^{n \times \lambda}, \\ \mathbf{k} &\leftarrow \{0, 1\}^\lambda. \end{aligned}$$

where q is prime¹. Set $\text{mpk} = (\mathbf{A}_0, \mathbf{B}_0, \mathbf{B}_1, \mathbf{W}, \mathbf{V}, \mathbf{D})$, and $\text{msk} = (\mathbf{T}_{\mathbf{A}_0}, \mathbf{T}_{\mathbf{B}_0}, \mathbf{k})$. Return (mpk, msk) .

- $\text{Enc}(\text{mpk}, \mathbf{x} \in \{0, 1\}^\ell, \boldsymbol{\mu} \in \{0, 1\}^\lambda)$: Let $\ell = |\mathbf{x}|$. Sample

$$\begin{aligned} \mathbf{s} &\leftarrow \mathbb{Z}_q^n, \mathbf{e}_0 \leftarrow \mathcal{D}_{\mathbb{Z}^m, \chi}, \mathbf{e}_4 \leftarrow \mathcal{D}_{\mathbb{Z}^\lambda, \chi}, \mathbf{e}_5 \leftarrow \mathcal{D}_{\mathbb{Z}^m, \chi'}, \\ \mathbf{s}_j &\leftarrow \mathbb{Z}_q^n, \mathbf{e}_{1,j} \leftarrow \mathcal{D}_{\mathbb{Z}^m, \chi}, \mathbf{e}_{2,j} \leftarrow \mathcal{D}_{\mathbb{Z}^m, \chi'}, \mathbf{e}_{3,j} \leftarrow \mathcal{D}_{\mathbb{Z}^m, \chi''} \quad \text{for all } j \in [\ell]. \end{aligned}$$

Compute

$$\begin{aligned} \mathbf{c}_0 &:= \mathbf{s} \cdot \mathbf{A}_0 + \mathbf{e}_0 \pmod{q}, \\ \mathbf{c}_{1,j} &:= \mathbf{s}_j \cdot \mathbf{B}_0 + \mathbf{e}_{1,j} \pmod{q} \quad \text{for all } j \in [\ell], \\ \mathbf{c}_{2,j} &:= \mathbf{s}_j \cdot (\mathbf{W} + j \cdot \mathbf{G}) + \mathbf{s} \cdot \mathbf{G} + \mathbf{e}_{2,j} \pmod{q} \quad \text{for all } j \in [\ell], \\ \mathbf{c}_{3,j} &:= \mathbf{s}_j \cdot \mathbf{V} + x_j \cdot \mathbf{s} \cdot \mathbf{G} + \mathbf{e}_{3,j} \pmod{q} \quad \text{for all } j \in [\ell], \\ \mathbf{c}_4 &:= \mathbf{s} \cdot \mathbf{D} + \boldsymbol{\mu} \cdot \lfloor q/2 \rfloor + \mathbf{e}_4 \pmod{q}, \\ \mathbf{c}_5 &:= \mathbf{s} \cdot (\mathbf{B}_1 - \ell \cdot \mathbf{G}) + \mathbf{e}_5 \pmod{q}. \end{aligned}$$

Output $\text{ct}_{\mathbf{x}} := (\mathbf{c}_0, \{\mathbf{c}_{1,j}\}_{j \in [\ell]}, \{\mathbf{c}_{2,j}\}_{j \in [\ell]}, \{\mathbf{c}_{3,j}\}_{j \in [\ell]}, \mathbf{c}_4, \mathbf{c}_5)$.

¹ We can also adapt the construction to support non-prime moduli using techniques from [MP12].

– $\text{KeyGen}(\text{mpk}, \text{msk}, f)$: Let ℓ equal the size of f 's inputs. For all $j \in [\ell]$, sample

$$\mathbf{K}_j \leftarrow \text{SamplePre}([\mathbf{B}_0 | \mathbf{W} + j \cdot \mathbf{G}], [\mathbf{T}_{\mathbf{B}_0} \mathbf{0}_{m \times m}], \mathbf{V}, \tau_1; \mathbf{F}(k, j)).$$

Parse

$$\mathbf{K}_j = \begin{bmatrix} \mathbf{Z}_j \\ \mathbf{R}_j \end{bmatrix},$$

and let $\mathbf{A}_j := \mathbf{G} \cdot \mathbf{R}_j \bmod q$. Let $\mathbf{A} := [\mathbf{A}_1 | \dots | \mathbf{A}_\ell]$ and $\mathbf{A}_f = \text{EvalF}(\mathbf{A}, f)$.

Sample

$$\mathbf{K}_f \leftarrow \text{SamplePre}([\mathbf{A}_0 | \mathbf{A}_f | \mathbf{B}_1 - \ell \cdot \mathbf{G}], \begin{bmatrix} \mathbf{T}_{\mathbf{A}_0} \\ \mathbf{0}_{m \times m} \\ \mathbf{0}_{m \times m} \end{bmatrix}, \mathbf{D}, \tau_2).$$

Output $\text{sk}_f := (\{\mathbf{K}_j\}_{j \in [\ell]}, \mathbf{K}_f)$. Here, \mathbf{K}_f is the private component, and $\{\mathbf{K}_j\}$ is the public component and can be reused over all functions of input length at most ℓ .

– $\text{Dec}(\text{mpk}, \text{sk}_f, f, \text{ct}_{\mathbf{x}}, \mathbf{x})$: Let $\ell = |\mathbf{x}|$. Parse $\text{ct}_{\mathbf{x}} = (\mathbf{c}_0, \{\mathbf{c}_{1,j}\}_{j \in [\ell]}, \{\mathbf{c}_{2,j}\}_{j \in [\ell]}, \{\mathbf{c}_{3,j}\}_{j \in [\ell]}, \mathbf{c}_4, \mathbf{c}_5)$, $\text{sk}_f = (\{\mathbf{K}_j\}_{j \in [\ell]}, \mathbf{K}_f)$, and $\mathbf{K}_j = \begin{bmatrix} \mathbf{Z}_j \\ \mathbf{R}_j \end{bmatrix}$ for all $j \in [\ell]$. Let $\mathbf{A}_j = \mathbf{G} \cdot \mathbf{R}_j$ and $\mathbf{A} = [\mathbf{A}_1 \dots \mathbf{A}_\ell]$. Compute $\mathbf{H}_{\mathbf{A}, f, \mathbf{x}} = \text{EvalFX}(\mathbf{A}, f, \mathbf{x})$. For each $j \in [\lambda]$, check if the j -th entry of

$$\mathbf{c}_4 - \left[\mathbf{c}_0 \left| \left(\left[[\mathbf{c}_{1,1} \ \mathbf{c}_{2,1}] \cdot \mathbf{K}_1 \mid \dots \mid [\mathbf{c}_{1,\ell} \ \mathbf{c}_{2,\ell}] \cdot \mathbf{K}_\ell \right] - [\mathbf{c}_{3,1} \ \dots \ \mathbf{c}_{3,\ell}] \right) \cdot \mathbf{H}_{\mathbf{A}, f, \mathbf{x}} \right| \mathbf{c}_5 \right] \cdot \mathbf{K}_f$$

is $q/4$ -close to $q/2$. If so, set $\mu_j := 1$. Else, $\mu_j := 0$. Return $\boldsymbol{\mu}$.

Parameters. We have 3 gaussian parameters:

$$\underbrace{\approx \|\mathbf{e}_0\|, \|\mathbf{e}_{1,j}\|, \|\mathbf{e}_4\|}_{\chi} \leq \underbrace{\approx \|\mathbf{e}_{2,j}\|, \|\mathbf{e}_5\|}_{\chi'} \leq \underbrace{\approx \|\mathbf{e}_{3,j}\|}_{\chi''}.$$

The parameters requirements can be compactly specified as:

$m \geq O(n \log q)$	<i>trapdoor generation (Lemma 2)</i>
$2^{n^\delta} \geq q/\chi_0, \quad \chi \geq O(n + \lambda)$	LWE n, χ_{s_0}, q hardness ($\text{H}_{3,i,6} \approx_s \text{H}_{3,i,7}, \text{H}_2 \approx_c \text{H}_3, \text{H}_7 \approx_c \text{H}_8$)
$\chi' \geq \chi \cdot \text{poly}(\lambda, m) \cdot \lambda^{\omega(1)}$	noise flooding ($\text{H}_{3,i,5} \approx_s \text{H}_{3,i,6}, \text{H}_6 \approx_s \text{H}_7$)
$\chi'' \geq \chi' \cdot \tau_1 \cdot \text{poly}(\lambda, m) \cdot \lambda^{\omega(1)}$	noise flooding ($\text{H}_{3,i,4} \approx_s \text{H}_{3,i,5}, \text{H}_6 \approx_s \text{H}_7$)
$m \geq (n + 1) \cdot \log q + \omega(\log n) + 2\lambda$	(G)LHL ($\text{H}_{3,i,0} \approx_s \text{H}_{3,i,1}, \text{H}_{3,i,7} \approx_s \text{H}_{3,i,8}$)
$\tau_1 \geq O(m^2)$	trapdoor generation ($\text{H}_{3,i,2} \approx_s \text{H}_{3,i,3}$)
$\tau_2 \geq \lambda^{\omega(1)} \cdot m^3 \cdot B$	trapdoor generation ($\text{H}_4 \approx_s \text{H}_5$)
$q \geq \text{poly}(\lambda, m, \ell) \cdot \tau_2 \cdot \tau_1 \cdot B \cdot (\chi + \chi' + \chi'')$	correctness (Theorem 2)

We bound the adversarially chosen parameter d, ℓ by $\lambda^{\omega(1)}$. Taking $\lambda_1 = \lambda^{\omega(1)}$, and additionally bounding the $\text{poly}(\lambda, m, \ell)$ terms by λ_1 , we can set

$$\begin{aligned} m &= O(n \log q), & \chi' &= \lambda_1^3, & \chi'' &= \lambda_1^6, \\ \chi &= \lambda_1, & \tau_2 &= B \cdot \lambda_1^2 = \lambda_1^{O(d)}, \\ \tau_1 &= \lambda_1, & n &= O(\log B + \log \lambda_1)^{1/\delta} = O(d \cdot \log \lambda_1)^{1/\delta}, \\ q &= B \cdot \tau_2 \cdot \lambda_1^8 = \lambda_1^{O(d)}, \end{aligned} \tag{3}$$

where in the last two lines, we use $B \leq m^{O(d)} \leq \lambda_1^{O(d)}$.

Efficiency. Our ABE scheme achieves

$$|\text{mpk}| = O((n \cdot \log q)^2), \quad |\text{ct}| = O(\ell \cdot n \cdot (\log q)^2), \quad |\text{sk}| = O(\ell \cdot (n \cdot \log q)^2 \cdot \log \tau_1 + \lambda \cdot n \cdot \log q \cdot \log \tau_2).$$

This yields the following parameter sizes (in bits) for our ABE scheme:

$$|\text{mpk}| = O_\lambda(d^{2+2/\delta}), \quad |\text{ct}| = O_\lambda(\ell \cdot d^{2+1/\delta}), \quad |\text{sk}| = O_\lambda(\ell \cdot d^{2+2/\delta}).$$

where $O_\lambda(\cdot)$ hides factors polynomial in λ (bounded by λ^4). Here, we use $n = O(d^{1/\delta} \cdot \lambda)$, $\log q = O(d \cdot \lambda)$, where we do optimize on the dependency on d but not on λ .

Comparison with BGGHNSVV14 ABE. To compare concrete efficiency of our construction against the BGGHNSVV14 ABE, let n, m, q denote the parameters in our scheme and n_0, m_0, q_0 those in BGGHNSVV14. Since $q_0 \geq B$, we can set

$$q = q_0 \cdot \lambda^{\omega(1)}.$$

This implies that we have

$$\log q = (1 + o(1)) \cdot \log q_0, \quad n = (1 + o(1)) \cdot n_0, \quad \text{and} \quad m = (1 + o(1)) \cdot m_0.$$

In particular, n, m , and $\log q$ factors are essentially the same in both schemes. Therefore, to compare concrete efficiency with BGGHNSVV14 ABE, we can compare the number of field (i.e., \mathbb{Z}_q) elements and operations.

- Our ciphertext size is $(3\ell + 2) \cdot m + \lambda$ elements in \mathbb{Z}_q , whereas that in BGGHNSVV14 is $(\ell + 1) \cdot m + \lambda$ elements.
- Encryption requires $(3\ell + 2) \cdot m + \lambda$ vector-vector products over \mathbb{Z}_q^n and sampling $(3\ell + 2) \cdot m + \lambda$ gaussians over \mathbb{Z}_q , whereas that in BGGHNSVV14 requires $(\ell + 1) \cdot m + \lambda$ vector-vector products and $(\ell + 1) \cdot m + \lambda$ gaussians.
- Our secret key contains a private component with $m\lambda$ \mathbb{Z}_q -elements, and a public component with $m\ell n$ \mathbb{Z}_q -elements, whereas that in BGGHNSVV14 is $m\lambda$ elements.
- Decryption in both schemes are dominated by $s \cdot \text{poly}(\lambda)$ time to compute $\mathbf{H}_{\mathbf{A}, f, \mathbf{x}}$, with an additive $\ell \cdot \text{poly}(\lambda)$ overhead in our scheme.

Notice that this also applies to GKW16 since it uses the BGGHNSVV14 scheme as the underlying building block.

Theorem 2 (Correctness). *Let Π be the ABE construction described in Sect. 3, with parameters as in Eq. 3. Then Π is correct.*

Proof Fix \mathbf{x} , f such that $f(\mathbf{x}) = 0$. The bulk of the proof lies in showing that

$$\begin{aligned} \left[\mathbf{c}_0 \mid \left(\left[[\mathbf{c}_{1,1} \ \mathbf{c}_{2,1}] \cdot \mathbf{K}_1 \mid \dots \mid [\mathbf{c}_{1,\ell} \ \mathbf{c}_{2,\ell}] \cdot \mathbf{K}_\ell \right] - [\mathbf{c}_{3,1} \ \dots \ \mathbf{c}_{3,\ell}] \right) \cdot \mathbf{H}_{\mathbf{A},f,\mathbf{x}} \mid \mathbf{c}_5 \right] \\ = \mathbf{s} \cdot [\mathbf{A}_0 \mid \mathbf{A}_f \mid \mathbf{B}_1 - \ell \cdot \mathbf{G}] + \mathbf{e}'_{f,\mathbf{x}} \bmod q \end{aligned} \quad (4)$$

where $\|\mathbf{e}'_{f,\mathbf{x}}\|$ is small. Correctness then follows readily from the fact that

$$\mathbf{c}_4 - (\mathbf{s} \cdot [\mathbf{A}_0 \mid \mathbf{A}_f \mid \mathbf{B}_1 - \ell \cdot \mathbf{G}] + \mathbf{e}'_{f,\mathbf{x}}) \cdot \mathbf{K}_f = \boldsymbol{\mu} \cdot \lfloor q/2 \rfloor + \mathbf{e}_4 - \mathbf{e}'_{f,\mathbf{x}} \cdot \mathbf{K}_f \bmod q.$$

To prove Eq. (4):

– First, for any $j \in [\ell]$, we have

$$\begin{aligned} [\mathbf{c}_{1,j} \ \mathbf{c}_{2,j}] \cdot \mathbf{K}_j &= (\mathbf{s}_j \cdot [\mathbf{B}_0 \mid \mathbf{W} + j \cdot \mathbf{G}] + \mathbf{s} \cdot [\mathbf{0}_{n \times m} \mid \mathbf{G}] + [\mathbf{e}_{1,j} \mid \mathbf{e}_{2,j}]) \cdot \mathbf{K}_j \\ &= \mathbf{s}_j \cdot [\mathbf{B}_0 \mid \mathbf{W} + j \cdot \mathbf{G}] \cdot \mathbf{K}_j + \mathbf{s} \cdot [\mathbf{0}_{n \times m} \mid \mathbf{G}] \cdot \mathbf{K}_j + \boxed{[\mathbf{e}_{1,j} \mid \mathbf{e}_{2,j}] \cdot \mathbf{K}_j} \\ &\approx \mathbf{s}_j \cdot \mathbf{V} + \mathbf{s} \cdot \mathbf{A}_j \bmod q. \end{aligned}$$

– Further, we have

$$\begin{aligned} \mathbf{s}_j \cdot \mathbf{V} + \mathbf{s} \cdot \mathbf{A}_j - \mathbf{c}_{3,j} &= \mathbf{s}_j \cdot \mathbf{V} + \mathbf{s} \cdot \mathbf{A}_j - (\mathbf{s}_j \cdot \mathbf{V} + x_j \cdot \mathbf{s} \cdot \mathbf{G} + \boxed{\mathbf{e}_{3,j}}) \\ &\approx \mathbf{s} \cdot (\mathbf{A}_j - x_j \cdot \mathbf{G}) \bmod q. \end{aligned}$$

– We deduce that

$$[[\mathbf{c}_{1,1} \ \mathbf{c}_{2,1}] \cdot \mathbf{K}_1 \mid \dots \mid [\mathbf{c}_{1,\ell} \ \mathbf{c}_{2,\ell}] \cdot \mathbf{K}_\ell] - [\mathbf{c}_{3,1} \ \dots \ \mathbf{c}_{3,\ell}] = \mathbf{s} \cdot (\mathbf{A} - \mathbf{x} \otimes \mathbf{G}) + \mathbf{e}_x \bmod q,$$

where $\mathbf{A} := [\mathbf{A}_1 \mid \dots \mid \mathbf{A}_\ell]$ and $\mathbf{e}_x := [[\mathbf{e}_{1,1} \ \mathbf{e}_{2,1}] \cdot \mathbf{K}_1 \mid \dots \mid [\mathbf{e}_{1,\ell} \ \mathbf{e}_{2,\ell}] \cdot \mathbf{K}_\ell] - [\mathbf{e}_{3,1} \mid \dots \mid \mathbf{e}_{3,\ell}]$.

– Using the key equation

$$(\mathbf{A} - \mathbf{x} \otimes \mathbf{G}) \cdot \mathbf{H}_{\mathbf{A},f,\mathbf{x}} = \mathbf{A}_f \bmod q,$$

as $f(\mathbf{x}) = 0$, we have

$$\begin{aligned} [[\mathbf{c}_{1,1} \ \mathbf{c}_{2,1}] \cdot \mathbf{K}_1 \mid \dots \mid [\mathbf{c}_{1,\ell} \ \mathbf{c}_{2,\ell}] \cdot \mathbf{K}_\ell] \cdot \mathbf{H}_{\mathbf{A},f,\mathbf{x}} &= \mathbf{s} \cdot (\mathbf{A} - \mathbf{x} \otimes \mathbf{G}) + \mathbf{e}_x \cdot \mathbf{H}_{\mathbf{A},f,\mathbf{x}} \\ &= \mathbf{s} \cdot \mathbf{A}_f + \boxed{\mathbf{e}_x \cdot \mathbf{H}_{\mathbf{A},f,\mathbf{x}}} \\ &\approx \mathbf{s} \cdot \mathbf{A}_f \bmod q. \end{aligned}$$

– Putting everything together, we obtain Eq. (4) with

$$\begin{aligned} \mathbf{e}'_{f,\mathbf{x}} &= [\mathbf{e}_0 \mid \mathbf{e}_x \cdot \mathbf{H}_{\mathbf{A},f,\mathbf{x}} \mid \mathbf{e}_5] \\ &= [\mathbf{e}_0 \mid ([[\mathbf{e}_{1,1} \ \mathbf{e}_{2,1}] \cdot \mathbf{K}_1 \mid \dots \mid [\mathbf{e}_{1,\ell} \ \mathbf{e}_{2,\ell}] \cdot \mathbf{K}_\ell] - [\mathbf{e}_{3,1} \mid \dots \mid \mathbf{e}_{3,\ell}]) \cdot \mathbf{H}_{\mathbf{A},f,\mathbf{x}} \mid \mathbf{e}_5], \end{aligned}$$

where

$$\begin{aligned} \|\mathbf{e}'_{f,\mathbf{x}}\| &\leq \lambda \cdot \chi \\ &\quad + \lambda^2 \cdot 2 \cdot \ell \cdot m^2 \cdot (\chi + \chi' + \chi'') \cdot \tau_1 \cdot B \\ &\quad + \lambda \cdot \chi'. \end{aligned}$$

In particular, the norm of the final error term is, with overwhelming probability in λ , bounded by

$$\begin{aligned} \|\mathbf{e}_4\| + \|\mathbf{e}'_{f,\mathbf{x}} \cdot \mathbf{K}_f\| &\leq \lambda \cdot \chi \\ &\quad + \lambda \cdot 3 \cdot m \cdot \tau_2 \cdot \left(\lambda \cdot \chi \right. \\ &\quad \quad \quad \left. + \lambda^2 \cdot 2 \cdot \ell \cdot m^2 \cdot (\chi + \chi' + \chi'') \cdot \tau_1 \cdot B \right. \\ &\quad \quad \quad \left. + \lambda \cdot \chi' \right), \end{aligned}$$

where we have used that $\|\mathbf{K}_f\| \leq \lambda \cdot \tau_2$ and that $\mathbf{e}'_{f,\mathbf{x}}$ is a vector of length $3 \cdot m$. Since

$$q \geq \text{poly}(\lambda, m, \ell) \cdot \tau_2 \cdot \tau_1 \cdot B \cdot (\chi + \chi' + \chi'')$$

the theorem follows. □

Theorem 3 (Security). *Let Π be the KP-ABE construction described in Sect. 3, with parameters set as in Eq. (3), and F a PRF. Then, for any semi-adaptive adversary \mathcal{A} that runs in time $T = T(\lambda)$, there exists adversaries \mathcal{B}_0 , \mathcal{B}_1 and \mathcal{B}_2 against PRF-security, $\text{LWE}_{n,m,\chi,q}$, and $\text{LWE}_{n,m+\lambda,\chi,q}$ respectively, such that*

$$\text{Adv}_{\mathcal{A},\Pi}^{\text{sa-ABE}}(\lambda) \leq T \cdot \left(\text{Adv}_{\mathcal{B}_0,F}^{\text{PRF}}(\lambda) + \text{Adv}_{\mathcal{B}_1}^{\text{LWE}_{n,m,\chi,q}}(\lambda) + \text{Adv}_{\mathcal{B}_2}^{\text{LWE}_{n,m+\lambda,\chi,q}}(\lambda) + \text{negl}(\lambda) \right).$$

Proof Consider the following sequence of hybrids, summarized in Fig. 2 and Fig. 3. Let $\text{Adv}_i(\mathcal{A})$ denote the advantage of \mathcal{A} in hybrid H_i . Notice that we can bound the length ℓ of the input domain of any function f queried to the KeyGen oracle by T , i.e., an adversary \mathcal{A} running in time T will never obtain \mathbf{K}_j for $j > T$.

– H_0 : This is identical to the real security game. Therefore

$$\text{Adv}_{\mathcal{A},\Pi}^{\text{sa-ABE}}(\lambda) = \text{Adv}_0(\mathcal{A}).$$

– H_1 : This is identical to H_0 , except for the fact that the reduction guesses $|\mathbf{x}^*| = \ell^*$ before generating the public parameters. If the guess is not correct, the reduction aborts. Since \mathcal{A} runs in time T , one has that $\ell^* \leq T$, so the reduction can guess ℓ^* and incur a security loss of T . In other words, we have that

$$\text{Adv}_0(\mathcal{A}) \leq T \cdot \text{Adv}_1(\mathcal{A}).$$

Hybrid	mpk	ct	sk _f	justification
H ₀	$(A_0, T_{A_0}) \leftarrow \text{TrapGen}(1^n, 1^m, q)$ $(B_0, T_{B_0}) \leftarrow \text{TrapGen}(1^n, 1^m, q)$ $W \leftarrow \mathbb{Z}_q^{n \times m}$ $V \leftarrow \mathbb{Z}_q^{m \times m}$ $B_1 \leftarrow \mathbb{Z}_q^{n \times m}$ $D \leftarrow \mathbb{Z}_q^{m \times \lambda}$ $k \leftarrow \mathcal{K}$	$c_0 \approx s \cdot A_0$ $c_{1,j} \approx s_j \cdot B_0$ $c_{2,j} \approx s_j \cdot (W + j \cdot G) + s \cdot G$ $c_{3,j} \approx s_j \cdot V + x_j^* \cdot s \cdot G$ $c_4 \approx s \cdot D + \mu \cdot [q/2]$ $c_5 \approx s \cdot (B_1 - x^* \cdot G)$	T_{A_0}, T_{B_0}, k $K_j = \begin{bmatrix} Z_j \\ R_j \end{bmatrix} \leftarrow \text{SamplePre} \left(\begin{bmatrix} [B_0 W + j \cdot G], \\ T_{B_0} \\ \mathbf{0}_{m \times m} \end{bmatrix}, V, \tau_1; F(k, j) \right)$ $A_j = G \cdot R_j$ $K_f \leftarrow \text{SamplePre} \left([A_0 A_f B_1 - \ell \cdot G], \begin{bmatrix} T_{A_0} \\ \mathbf{0}_{m \times m} \end{bmatrix}, D, \tau_2 \right)$	
H ₁	↓	↓	↓	guess $ x^* = \ell^*$
H ₂	↓	↓	$K_j = \begin{bmatrix} Z_j \\ R_j \end{bmatrix} \leftarrow \text{SamplePre} \left(\begin{bmatrix} [B_0 W + j \cdot G], \\ T_{B_0} \\ \mathbf{0}_{m \times m} \end{bmatrix}, V, \tau_1; r_j \right)$	PRF security
H ₃	↓	$c_{1,j}, c_{2,j} \leftarrow \mathbb{Z}_q^{m \times m}$ $c_{3,j} \approx [c_{1,j} c_{2,j}] \cdot K_j - s \cdot (A_j - x_j^* \cdot G)$	↓	LWE _{n,m,x,q} (Fig. 3)
H ₄	↓	↓	$R_j \leftarrow \mathcal{D}_{\mathbb{Z}_q^{m \times m}, \tau_1}$ $Z_j = \text{SamplePre} \left(\begin{bmatrix} B_0, T_{B_0} \\ V - [W + j \cdot G] \cdot R_j, \tau_1; r_{j,1} \end{bmatrix} \right)$ $K_j = \begin{bmatrix} Z_j \\ R_j \end{bmatrix}$	Lemma 3
H ₅	↓	↓	$A_j \leftarrow \mathbb{Z}_q^{n \times m}$ $R_j = \text{SamplePre}(G, I, A_j, \tau_1; r_{j,2})$ $Z_j = \text{SamplePre} \left(\begin{bmatrix} B_0, T_{B_0} \\ V - [W + j \cdot G] \cdot R_j, \tau_1; r_{j,1} \end{bmatrix} \right)$ $K_j = \begin{bmatrix} Z_j \\ R_j \end{bmatrix}$	Lemma 2 (Item 1)
H ₆	$B_1 = A_0 \cdot U + \ell^* \cdot G$	↓	$A_j = A_0 \cdot U_j + x_j^* \cdot G$	LHL
H ₇	↓	↓	K_f using $\begin{bmatrix} [U_1 \dots U_{\ell^*}] \cdot H_{A,f,x^*} \\ I_m \\ \mathbf{0}_{m \times m} \\ -U \\ \mathbf{0}_{m \times m} \\ I_m \end{bmatrix}$ if $\ell = \ell^*$ if $\ell \neq \ell^*$	Lemma 2 SamplePre
H ₈	$A_0 \leftarrow \mathbb{Z}_q^{n \times m}$	↓	↓	Lemma 2 TrapGen
H ₉	↓	$c_{3,j} \approx [c_{1,j} c_{2,j}] \cdot K_j$ $-c_0 \cdot U_j$ $c_5 \approx c_0 \cdot U$	↓	noise flooding
H ₁₀	↓	$c_0 \leftarrow \mathbb{Z}_q^m, c_4 \leftarrow \mathbb{Z}_q^\lambda$	↓	LWE _{n,m+\lambda,x,q}

Fig. 2. Summary of our security proof. ↓ denotes the same as the previous hybrid. We omit the noise terms in H₀. Starting from H₆, the proof is essentially the same as the BGGHNSVV14 ABE.

- H₂: This is identical to H₁, except for the following modification to KeyGen:
 - for all $j \in [T]$, sample once and for all a random string $r_j \leftarrow \{0, 1\}^{\text{poly}(\lambda)}$,
 - use r_j as randomness to sample K_j , i.e.

$$K_j \leftarrow \text{SamplePre} \left([B_0|W + j \cdot G], \begin{bmatrix} T_{B_0} \\ \mathbf{0}_{m \times m} \end{bmatrix}, V, \tau_1; r_j \right).$$

To show that $H_1 \approx_c H_2$, we rely on the PRF security of F. The reduction works as follows:

- it samples A, B_0, B_1, W, V and D as in H₁,
- it obtains x^* from the adversary \mathcal{A} ,
- it answers KeyGen queries as in H₁ but using the output $\mathcal{O}(j)$ of its PRF oracle as randomness to sample K_j ,
- whenever the adversary \mathcal{A} produces (μ_0, μ_1) , it produces the challenge ciphertext ct_{x^*} as in H₁.

Observe that

- if $\mathcal{O}(\cdot) = F(k, \cdot)$ is pseudorandom function instance, the view of the adversary \mathcal{A} is identical to H₁;

Hybrid	mpk	ct	sk _j	justification
H _{3,i,0}	↓	↓	↓	H _{3,1,0} = H ₂ , H _{3,i,0} = H _{3,i-1,0} , i > 1
H _{3,i,1}	$\mathbf{W} = \mathbf{B}_0 \cdot \tilde{\mathbf{W}}_i - i \cdot \mathbf{G}$	↓	↓	LHL
H _{3,i,2}	$\mathbf{V} = [\mathbf{B}_0 \mid \mathbf{W} + i \cdot \mathbf{G}] \cdot \mathbf{K}_i$	↓	$\mathbf{Z}_i, \mathbf{R}_i \leftarrow \mathcal{D}_{\mathbb{Z}^{m \times m}, \tau_1}$ $\mathbf{K}_i = \begin{bmatrix} \mathbf{Z}_i \\ \mathbf{R}_i \end{bmatrix}$	Lemma 2 (Item 1)
H _{3,i,3}	↓	↓	$\mathbf{K}_j \leftarrow \text{SamplePre} \left(\begin{bmatrix} \mathbf{B}_0 \mathbf{W} + j \cdot \mathbf{G} \\ \tilde{\mathbf{W}}_j \\ -\mathbf{I}_m \end{bmatrix}, \mathbf{V}, \tau_1; \mathbf{r}_j \right), j \neq i$	Lemma 2 SamplePre
H _{3,i,4}	$\mathbf{B}_0 \leftarrow \mathbb{Z}_q^{n \times m}$	↓	↓	Lemma 2 TrapGen
H _{3,i,5}	↓	$\mathbf{c}_{3,i} = \begin{bmatrix} \mathbf{c}_{1,i} & \mathbf{c}_{2,i} \\ -\mathbf{s} \cdot (\mathbf{A}_i - \mathbf{x}_i^* \cdot \mathbf{G}) + \mathbf{e}_{3,i} \end{bmatrix}$	↓	noise flooding
H _{3,i,6}	↓	$\mathbf{c}_{2,i} = \mathbf{c}_{1,i} \cdot \tilde{\mathbf{W}}_i + i \cdot \mathbf{s} \cdot \mathbf{G} + \mathbf{e}_{2,i}$	↓	noise flooding
H _{3,i,7}	↓	$\mathbf{c}_{1,i} \leftarrow \mathbb{Z}_q^m$	↓	LWE _{n,m,x,q}
H _{3,i,8}	↓	$\mathbf{c}_{2,i} \leftarrow \mathbb{Z}_q^m$	↓	?? GLHL
H _{3,i,9}	same as H ₂	↓	same as H ₂	H _{3,i,4} ≈ _s H _{3,i,0}

Fig. 3. Summary for H₂ ≈_c H₃. The sequence of hybrid is repeated for all i ∈ [ℓ*]. That is, H₂ = H_{3,1,0} ≈ ⋯ ≈ H_{3,1,9} = H_{3,2,0} ≈ ⋯ ≈ H_{3,ℓ*,9} = H₃.

- if $\mathcal{O}(\cdot) = F(\cdot)$ is a truly random function instance, the view of \mathcal{A} is identical to H₂.

We conclude that

$$\text{Adv}_1(\mathcal{A}) \leq \text{Adv}_2(\mathcal{A}) + \text{Adv}_{\mathcal{B}_0, F}^{\text{PRF}}(\lambda).$$

and in particular, that H₁ ≈_c H₂.

For i ∈ [ℓ*]:

- H_{3,i,1}: This is the same as previous hybrid, except for the following modification to \mathbf{W} in mpk:
 - sample $\tilde{\mathbf{W}}_i \leftarrow \{1, -1\}^{m \times m}$,
 - set $\mathbf{W} := \mathbf{B}_0 \cdot \tilde{\mathbf{W}}_i - i \cdot \mathbf{G}$.

Since $\tilde{\mathbf{W}}_i$ is sampled uniformly and $m \geq (n + 1) \cdot \log q + 2 \cdot \lambda$, statistical indistinguishability of H_{3,i,1} from previous hybrid follows from the leftover hash lemma. Notice that, for all j ∈ [ℓ*], we can now rewrite

$$\mathbf{W} + j \cdot \mathbf{G} = \mathbf{B}_0 \cdot \tilde{\mathbf{W}}_i + (j - i) \cdot \mathbf{G}$$

- H_{3,i,2} This is the same as H_{3,i,1}, except for the following modification to \mathbf{V} in mpk and to \mathbf{K}_i in KeyGen queries:
 - Parse $\mathbf{r}_i = [\mathbf{r}_{i,1} \mid \mathbf{r}_{i,2}]$ and sample $\mathbf{Z}_i, \mathbf{R}_i \leftarrow \mathcal{D}_{\mathbb{Z}^{m \times m}, \tau_1}$ using as random coins $\mathbf{r}_{i,1}$ and $\mathbf{r}_{i,2}$ respectively,
 - set $\mathbf{K}_i := \begin{bmatrix} \mathbf{Z}_i \\ \mathbf{R}_i \end{bmatrix}$,
 - set $\mathbf{V} := [\mathbf{B}_0 \mid \mathbf{W} + i \cdot \mathbf{G}] \cdot \mathbf{K}_i$.

By the properties of the `SamplePre` algorithm (Lemma 2, Item 1.), the distribution of \mathbf{V} and \mathbf{K}_i is statistically indistinguishable between $\mathsf{H}_{3,i,1}$ and $\mathsf{H}_{3,i,2}$. Therefore

$$\text{Adv}_{3,i,1}(\mathcal{A}) \leq \text{Adv}_{3,i,2}(\mathcal{A}) + \text{negl}(\lambda).$$

- $\mathsf{H}_{3,i,3}$: This is the same as $\mathsf{H}_{3,i,2}$, except for the following modification to `KeyGen` queries when $j \neq i$:

- compute $\mathbf{T} := \begin{bmatrix} \tilde{\mathbf{W}}_i \\ -\mathbf{I}_m \end{bmatrix}$ and observe that

$$[\mathbf{B}_0 \mid \mathbf{W} + j \cdot \mathbf{G}] \cdot \mathbf{T} = [\mathbf{B}_0 \mid \mathbf{B}_0 \cdot \tilde{\mathbf{W}}_i + (j - i) \cdot \mathbf{G}] \cdot \begin{bmatrix} \tilde{\mathbf{W}}_i \\ -\mathbf{I}_m \end{bmatrix} = (i - j) \cdot \mathbf{G}.$$

- compute

$$\mathbf{K}_j \leftarrow \text{SamplePre}([\mathbf{B}_0 \mid \mathbf{W} + j \cdot \mathbf{G}], \mathbf{T}, \mathbf{V}, \tau_1; \mathbf{r}_j)$$

to answer `KeyGen` queries. This works as long as

$$\tau_1 \geq O(m^2) \geq O(m^2 \cdot \|\tilde{\mathbf{W}}\|). \quad (5)$$

Therefore, since τ_1 satisfies such constraint by our choice of parameters, we have that

$$\text{Adv}_{3,i,2}(\mathcal{A}) \leq \text{Adv}_{3,i,3}(\mathcal{A}) + \text{negl}(\lambda).$$

Notice that the reduction does not use $\mathbf{T}_{\mathbf{B}_0}$ anymore.

- $\mathsf{H}_{3,i,4}$: This is the same as $\mathsf{H}_{3,i,3}$, except for the following modification to \mathbf{B}_0 in `mpk`:

- sample $\mathbf{B}_0 \leftarrow \mathbb{Z}_q^{n \times m}$ instead of $(\mathbf{B}_0, \mathbf{T}_{\mathbf{B}_0}) \leftarrow \text{TrapGen}(1^n, 1^m, q)$.

By the properties of the `TrapGen` algorithm (Lemma 2), the distribution of \mathbf{B}_0 is statistically indistinguishable between $\mathsf{H}_{3,i,3}$ and $\mathsf{H}_{3,i,4}$. Therefore,

$$\text{Adv}_{3,i,3}(\mathcal{A}) \leq \text{Adv}_{3,i,4}(\mathcal{A}) + \text{negl}(\lambda).$$

- $\mathsf{H}_{3,i,5}$: This is the same as $\mathsf{H}_{3,i,4}$ except for the following modification to $\mathbf{c}_{3,i}$ in the challenge ciphertext:

- set

$$\mathbf{c}_{3,i} := [\mathbf{c}_{1,i} \mid \mathbf{c}_{2,i}] \cdot \mathbf{K}_i - \mathbf{s} \cdot (\mathbf{A}_i - x_i^* \cdot \mathbf{G}) + \mathbf{e}'_{3,i},$$

for $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, $\mathbf{e}'_{3,1} \leftarrow \mathcal{D}_{\mathbb{Z}^m, \chi''}$.

First, recall that in $\mathsf{H}_{3,i,4}$, we have

$$\begin{aligned} \mathbf{c}_{3,i} &= \mathbf{s}_i \cdot \mathbf{V} + x_i^* \cdot \mathbf{s} \cdot \mathbf{G} + \mathbf{e}_{3,i} \\ &= \underbrace{\mathbf{s}_i \cdot [\mathbf{B}_0 \mid \mathbf{W} + i \cdot \mathbf{G}]}_{[\mathbf{c}_{1,i} \mid \mathbf{c}_{2,i}] - [\mathbf{0} \mid \mathbf{s} \cdot \mathbf{G}] - [\mathbf{e}_{1,i} \mid \mathbf{e}_{2,i}]} \cdot \underbrace{\begin{bmatrix} \mathbf{Z}_i \\ \mathbf{R}_i \end{bmatrix}}_{\mathbf{K}_i} + x_i^* \cdot \mathbf{s} \cdot \mathbf{G} + \mathbf{e}_{3,i} \\ &= [\mathbf{c}_{1,i} \mid \mathbf{c}_{2,i}] \cdot \mathbf{K}_i - \mathbf{s} \cdot (\mathbf{A}_i - x_i^* \cdot \mathbf{G}) + \boxed{\mathbf{e}_{3,i} - [\mathbf{e}_{1,i} \mid \mathbf{e}_{2,i}] \cdot \mathbf{K}_i} \pmod{q} \end{aligned}$$

where in the last equality we have used that $\mathbf{A}_i = \mathbf{G} \cdot \mathbf{R}_i$ and the boxed term is the term in $\mathsf{H}_{3,i,4}$ that will be modified in $\mathsf{H}_{3,i,5}$. By noise flooding, we have

$$([\mathbf{e}_{1,i} \mid \mathbf{e}_{2,i}], \mathbf{K}_i, \boxed{\mathbf{e}_{3,i} - [\mathbf{e}_{1,i} \mid \mathbf{e}_{2,i}] \cdot \mathbf{K}_i}) \approx_s ([\mathbf{e}_{1,i} \mid \mathbf{e}_{2,i}], \mathbf{K}_i, \mathbf{e}'_{3,i}),$$

as long as

$$\begin{aligned} \chi'' &\geq 2 \cdot m \cdot \chi' \cdot \tau_1 \cdot \lambda^{\omega(1)}, \\ &\geq \|[\mathbf{e}_{1,i} \mid \mathbf{e}_{2,i}] \cdot \mathbf{K}_i\| \cdot \lambda^{\omega(1)}. \end{aligned}$$

We conclude that

$$\text{Adv}_{3,i,4}(\mathcal{A}) \leq \text{Adv}_{3,i,5}(\mathcal{A}) + \text{negl}(\lambda).$$

- $\mathsf{H}_{3,i,6}$: This is the same as $\mathsf{H}_{3,i,5}$, except for the following modification to $\mathbf{c}_{2,i}$ in the challenge ciphertext:

- set

$$\mathbf{c}_{2,i} := \mathbf{c}_{1,i} \cdot \tilde{\mathbf{W}}_i + \mathbf{s} \cdot \mathbf{G} + \mathbf{e}'_{2,i},$$

for $\mathbf{e}'_{2,i} \leftarrow \mathcal{D}_{\mathbb{Z}^m, \chi'}$ and $\mathbf{s} \leftarrow \mathbb{Z}_q^n$.

First, recall that in $\mathsf{H}_{3,i,5}$, we have

$$\begin{aligned} \mathbf{c}_{2,i} &= \mathbf{s}_i \cdot (\mathbf{W} + i \cdot \mathbf{G}) + \mathbf{s} \cdot \mathbf{G} + \mathbf{e}_{2,i} \\ &= \mathbf{s}_i \cdot (\mathbf{B}_0 \cdot \tilde{\mathbf{W}}_i) + \mathbf{s} \cdot \mathbf{G} + \mathbf{e}_{2,i} \\ &= \underbrace{(\mathbf{s}_i \cdot \mathbf{B}_0 + \mathbf{e}_{1,i})}_{\mathbf{c}_{1,i}} \cdot \tilde{\mathbf{W}}_i + \mathbf{s} \cdot \mathbf{G} + \boxed{\mathbf{e}_{2,i} - \mathbf{e}_{1,i} \cdot \tilde{\mathbf{W}}_i} \pmod q \end{aligned}$$

where the boxed term is the term in $\mathsf{H}_{3,i,5}$ that will be modified in $\mathsf{H}_{3,i,6}$. By noise flooding, we have

$$\left(\mathbf{e}_{1,i}, \tilde{\mathbf{W}}_i, \boxed{\mathbf{e}_{2,i} - \mathbf{e}_{1,i} \cdot \tilde{\mathbf{W}}_i} \right) \approx_s \left(\mathbf{e}_{1,i}, \tilde{\mathbf{W}}_i, \mathbf{e}'_{2,i} \right),$$

as long as

$$\begin{aligned} \chi' &\geq m \cdot \chi \cdot \lambda^{\omega(1)} \\ &\geq \|\mathbf{e}_{1,i} \cdot \tilde{\mathbf{W}}_i\| \cdot \lambda^{\omega(1)}. \end{aligned}$$

We conclude that

$$\text{Adv}_{3,i,5}(\mathcal{A}) \leq \text{Adv}_{3,i,6}(\mathcal{A}) + \text{negl}(\lambda).$$

- $\mathsf{H}_{3,i,7}$: This is the same as $\mathsf{H}_{3,i,6}$, except for the following modification to $\mathbf{c}_{1,i}$ in the challenge ciphertext:

- sample

$$\mathbf{c}_{1,i} \leftarrow \mathbb{Z}_q^m.$$

Recall that in $H_{3,i,6}$, we have

$$\mathbf{c}_{1,i} = \mathbf{s}_i \cdot \mathbf{B}_0 + \mathbf{e}_{1,i},$$

where $\mathbf{s}_i \leftarrow \mathbb{Z}_q^n$, $\mathbf{e}_{1,i} \leftarrow \mathcal{D}_{\mathbb{Z}^m, \chi}$. To show that $H_{3,i,6} \approx_c H_{3,i,7}$, we rely on $\text{LWE}_{n,m,\chi,q}$. The reduction works as follows:

- it parses $\mathbf{B} = \mathbf{B}_0 \in \mathbb{Z}_q^{n \times m}$ and $\tilde{\mathbf{c}} = \mathbf{c}_{1,i} \in \mathbb{Z}_q^m$ from the $\text{LWE}_{n,m,\chi,q}$ instance,
- it samples $\tilde{\mathbf{W}}_i \leftarrow \{0, 1\}^{m \times m}$ and computes $\mathbf{A}_0, \mathbf{W}, \mathbf{V}, \mathbf{D}$ as in $H_{3,i,6}$, while using \mathbf{B}_0 obtained from the LWE instance,
- it receives \mathbf{x}^* from the adversary \mathcal{A} ,
- it answers KeyGen queries using \mathbf{T} (which can be computed from $\tilde{\mathbf{W}}_i$) as in $H_{3,i,6}$,
- whenever the adversary \mathcal{A} outputs $(\boldsymbol{\mu}_0, \boldsymbol{\mu}_1)$, it samples

$$\begin{aligned} b &\leftarrow \{0, 1\}, \mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e}_0 \leftarrow \mathcal{D}_{\mathbb{Z}^m, \chi}, \mathbf{e}_4 \leftarrow \mathcal{D}_{\mathbb{Z}^\lambda, \chi}, \mathbf{e}_5 \leftarrow \mathcal{D}_{\mathbb{Z}^\lambda, \chi'} \\ \mathbf{c}_{1,j} &\leftarrow \mathbb{Z}_q^m, \mathbf{c}_{2,j} \leftarrow \mathbb{Z}_q^m \quad \text{for } j \in [i-1], \\ \mathbf{e}'_{2,i} &\leftarrow \mathcal{D}_{\mathbb{Z}^m, \chi'}, \mathbf{e}_{3,i} \leftarrow \mathcal{D}_{\mathbb{Z}^m, \chi''}, \text{ and} \\ \mathbf{s}_j &\leftarrow \mathbb{Z}_q^n, \mathbf{e}_{1,j} \leftarrow \mathcal{D}_{\mathbb{Z}^m, \chi}, \mathbf{e}_{2,j} \leftarrow \mathcal{D}_{\mathbb{Z}^m, \chi'}, \mathbf{e}_{3,j} \leftarrow \mathcal{D}_{\mathbb{Z}^m, \chi''} \text{ for } j \in [i+1 : \ell^*] \end{aligned}$$

and outputs

$$\text{ct} = \left(\begin{array}{ccc} \mathbf{s} \cdot \mathbf{A}_0 + \mathbf{e}_0 & & \\ \{\mathbf{c}_{1,j}\}_{j \in [i]}, & \{\mathbf{s}_j \cdot \mathbf{B}_0 + \mathbf{e}_{1,j}\}_{j \in [i+1:\ell^*]} & \\ \{\mathbf{c}_{2,j}\}_{j \in [i-1]}, & \mathbf{c}_{1,i} \cdot \tilde{\mathbf{W}}_i + \mathbf{s} \cdot \mathbf{G} + \mathbf{e}_{2,i}, & \{\mathbf{s}_j \cdot (\mathbf{W} + j \cdot \mathbf{G}) + \mathbf{s} \cdot \mathbf{G} + \mathbf{e}_{2,j}\}_{j \in [i+1:\ell^*]} \\ \{\mathbf{c}_{1,j} \mid \mathbf{c}_{2,j}\} \cdot \mathbf{K}_j - \mathbf{s} \cdot (\mathbf{A}_i - \mathbf{x}_i^* \cdot \mathbf{G}) + \mathbf{e}'_{3,i}\}_{j \in [i]}, & \{\mathbf{s}_j \cdot \mathbf{V} + \mathbf{x}_i^* \cdot \mathbf{s} \cdot \mathbf{G} + \mathbf{e}_{3,j}\}_{j \in [i+1:\ell^*]} & \\ & \mathbf{s} \cdot \mathbf{D} + \boldsymbol{\mu} \cdot \lfloor q/2 \rfloor + \mathbf{e}_4 & \\ & \mathbf{s} \cdot (\mathbf{B}_1 - |\mathbf{x}^*| \cdot \mathbf{G}) + \mathbf{e}_5 & \end{array} \right)$$

Observe that

- if $(\mathbf{B}, \tilde{\mathbf{c}})$ is a structured $\text{LWE}_{n,m,\chi,q}$ instance, the view of the adversary \mathcal{A} is identical to $H_{3,i,6}$;
- if $(\mathbf{B}, \tilde{\mathbf{c}})$ is a uniform random instance, the view of \mathcal{A} is identical to $H_{3,i,7}$.

We conclude that

$$\text{Adv}_{3,i,6}(\mathcal{A}) \leq \text{Adv}_{3,i,7}(\mathcal{A}) + \text{Adv}_{\mathcal{B}_1}^{\text{LWE}_{n,m,\chi,q}}(\lambda).$$

- $H_{3,i,8}$: This is the same as $H_{3,i,7}$, except for the following modification to $\mathbf{c}_{2,i}$ in the challenge ciphertext:
 - sample

$$\mathbf{c}_{2,i} \leftarrow \mathbb{Z}_q^m.$$

Recall that in $H_{3,i,7}$, we have

$$\mathbf{c}_{2,i} = \mathbf{c}_{1,i} \cdot \tilde{\mathbf{W}}_i + \mathbf{s} \cdot \mathbf{G} + \mathbf{e}'_{2,i}.$$

Since $\mathbf{c}_{1,i}$ is uniform random in \mathbb{Z}_q^m in $H_{3,i,7}$, $\tilde{\mathbf{W}}_i$ is sampled uniformly and $m \geq (n+1) \cdot \log q + 2 \cdot \lambda + \omega(\log n)$, indistinguishability ($H_{3,i,7} \approx_s H_{3,i,8}$) follows from the generalized leftover hash lemma (the adversary's view includes $\mathbf{W} = \mathbf{B}_0 \cdot \tilde{\mathbf{W}}_i - i \cdot \mathbf{G}$), that is

$$\text{Adv}_{3,i,7}(\mathcal{A}) \leq \text{Adv}_{3,i,8}(\mathcal{A}) + \text{negl}(\lambda).$$

- $H_{3,i,9}$: This is the same as $H_{3,i,8}$, except for the following modification to \mathbf{B}_0 in `mpk` and to the `KeyGen` queries:
 - sample $(\mathbf{B}_0, \mathbf{T}_{\mathbf{B}_0}) \leftarrow \text{TrapGen}(1^n, 1^m, q)$, instead of $\mathbf{B}_0 \leftarrow \mathbb{Z}_q^{n \times m}$,
 - compute

$$\mathbf{K}_j \leftarrow \text{SamplePre}([\mathbf{B}_0 | \mathbf{W} + j \cdot \mathbf{G}], \begin{bmatrix} \mathbf{T}_{\mathbf{B}_0} \\ \mathbf{0}_{m \times m} \end{bmatrix}, \mathbf{V}, \tau_1; \mathbf{r}_j),$$

to answer `KeyGen` queries.

By the properties of the `TrapGen` algorithm (Lemma 2) and that of the `SamplePre` algorithm, the distribution of \mathbf{B}_0 and that of answers to the `KeyGen` queries are statistically indistinguishable between $H_{3,i,8}$ and $H_{3,i,9}$. Therefore,

$$\text{Adv}_{3,i,8}(\mathcal{A}) \leq \text{Adv}_{3,i,9}(\mathcal{A}) + \text{negl}(\lambda).$$

- H_4 : This is the same as $H_{3,\ell,9}$, except for the following modification to \mathbf{K}_j for all $j \in [T]$, and to the relative `KeyGen` queries:
 - sample, once and for all key queries, $\mathbf{R}_j \leftarrow \mathcal{D}_{\mathbb{Z}^{m \times m}, \tau_1}$,
 - compute

$$\mathbf{Z}_j \leftarrow \text{SamplePre}(\mathbf{B}_0, \mathbf{T}_{\mathbf{B}_0}, \mathbf{V} - [\mathbf{W} + j \cdot \mathbf{G}] \cdot \mathbf{R}_j, \tau_1; \mathbf{r}_{j,1}),$$

- set $\mathbf{K}_j := \begin{bmatrix} \mathbf{Z}_j \\ \mathbf{R}_j \end{bmatrix}$.

By the properties of the `SamplePre` algorithm and Lemma 3, the distribution of $\{\mathbf{K}_j\}_{j \in [T]}$ is statistically indistinguishable between $H_{3,\ell,9}$ and H_4 . Therefore,

$$\text{Adv}_{3,\ell,9}(\mathcal{A}) \leq \text{Adv}_4(\mathcal{A}) + \text{negl}(\lambda).$$

- H_5 : This is the same as H_4 , except for the following modification to $\mathbf{A}_j, \mathbf{K}_j$ for all $j \in [T]$, and to the relative `KeyGen` queries:
 - sample $\mathbf{A}_j \leftarrow \mathbb{Z}_q^{n \times m}$,
 - set $\mathbf{R}_j = \text{SamplePre}(\mathbf{G}, \mathbf{I}, \mathbf{A}_j, \tau_1; \mathbf{r}_{j,2})$,
 - compute

$$\mathbf{Z}_j \leftarrow \text{SamplePre}(\mathbf{B}_0, \mathbf{T}_{\mathbf{B}_0}, \mathbf{V} - [\mathbf{W} + j \cdot \mathbf{G}] \cdot \mathbf{R}_j, \tau_1; \mathbf{r}_{j,1}),$$

- set $\mathbf{K}_j := \begin{bmatrix} \mathbf{Z}_j \\ \mathbf{R}_j \end{bmatrix}$.

By the properties of the `SamplePre` algorithm (Lemma 2, Item 1.), the distribution of $\{\mathbf{A}_j, \mathbf{K}_j\}_{j \in [T]}$ is statistically indistinguishable between H_4 and H_5 . Therefore,

$$\text{Adv}_4(\mathcal{A}) \leq \text{Adv}_5(\mathcal{A}) + \text{negl}(\lambda).$$

- H_6 : This is the same as H_5 , except for the following modification to \mathbf{B}_1 and \mathbf{A}_j for all $j \in [\ell^*]$:
 - sample $\mathbf{U} \leftarrow \{0, 1\}^{m \times m}$ and $\mathbf{U}_j \leftarrow \{0, 1\}^{m \times m}$ for $j \in [\ell^*]$,
 - set $\mathbf{B}_1 = \mathbf{A}_0 \cdot \mathbf{U} + \ell^* \cdot \mathbf{G}$ and $\mathbf{A}_j := \mathbf{A}_0 \cdot \mathbf{U}_j + x_j^* \cdot \mathbf{G}$ for $j \in [\ell^*]$.

Since \mathbf{U}, \mathbf{U}_j are sampled uniformly and $m \geq (n + 1) \cdot \log q + 2 \cdot \lambda$, indistinguishability ($\mathbf{H}_5 \approx_s \mathbf{H}_6$) follows from the leftover hash lemma, that is

$$\text{Adv}_5(\mathcal{A}) \leq \text{Adv}_6(\mathcal{A}) + \text{negl}(\lambda).$$

- \mathbf{H}_7 : This is the same as \mathbf{H}_6 , except for the following modification to change answers to **KeyGen** queries
 - recall the key equation

$$(\mathbf{A} - \mathbf{x} \otimes \mathbf{G}) \cdot \mathbf{H}_{\mathbf{A},f,\mathbf{x}} = \mathbf{A}_f - f(\mathbf{x}) \cdot \mathbf{G} \pmod{q},$$

and that a valid adversary can only make **KeyGen** queries for functions f for which $f(\mathbf{x}^*) = 1$, and that $|\mathbf{x}^*| = \ell^*$. Using these facts, for functions f with input length ℓ^* , one has that

$$\begin{aligned} \mathbf{A}_f &= (\mathbf{A} - \mathbf{x}^* \otimes \mathbf{G}) \cdot \mathbf{H}_{\mathbf{A},f,\mathbf{x}^*} + f(\mathbf{x}^*) \cdot \mathbf{G} \\ &= ([\mathbf{A}_1 | \dots | \mathbf{A}_{\ell^*}] - \mathbf{x}^* \otimes \mathbf{G}) \cdot \mathbf{H}_{\mathbf{A},f,\mathbf{x}^*} + f(\mathbf{x}^*) \cdot \mathbf{G} \\ &= (\mathbf{A}_0 \cdot [\mathbf{U}_1 | \dots | \mathbf{U}_{\ell^*}] + \mathbf{x}^* \otimes \mathbf{G} - \mathbf{x}^* \otimes \mathbf{G}) \cdot \mathbf{H}_{\mathbf{A},f,\mathbf{x}^*} + f(\mathbf{x}^*) \cdot \mathbf{G} \\ &= (\mathbf{A}_0 \cdot \underbrace{[\mathbf{U}_1 | \dots | \mathbf{U}_{\ell^*}] \cdot \mathbf{H}_{\mathbf{A},f,\mathbf{x}^*}}_{\mathbf{U}_f} + f(\mathbf{x}^*) \cdot \mathbf{G}) \\ &= (\mathbf{A}_0 \cdot \mathbf{U}_f + \mathbf{G} \pmod{q}, \end{aligned}$$

where in the second equality we have used the definition of \mathbf{A}_j for $j \in [\ell^*]$.

- compute $\mathbf{T}_f := \begin{bmatrix} -\mathbf{U}_f \\ \mathbf{I}_m \mathbf{0}_{m \times m} \end{bmatrix}$ and observe that $[\mathbf{A}_0 | \mathbf{A}_f | \mathbf{B}_1 - \ell^* \cdot \mathbf{G}] \cdot \mathbf{T}_f = \mathbf{G}$.
- for functions f whose input length is ℓ^* , compute

$$\mathbf{K}_f \leftarrow \text{SamplePre}([\mathbf{A}_0 | \mathbf{A}_f | \mathbf{B}_1 - \ell \cdot \mathbf{G}], \mathbf{T}_f, \mathbf{D}, \tau_2)$$

to answer **KeyGen** queries.

- for functions f with input length $\ell \neq \ell^*$, observe that

$$\begin{aligned} [\mathbf{A}_0 | \mathbf{A}_f | \mathbf{B}_1 - \ell \cdot \mathbf{G}] \cdot \begin{bmatrix} -\mathbf{U} \\ \mathbf{0}_{m \times m} \\ \mathbf{I}_m \end{bmatrix} &= [\mathbf{A}_0 | \mathbf{A}_f | \mathbf{A}_0 \cdot \mathbf{U} + \ell^* \cdot \mathbf{G} - \ell \cdot \mathbf{G}] \cdot \begin{bmatrix} -\mathbf{U} \\ \mathbf{0}_{m \times m} \\ \mathbf{I}_m \end{bmatrix} \\ &= [\mathbf{A}_0 | \mathbf{A}_f | \mathbf{A}_0 \cdot \mathbf{U} + (\ell^* - \ell) \cdot \mathbf{G}] \cdot \begin{bmatrix} -\mathbf{U} \\ \mathbf{0}_{m \times m} \\ \mathbf{I}_m \end{bmatrix} \\ &= \underbrace{(\ell^* - \ell)}_{\neq 0} \cdot \mathbf{G} \pmod{q}. \end{aligned}$$

- for such functions (with input length $\ell \neq \ell^*$), compute

$$\mathbf{K}_f \leftarrow \text{SamplePre} \left([\mathbf{A}_0 | \mathbf{A}_f | \mathbf{B}_1 - \ell \cdot \mathbf{G}], \begin{bmatrix} -\mathbf{U} \\ \mathbf{0}_{m \times m} \\ \mathbf{I}_m \end{bmatrix}, \mathbf{D}, \tau_2 \right)$$

to answer **KeyGen** queries.

- these procedures work as long as

$$\begin{aligned}
 \tau_2 &\geq m^3 \cdot \ell^* \cdot B \\
 &\geq m^2 \cdot m \cdot \ell^* \cdot B \\
 &\geq O(m^2 \cdot (\|\mathbf{U}_f\| + \|\mathbf{U}\|))
 \end{aligned} \tag{6}$$

By properties of the `SamplePre` algorithm,

$$\text{Adv}_6(\mathcal{A}) \leq \text{Adv}_7(\mathcal{A}) + \text{negl}(\lambda).$$

Notice that the reduction does not use $\mathbf{T}_{\mathbf{A}_0}$ anymore.

- \mathbf{H}_8 : This is the same as \mathbf{H}_7 , except for the following modification to \mathbf{A}_0 in `mpk`:

- sample $\mathbf{A}_0 \leftarrow \mathbb{Z}_q^{n \times m}$ instead of $(\mathbf{A}_0, \mathbf{T}_{\mathbf{A}_0}) \leftarrow \text{TrapGen}(1^n, 1^m, q)$.

By the properties of the `TrapGen` algorithm (Lemma 2), the distribution of \mathbf{A}_0 is statistically indistinguishable between \mathbf{H}_7 and \mathbf{H}_8 . Therefore

$$\text{Adv}_7(\mathcal{A}) \leq \text{Adv}_8(\mathcal{A}) + \text{negl}(\lambda).$$

- \mathbf{H}_9 : This is the same as \mathbf{H}_8 except for the following modification to $\mathbf{c}_{3,j}$ and \mathbf{c}_5 in the challenge ciphertext:

- set

$$\begin{aligned}
 \mathbf{c}_{3,j} &:= [\mathbf{c}_{1,j} \mid \mathbf{c}_{2,j}] \cdot \mathbf{K}_j - \mathbf{c}_0 \cdot \mathbf{U}_j + \mathbf{e}''_{3,j}, \text{ and} \\
 \mathbf{c}_5 &:= \mathbf{c}_0 \cdot \mathbf{U} + \mathbf{e}'_5,
 \end{aligned}$$

for $\mathbf{e}_{3,j} \leftarrow \mathcal{D}_{\mathbb{Z}^m, \chi''}$, $\mathbf{e}'_5 \leftarrow \mathcal{D}_{\mathbb{Z}^m, \chi'}$.

First, recall that in \mathbf{H}_8 , we have

$$\begin{aligned}
 \mathbf{c}_{3,j} &= [\mathbf{c}_{1,j} \mid \mathbf{c}_{2,j}] \cdot \mathbf{K}_j - \underbrace{\mathbf{s} \cdot \mathbf{A}_0 \cdot \mathbf{U}_j}_{\mathbf{c}_0 \cdot \mathbf{U}_j - \mathbf{e}_0 \cdot \mathbf{U}_j} + \mathbf{e}'_{3,j} \\
 &= [\mathbf{c}_{1,j} \mid \mathbf{c}_{2,j}] \cdot \mathbf{K}_j - \mathbf{c}_0 \cdot \mathbf{U}_j + \boxed{\mathbf{e}_0 \cdot \mathbf{U}_j + \mathbf{e}'_{3,j}} \pmod q,
 \end{aligned}$$

and

$$\begin{aligned}
 \mathbf{c}_5 &= \underbrace{\mathbf{s} \cdot \mathbf{A}_0 \cdot \mathbf{U}}_{\mathbf{c}_0 \cdot \mathbf{U} - \mathbf{e}_0 \cdot \mathbf{U}} + \mathbf{e}_5 \\
 &= \mathbf{c}_0 \cdot \mathbf{U} + \boxed{\mathbf{e}_5 - \mathbf{e}_0 \cdot \mathbf{U}} \pmod q.
 \end{aligned}$$

where the boxed terms are the term in \mathbf{H}_8 that will be modified in \mathbf{H}_9 . By noise flooding, we have

$$\left(\mathbf{e}'_{3,j}, \mathbf{U}_j, \boxed{\mathbf{e}_0 \cdot \mathbf{U}_j + \mathbf{e}'_{3,j}} \right) \approx_s \left(\mathbf{e}'_{3,j}, \mathbf{U}_j, \mathbf{e}''_{3,j} \right),$$

and

$$\left(\mathbf{e}_0, \mathbf{U}, \boxed{\mathbf{e}_5 - \mathbf{e}_0 \cdot \mathbf{U}} \right) \approx_s \left(\mathbf{e}_0, \mathbf{U}, \mathbf{e}'_5 \right),$$

as long as

$$\begin{aligned}\chi'' &\geq m \cdot \chi \cdot \lambda^{\omega(1)}, \\ &\geq \|\mathbf{e}_0 \cdot \mathbf{U}_j\| \cdot \lambda^{\omega(1)},\end{aligned}$$

and

$$\begin{aligned}\chi' &\geq m \cdot \chi \cdot \lambda^{\omega(1)}, \\ &\geq \|\mathbf{e}_0 \cdot \mathbf{U}\| \cdot \lambda^{\omega(1)},\end{aligned}$$

respectively. We conclude that

$$\text{Adv}_8(\mathcal{A}) \leq \text{Adv}_9(\mathcal{A}) + \text{negl}(\lambda).$$

- H_{10} : This is the same as H_9 , except for the following modification to \mathbf{c}_0 and \mathbf{c}_4 in the challenge ciphertext:

- sample

$$\mathbf{c}_0 \leftarrow \mathbb{Z}_q^m, \mathbf{c}_4 \leftarrow \mathbb{Z}_q^\lambda$$

Recall that in H_9 , we have

$$\mathbf{c}_0 = \mathbf{s} \cdot \mathbf{A}_0 + \mathbf{e}_0, \quad \mathbf{c}_4 = \mathbf{s} \cdot \mathbf{D} + \boldsymbol{\mu} \cdot \lfloor q/2 \rfloor + \mathbf{e}_4$$

where $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, $\mathbf{e}_0 \leftarrow \mathcal{D}_{\mathbb{Z}^m, \chi}$, and $\mathbf{e}_4 \leftarrow \mathcal{D}_{\mathbb{Z}^m, \chi}$. To show that $H_9 \approx_c H_{10}$, we rely on $\text{LWE}_{n, m+\lambda, \chi, q}$. The reduction works as follows:

- it parses $\mathbf{B} = [\mathbf{A}_0 | \mathbf{D}] \in \mathbb{Z}_q^{n \times (m+\lambda)}$ and $\tilde{\mathbf{c}} = [\mathbf{c}_0 | \mathbf{c}_4] \in \mathbb{Z}_q^{m+\lambda}$ from the $\text{LWE}_{n, m+\lambda, \chi, q}$ instance,
- it samples $\mathbf{B}_0, \mathbf{W}, \mathbf{V}, \mathbf{U}$ as in H_7 , while using \mathbf{A}_0, \mathbf{D} obtained from the LWE instance,
- it receives \mathbf{x}^* from the adversary \mathcal{A} ,
- it samples $\{\mathbf{U}_j\}_{j \in [\ell^*]}$ and implicitly sets $\{\mathbf{A}_j\}_{j \in [\ell^*]}$ as in H_9 ,
- it answers KeyGen queries using \mathbf{U}_f or \mathbf{U} as in H_9 ,
- whenever the adversary \mathcal{A} outputs $(\boldsymbol{\mu}_0, \boldsymbol{\mu}_1)$, it samples

$$\begin{aligned}b &\leftarrow \{0, 1\}, \mathbf{e}'_5 \leftarrow \mathcal{D}_{\mathbb{Z}^\lambda, \chi'} \\ \mathbf{c}_{1,j} &\leftarrow \mathbb{Z}_q^m, \mathbf{c}_{2,j} \leftarrow \mathbb{Z}_q^m \quad \text{for } j \in [\ell^*], \text{ and} \\ \mathbf{e}''_{3,j} &\leftarrow \mathcal{D}_{\mathbb{Z}^m, \chi''} \quad \text{for } j \in [\ell^*]\end{aligned}$$

and outputs

$$\text{ct} = \left(\begin{array}{c} \mathbf{c}_0 \\ \{\mathbf{c}_{1,j}\}_{j \in [\ell^*]}, \\ \{\mathbf{c}_{2,j}\}_{j \in [\ell^*]}, \\ \{[\mathbf{c}_{1,j} \mid \mathbf{c}_{2,j}] \cdot \mathbf{K}_j - \mathbf{c}_0 \cdot \mathbf{U}_j + \mathbf{e}''_{3,i}\}_{j \in [\ell^*]} \\ \mathbf{c}_4 + \boldsymbol{\mu}_b \cdot \lfloor q/2 \rfloor \\ \mathbf{c}_0 \cdot \mathbf{U} + \mathbf{e}'_5 \end{array} \right)$$

Observe that

- if $(\mathbf{B}, \tilde{\mathbf{c}})$ is a structured $\text{LWE}_{n,m+\lambda,\chi,q}$ instance, the view of the adversary \mathcal{A} is identical to H_9 ;
- if $(\mathbf{B}, \tilde{\mathbf{c}})$ is a uniform random instance, the view of \mathcal{A} is identical to H_{10} .

We conclude that

$$\text{Adv}_9(\mathcal{A}) \leq \text{Adv}_{10}(\mathcal{A}) + \text{Adv}_{\mathcal{B}_2}^{\text{LWE}_{n,m+\lambda,\chi,q}}(\lambda).$$

Putting everything together, we obtain

$$\text{Adv}_{\mathcal{A},\Pi}^{\text{sa-ABE}}(\lambda) \leq T \cdot \left(\text{Adv}_{\mathcal{B}_0,\mathcal{F}}^{\text{PRF}}(\lambda) + \text{Adv}_{\mathcal{B}_1}^{\text{LWE}_{n,m,\chi,q}}(\lambda) + \text{Adv}_{\mathcal{B}_2}^{\text{LWE}_{n,m+\lambda,\chi,q}}(\lambda) + \text{negl}(\lambda) \right),$$

as claimed. □

4 Unbounded Inner Product Predicate Encryption

We consider inner product predicate encryption, where ciphertexts are associated with $\mathbf{x} \in \mathbb{Z}_q^\ell$, keys with $\mathbf{y} \in \mathbb{Z}_q^\ell$, and decryption is possible iff $\mathbf{x} \cdot \mathbf{y}^\top = 0$, where q is prime.

Predicate Encryption. The syntax is exactly the same as ABE except Dec only gets $(\text{mpk}, \text{sk}_f, f, \text{ct}_x)$ but not \mathbf{x} . Correctness is defined analogously. For security, we require weak attribute-hiding with semi-adaptive security as captured by the following advantage function:

$$\text{Adv}_{\mathcal{A},\Pi}^{\text{PE}}(\lambda) := \Pr \left[\begin{array}{l} (\text{mpk}, \text{msk}) \leftarrow \text{Setup}(1^\lambda, \mathcal{F}) \\ \mathbf{x}_0^*, \mathbf{x}_1^* \leftarrow \mathcal{A}(1^\lambda, \text{mpk}) \\ b = b' : (\boldsymbol{\mu}_0, \boldsymbol{\mu}_1) \leftarrow \mathcal{A}^{\text{KeyGen}(\text{mpk}, \text{msk}, \cdot)}(\text{mpk}) \\ b \leftarrow \{0, 1\}; \text{ct} \leftarrow \text{Enc}(\text{mpk}, \mathbf{x}_b^*, \boldsymbol{\mu}_b) \\ b' \leftarrow \mathcal{A}^{\text{KeyGen}(\text{mpk}, \text{msk}, \cdot)}(\text{ct}) \end{array} \right] - \frac{1}{2},$$

with the restriction that $|\mathbf{x}_0^*| = |\mathbf{x}_1^*|$ and all queries $f : \{0, 1\}^\ell \rightarrow \{0, 1\}$ that \mathcal{A} sent to $\text{KeyGen}(\text{mpk}, \text{msk}, \cdot)$ satisfy either (i) $\ell \neq |\mathbf{x}_0^*|$ or (ii) $f(\mathbf{x}_0^*) \neq 0$ and $f(\mathbf{x}_1^*) \neq 0$.

Homomorphic Computation on Matrices. Following [AFV11], we have:

$$(\mathbf{A} - \mathbf{x} \otimes \mathbf{G}_{n,q}) \cdot \overbrace{(\mathbf{I}_\ell \otimes \mathbf{G}_{n,q})^{-1}(\mathbf{y}^\top)}^{\mathbf{H}_{\mathbf{A},\mathbf{y}}} = \overbrace{\mathbf{A} \cdot (\mathbf{I}_\ell \otimes \mathbf{G}_{n,q})^{-1}(\mathbf{y}^\top)}^{\mathbf{A}_y} - \mathbf{x} \cdot \mathbf{y}^\top \otimes \mathbf{G} \quad (7)$$

Observe that computing $\mathbf{H}_{\mathbf{A},\mathbf{y}}$ does not require knowing \mathbf{x} .

4.1 Our Construction

Our inner product predicate encryption scheme Π' is exactly the same as our ABE, with the following modifications:

- In Enc, we have $\mathbf{x} \in \mathbb{Z}_q^\ell, x_j \in \mathbb{Z}_q$ instead of $\mathbf{x} \in \{0, 1\}^\ell, x_j \in \{0, 1\}$;
- In KeyGen, we replace \mathbf{A}_f in \mathbf{K}_f with \mathbf{A}_y in \mathbf{K}_y , namely

$$\boxed{\mathbf{K}_y} \leftarrow \text{SamplePre} \left([\mathbf{A}_0 | \boxed{\mathbf{A}_y} | \mathbf{B}_1 - \ell \cdot \mathbf{G}], \begin{bmatrix} \mathbf{T}_{\mathbf{A}_0} \\ \mathbf{0}_{m \times m} \\ \mathbf{0}_{m \times m} \end{bmatrix}, \mathbf{D}, \tau_2 \right).$$

- In Dec, we replace $\mathbf{H}_{\mathbf{A},f,\mathbf{x}}$ with $\mathbf{H}_{\mathbf{A},\mathbf{y}}$ and \mathbf{K}_f with \mathbf{K}_y , namely

$$\mathbf{c}_4 - \left[\mathbf{c}_0 \mid \left(\left([\mathbf{c}_{1,1} \ \mathbf{c}_{2,1}] \cdot \mathbf{K}_1 \mid \dots \mid [\mathbf{c}_{1,\ell} \ \mathbf{c}_{2,\ell}] \cdot \mathbf{K}_\ell \right) - [\mathbf{c}_{3,1} \ \dots \ \mathbf{c}_{3,\ell}] \right) \cdot \boxed{\mathbf{H}_{\mathbf{A},\mathbf{y}}} \mid \mathbf{c}_5 \right] \cdot \boxed{\mathbf{K}_y}$$

- We can set the parameters as before, but with $B = 1$ (since $\|\mathbf{H}_{\mathbf{A},\mathbf{y}}\| \leq 1$), which yields:

Theorem 4 (Correctness). *Let Π' be the inner product predicate encryption scheme just described, with parameters as in Eq. (3) and $B = 1$. Then Π' is correct.*

Correctness is exactly the same as in the ABE, except we use (7). In particular, in the derivation of correctness we only need to replace Eq. (4) with

$$\begin{aligned} \left[\mathbf{c}_0 \mid \left(\left([\mathbf{c}_{1,1} \ \mathbf{c}_{2,1}] \cdot \mathbf{K}_1 \mid \dots \mid [\mathbf{c}_{1,\ell} \ \mathbf{c}_{2,\ell}] \cdot \mathbf{K}_\ell \right) - [\mathbf{c}_{3,1} \ \dots \ \mathbf{c}_{3,\ell}] \right) \cdot \mathbf{H}_{\mathbf{A},\mathbf{y}} \mid \mathbf{c}_5 \right] \\ = \mathbf{s} \cdot [\mathbf{A}_0 | \mathbf{A}_y | \mathbf{B}_1 - \ell \cdot \mathbf{G}] + \mathbf{e}'_{f,\mathbf{y}} \text{ mod } q. \end{aligned} \tag{8}$$

4.2 Security Proof

We sketch here the main modifications required. The security proof starts out exactly as in our ABE, except

- we replace \mathbf{A}_f with \mathbf{A}_y and $\mathbf{H}_{\mathbf{A},f,\mathbf{x}^*}$ with $\mathbf{H}_{\mathbf{A},\mathbf{y}}$ and \mathbf{x}^* with \mathbf{x}_b^* .
- to show $\mathbf{H}_6 \approx_s \mathbf{H}_7$, we use $\mathbf{x}_b^* \cdot \mathbf{y}^\dagger \neq 0$, instead of $f(\mathbf{x}^*) \neq 0$.

In addition, we require the following additional games analogous to those in [AFV11]:

- \mathbf{H}_{11} : sample \mathbf{A}_0 together with a trapdoor $\mathbf{T}_{\mathbf{A}_0}$ via $(\mathbf{A}_0, \mathbf{T}_{\mathbf{A}_0}) \leftarrow \text{TrapGen}(1^n, 1^m, q)$. We have $\mathbf{H}_{10} \approx_s \mathbf{H}_{11}$ by properties of the TrapGen algorithm.
- \mathbf{H}_{12} : sample \mathbf{K}_y using the trapdoor for \mathbf{A}_0 :

$$\mathbf{K}_y \leftarrow \text{SamplePre} \left([\mathbf{A}_0 | \mathbf{A}_y | \mathbf{B}_1 - \ell \cdot \mathbf{G}], \begin{bmatrix} \mathbf{T}_{\mathbf{A}_0} \\ \mathbf{0}_{m \times m} \\ \mathbf{0}_{m \times m} \end{bmatrix}, \mathbf{D}, \tau_2 \right)$$

We have $\mathbf{H}_{11} \approx_s \mathbf{H}_{12}$ by properties of the SamplePre algorithm.

Hybrid	mpk	ct	sk _f	justification
H ₀	$(\mathbf{A}_0, \mathbf{T}_{\mathbf{A}_0}) \leftarrow \text{TrapGen}(1^n, 1^m, q)$ $(\mathbf{B}_0, \mathbf{T}_{\mathbf{B}_0}) \leftarrow \text{TrapGen}(1^n, 1^m, q)$ $\mathbf{W} \leftarrow \mathbb{Z}_q^{n \times m}$ $\mathbf{V} \leftarrow \mathbb{Z}_q^{n \times m}$ $\mathbf{B}_1 \leftarrow \mathbb{Z}_q^{n \times m}$ $\mathbf{D} \leftarrow \mathbb{Z}_q^{n \times \lambda}$ $\mathbf{k} \leftarrow \mathcal{K}$	$\mathbf{c}_0 \approx \mathbf{s} \cdot \mathbf{A}_0$ $\mathbf{c}_{1,j} \approx \mathbf{s}_j \cdot \mathbf{B}_0$ $\mathbf{c}_{2,j} \approx \mathbf{s}_j \cdot (\mathbf{W} + \mathbf{j} \cdot \mathbf{G}) + \mathbf{s} \cdot \mathbf{G}$ $\mathbf{c}_{3,j} \approx \mathbf{s}_j \cdot \mathbf{V} + x_{b,j}^* \cdot \mathbf{s} \cdot \mathbf{G}$ $\mathbf{c}_4 \approx \mathbf{s} \cdot \mathbf{D} + \boldsymbol{\mu} \cdot \lfloor q/2 \rfloor$ $\mathbf{c}_5 \approx \mathbf{s} \cdot (\mathbf{B}_1 - \mathbf{x}_i^* \cdot \mathbf{G})$	$\mathbf{T}_{\mathbf{A}_0}, \mathbf{T}_{\mathbf{B}_0}, \mathbf{k}$ $\mathbf{K}_j = \begin{bmatrix} \mathbf{Z}_j \\ \mathbf{R}_j \end{bmatrix} \leftarrow \text{SamplePre} \left(\begin{bmatrix} [\mathbf{B}_0] \mathbf{W} + \mathbf{j} \cdot \mathbf{G} \\ \mathbf{T}_{\mathbf{B}_0} \\ \mathbf{0}_{m \times m} \end{bmatrix}, \mathbf{V}, \tau_1; \mathbf{F}(\mathbf{k}, \mathbf{j}) \right)$ $\mathbf{A}_j = \mathbf{G} \cdot \mathbf{R}_j$ $\mathbf{K}_f \leftarrow \text{SamplePre} \left(\begin{bmatrix} [\mathbf{A}_0] \mathbf{A}_j [\mathbf{B}_1 - \ell \cdot \mathbf{G}] \\ \mathbf{T}_{\mathbf{A}_0} \\ \mathbf{0}_{m \times m} \end{bmatrix}, \mathbf{D}, \tau_2 \right)$	
H ₁	↓	↓	↓	guess $ \mathbf{x}^* = \ell^*$
H ₂	↓	↓	$\mathbf{K}_j = \begin{bmatrix} \mathbf{Z}_j \\ \mathbf{R}_j \end{bmatrix} \leftarrow \text{SamplePre} \left(\begin{bmatrix} [\mathbf{B}_0] \mathbf{W} + \mathbf{j} \cdot \mathbf{G} \\ \mathbf{T}_{\mathbf{B}_0} \\ \mathbf{0}_{m \times m} \end{bmatrix}, \mathbf{V}, \tau_1; \mathbf{r}_j \right)$	PRF security
H ₃	↓	$\mathbf{c}_{1,j}, \mathbf{c}_{2,j} \leftarrow \mathbb{Z}_q^m$ $\mathbf{c}_{3,j} \approx [\mathbf{c}_{1,j} \mid \mathbf{c}_{2,j}] \cdot \mathbf{K}_j - \mathbf{s} \cdot (\mathbf{A}_j - x_{b,j}^* \cdot \mathbf{G})$	↓	LWE _{n,m,χ,q} (Fig. 3)
H ₄	↓	↓	$\mathbf{R}_j \leftarrow \mathcal{D}_{\mathbb{Z}^{m \times m}, \tau_1}$ $\mathbf{Z}_j = \text{SamplePre} \left(\begin{bmatrix} \mathbf{B}_0, \mathbf{T}_{\mathbf{B}_0} \\ \mathbf{V} - [\mathbf{W} + \mathbf{j} \cdot \mathbf{G}] \cdot \mathbf{R}_j, \tau_1; \mathbf{r}_{j,1} \end{bmatrix} \right)$ $\mathbf{K}_j = \begin{bmatrix} \mathbf{Z}_j \\ \mathbf{R}_j \end{bmatrix}$	Lemma 3
H ₅	↓	↓	$\mathbf{A}_j \leftarrow \mathbb{Z}_q^{n \times m}$ $\mathbf{R}_j = \text{SamplePre}(\mathbf{G}, \mathbf{I}, \mathbf{A}_j, \tau_1; \mathbf{r}_{j,2})$ $\mathbf{Z}_j = \text{SamplePre} \left(\begin{bmatrix} \mathbf{B}_0, \mathbf{T}_{\mathbf{B}_0} \\ \mathbf{V} - [\mathbf{W} + \mathbf{j} \cdot \mathbf{G}] \cdot \mathbf{R}_j, \tau_1; \mathbf{r}_{j,1} \end{bmatrix} \right)$ $\mathbf{K}_j = \begin{bmatrix} \mathbf{Z}_j \\ \mathbf{R}_j \end{bmatrix}$	Lemma 2 (Item 1)
H ₆	$\mathbf{B}_1 = \mathbf{A}_0 \cdot \mathbf{U} + \ell^* \cdot \mathbf{G}$	↓	$\mathbf{A}_j = \mathbf{A}_0 \cdot \mathbf{U}_j + x_{b,j}^* \cdot \mathbf{G}$	LHL
H ₇	↓	↓	\mathbf{K}_y using $\begin{cases} \begin{bmatrix} [\mathbf{U}_1] \dots [\mathbf{U}_{\ell^*}] \cdot \mathbf{H}_{\mathbf{A}, \mathbf{y}} \\ \mathbf{I}_m \\ \mathbf{0}_{m \times m} \end{bmatrix} & \text{if } \ell = \ell^* \\ \begin{bmatrix} -\mathbf{U} \\ \mathbf{0}_{m \times m} \\ \mathbf{I}_m \end{bmatrix} & \text{if } \ell \neq \ell^* \end{cases}$	Lemma 2 SamplePre
H ₈	$\mathbf{A}_0 \leftarrow \mathbb{Z}_q^{n \times m}$	↓	↓	Lemma 2 TrapGen
H ₉	↓	$\mathbf{c}_{3,j} \approx [\mathbf{c}_{1,j} \mid \mathbf{c}_{2,j}] \cdot \mathbf{K}_j - \mathbf{c}_0 \cdot \mathbf{U}_j$ $\mathbf{c}_5 \approx \mathbf{c}_0 \cdot \mathbf{U}$	↓	noise flooding
H ₁₀	↓	$\mathbf{c}_0 \leftarrow \mathbb{Z}_q^m, \mathbf{c}_4 \leftarrow \mathbb{Z}_q^\lambda$	↓	LWE _{n,m+λ,χ,q}
H ₁₁	$(\mathbf{A}_0, \mathbf{T}_{\mathbf{A}_0}) \leftarrow \text{TrapGen}(1^n, 1^m, q)$	↓	↓	Lemma 2 TrapGen
H ₁₂	↓	↓	$\mathbf{K}_y \leftarrow \text{SamplePre} \left(\begin{bmatrix} [\mathbf{A}_0] \mathbf{A}_y [\mathbf{B}_1 - \ell \cdot \mathbf{G}] \\ \mathbf{T}_{\mathbf{A}_0} \\ \mathbf{0}_{m \times m} \\ \mathbf{0}_{m \times m} \end{bmatrix}, \mathbf{D}, \tau_2 \right)$	Lemma 2 SamplePre
H ₁₃	↓	$\mathbf{c}_{3,j} \leftarrow \mathbb{Z}_q^m$	$\mathbf{A}_j \leftarrow \mathbb{Z}_q^{n \times m}$	LHL

Fig. 4. Summary of our security hybrids. ↓ denotes the same as the previous hybrid. We omit the noise terms in H₀. Starting from H₁₀, the proof is essentially analogous to that in [AFV11].

- H₁₃: replace $\mathbf{A}_j, \mathbf{c}_{3,j}$ with random.² We have H₁₂ \approx_s H₁₃ via the leftover hash lemma as follows. First, in H₁₂, \mathbf{K}_y does not leak any additional information about \mathbf{U}_j beyond $\mathbf{A}_1, \dots, \mathbf{A}_{\ell^*}$. Then, the leftover hash lemma tells us

$$(\mathbf{A}_0, \mathbf{c}_0, \mathbf{c}_0 \cdot \mathbf{U}_1, \dots, \mathbf{c}_0 \cdot \mathbf{U}_{\ell^*}, \mathbf{A}_0 \cdot \mathbf{U}_0, \dots, \mathbf{A}_0 \cdot \mathbf{U}_{\ell^*})$$

is statistically close to random. This means:

$$(\mathbf{A}_0, \mathbf{c}_0, \dots, \underbrace{[\mathbf{c}_{1,j} \mid \mathbf{c}_{2,j}] \cdot \mathbf{K}_j - \mathbf{c}_0 \cdot \mathbf{U}_j}_{\mathbf{c}_{3,j}}, \dots, \underbrace{\mathbf{A}_0 \cdot \mathbf{U}_j + x_{b,j}^* \cdot \mathbf{G}}_{\mathbf{A}_j}, \dots)$$

is statistically close to random.

² Similar argument shows that \mathbf{c}_5 is also pseudorandom.

Finally, observe that in H_{13} , the view of the adversary is statistically independent of the challenges \mathbf{x}_b^* . The various hybrids are described in Fig. 4. This result is summarized in the following theorem.

Theorem 5 (Security). *Let Π' be the inner product predicate encryption scheme described in Sect. 4.1, with parameters set as in Eq. (3) with $B = 1$, and F a PRF. Then, for any semi-adaptive adversary \mathcal{A} that runs in time $T = T(\lambda)$, there exists adversaries \mathcal{B}_0 , \mathcal{B}_1 and \mathcal{B}_2 against PRF-security, $\text{LWE}_{n,m,\chi,q}$, and $\text{LWE}_{n,m+\lambda,\chi,q}$ respectively, such that*

$$\text{Adv}_{\mathcal{A},\Pi}^{PE}(\lambda) \leq T \cdot \left(\text{Adv}_{\mathcal{B}_0,F}^{PRF}(\lambda) + \text{Adv}_{\mathcal{B}_1}^{\text{LWE}_{n,m,\chi,q}}(\lambda) + \text{Adv}_{\mathcal{B}_2}^{\text{LWE}_{n,m+\lambda,\chi,q}}(\lambda) + \text{negl}(\lambda) \right).$$

References

- ABB10. hweta Agrawal, Dan Boneh, and Xavier Boyen. Efficient lattice (H)IBE in the standard model. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 553–572. Springer, Heidelberg, May / June 2010.
- AFV11. Shweta Agrawal, David Mandell Freeman, and Vinod Vaikuntanathan. Functional encryption for inner product predicates from learning with errors. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 21–40. Springer, Heidelberg, December 2011.
- Agr17. Shweta Agrawal. Stronger security for reusable garbled circuits, general definitions and attacks. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 3–35. Springer, Heidelberg, 2017.
- Ajt96. Miklós Ajtai. Generating hard instances of lattice problems (extended abstract). In *28th ACM STOC*, pages 99–108. ACM Press, May 1996.
- Att14. Nuttapong Attrapadung. Dual system encryption via doubly selective security: Framework, fully secure functional encryption for regular languages, and more. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 557–577. Springer, Heidelberg, 2014.
- Att16. Nuttapong Attrapadung. Dual system encryption framework in primeorder groups via computational pair encodings. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part II*, volume 10032 of *LNCS*, pages 591–623. Springer, Heidelberg, December 2016.
- BCTW16. Zvika Brakerski, David Cash, Rotem Tsabary, and Hoeteck Wee. Targeted homomorphic attribute-based encryption. In Martin Hirt and Adam D. Smith, editors, *TCC 2016-B, Part II*, volume 9986 of *LNCS*, pages 330–360. Springer, Heidelberg, October / November 2016.
- BGG⁺14. Dan Boneh, Craig Gentry, Sergey Gorbunov, Shai Halevi, Valeria Nikolaenko, Gil Segev, Vinod Vaikuntanathan, and Dhinakaran Vinayagamurthy. Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 533–556. Springer, Heidelberg, 2014.
- BV16. Zvika Brakerski and Vinod Vaikuntanathan. Circuit-ABE from LWE: Unbounded attributes and semi-adaptive security. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part III*, volume 9816 of *LNCS*, pages 363–384. Springer, Heidelberg, 2016.

- CGKW18. Jie Chen, Junqing Gong, Lucas Kowalczyk, and Hoeteck Wee. Unbounded ABE via bilinear entropy expansion, revisited. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part I*, volume 10820 of *LNCS*, pages 503–534. Springer, Heidelberg, April / May 2018.
- CHKP10. David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. Bonsai trees, or how to delegate a lattice basis. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 523–552. Springer, Heidelberg, May / June 2010.
- CW14. Jie Chen and Hoeteck Wee. Semi-adaptive attribute-based encryption and improved delegation for Boolean formula. In Michel Abdalla and Roberto De Prisco, editors, *SCN 14*, volume 8642 of *LNCS*, pages 277–297. Springer, Heidelberg, 2014.
- DRS04. Yevgeniy Dodis, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 523–540. Springer, Heidelberg, 2004.
- GKW16. Rishab Goyal, Venkata Koppula, and Brent Waters. Semi-adaptive security and bundling functionalities made generic and easy. In Martin Hirt and Adam D. Smith, editors, *TCC 2016-B, Part II*, volume 9986 of *LNCS*, pages 361–388. Springer, Heidelberg, October / November 2016.
- GPSW06. Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM CCS 2006*, pages 89–98. ACM Press, October / November 2006. Available as Cryptology ePrint Archive Report 2006/309.
- GPV08. Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 197–206. ACM Press, May 2008.
- GSW13. Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 75–92. Springer, Heidelberg, 2013.
- GVW13. Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Attribute-based encryption for circuits. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 545–554. ACM Press, June 2013.
- HILL99. Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999.
- KL15. Lucas Kowalczyk and Allison Bishop Lewko. Bilinear entropy expansion from the decisional linear assumption. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 524–541. Springer, Heidelberg, August 2015.
- KSW08. Jonathan Katz, Amit Sahai, and Brent Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In Nigel P. Smart, editor, *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 146–162. Springer, Heidelberg, 2008.

- Lew12. Allison B. Lewko. Tools for simulating features of composite order bilinear groups in the prime order setting. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 318–335. Springer, Heidelberg, 2012.
- LW11. Allison B. Lewko and Brent Waters. Unbounded HIBE and attribute-based encryption. In Kenneth G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 547–567. Springer, Heidelberg, 2011.
- MP12. Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 700–718. Springer, Heidelberg, 2012.
- OT12. Tatsuaki Okamoto and Katsuyuki Takashima. Fully secure unbounded inner-product and attribute-based encryption. In Xiaoyun Wang and Kazue Sako, editors, *ASIACRYPT 2012*, volume 7658 of *LNCS*, pages 349–366. Springer, Heidelberg, December 2012.
- RW13. Yannis Rouselakis and Brent Waters. Practical constructions and new proof methods for large universe attribute-based encryption. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *ACM CCS 2013*, pages 463–474. ACM Press, November 2013.
- SW05. Amit Sahai and Brent R. Waters. Fuzzy identity-based encryption. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 457–473. Springer, Heidelberg, 2005.
- Ver23. Tanya Verma. Inside geo key manager v2: re-imagining access control for distributed systems. <https://blog.cloudflare.com/inside-geo-key-managerv2/>, 2023.
- Yam16. Shota Yamada. Adaptively secure identity-based encryption from lattices with asymptotically shorter public parameters. In Marc Fischlin and Jean-Sebastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 32–62. Springer, Heidelberg, May 2016.



Partially Non-interactive Two-Round Lattice-Based Threshold Signatures

Rutchathon Chairattana-Apirom^(✉) , Stefano Tessaro^(ID) , and Chenzhi Zhu^(ID) 

Paul G. Allen School of Computer Science & Engineering,
University of Washington, Seattle, USA
{rchairat, tessaro, zhucz20}@cs.washington.edu

Abstract. This paper gives the first lattice-based two-round threshold signature based on standard lattice assumptions for which the first message is independent of the message being signed without relying on fully-homomorphic encryption, and our construction supports arbitrary thresholds.

Our construction provides a careful instantiation of a generic threshold signature construction by Tessaro and Zhu (EUROCRYPT '23) based on specific linear hash functions, which in turns can be seen as a generalization of the FROST scheme by Komlo and Goldberg (SAC '20). Our reduction techniques are new in the context of lattice-based cryptography. Also, our scheme does not use any heavy tools, such as NIZKs or homomorphic trapdoor commitments.

1 Introduction

Multiple novel applications, primarily motivated by blockchains (e.g., digital wallets [39]), are re-energizing a multi-decade agenda aimed at developing practical threshold signatures [32, 33] with the goal of reducing trust assumptions in systems using digital signatures. To this end, recall that in a *t-out-of-n threshold signature scheme*, a set of n signers each hold shares of a secret signing key associated with a public verification key. Any subset of at least t of these signers should be able to come together and run a signing protocol to produce a signature on any message. However, an adversary that controls an arbitrary subset of fewer than t signers should not be able, on its own, to come up with a valid signature, even when they maliciously deviate from the protocol.

Threshold signatures are currently the focus of standardization efforts by NIST [58] and IETF [23], and threshold signing protocols for a number of existing signature schemes have been given from a variety of cryptographic assumptions. These include threshold versions of BLS [4, 12], Schnorr [7, 22, 26, 41, 51, 52, 66] and (EC-)DSA [13, 19, 38–40, 42, 53], along with several schemes for ad-hoc signatures in pairing-free groups with specific properties [5, 27, 68]. Several RSA-based constructions [28, 31, 43, 65, 68] have also been proposed.

LATTICE-BASED THRESHOLD SIGNATURES. With the threat of quantum computers looming on the horizon (and, in particular, their ability to break all assumptions behind all aforementioned threshold signatures), a widely recognized goal

is to develop threshold signatures that are based on quantum-safe assumptions. The most natural candidate for such schemes are lattice-based assumptions, considering in particular the fact that NIST has selected DILITHIUM [56] and FALCON [61], two lattice-based signature schemes, for standardization. Regardless of quantum safety, it is also important to obtain constructions from a set of assumptions as diverse as possible.

While lattice-based cryptography has been enormously successful in enabling extremely sophisticated functionalities, building efficient lattice-based threshold signatures has turned out to be very challenging. In principle, the problem can be solved generically and round optimally with constructions [2, 14, 15] based on *Fully-Homomorphic Encryption* (FHE), but these require the homomorphic evaluation of the signing algorithm *within* the FHE, thus imposing a substantial computational and communication overhead on the signing process.

There have been attempts [20, 29] at giving more direct constructions of two-round signing protocols based on the Fiat-Shamir-with-abort paradigm [54], obtained by adapting constructions for the related notion of multi-signatures. These constructions only realize n -out-of- n threshold signatures, i.e., do not tolerate arbitrary thresholds $t < n$. Gur, Katz, and Silde [45] recently proposed a new two-round construction based on linearly homomorphic encryption (LHE) which supports arbitrary thresholds. Both rounds are message-dependent, and they rely on homomorphic trapdoor commitments and NIZKs to ensure security against malicious signers. For $n = 5$ and $t = 3$, their signatures and public keys have sizes 46.6 and 13.6 KB, respectively, whereas the communication costs for signing are roughly 3 MB per signer. Recent work by del Pino *et al.* [60] proposes a more efficient lattice-based threshold signature scheme that does not rely on FHE or the aforementioned heavy primitives, but the drawback is that the protocol has three message-dependent rounds. Other recent works consider adaptive security [49] and robustness [36], but their schemes require higher round complexity. In Table 1, we provide an overview of the aforementioned lattice-based threshold signatures, detailing round complexity and assumptions used in each construction. We further discuss a concurrent and independent work by Espitau *et al.* [35] below.

BETTER TWO-ROUND THRESHOLD SIGNATURES. In this paper, we pursue the question of designing better and more efficient two-round threshold signatures. Clearly, we would like to minimize communication along with signature and key sizes, but other properties are desirable. For example, a fundamental property of FROST [7, 51] is that it is *partially* non-interactive, in that while the signing protocol consists of two rounds, the first round messages are simply nonces *independent* of the message being signed. This allows us to recover some of the positive features of non-interactive schemes by preprocessing the initial round. Currently, with the exception of FHE-based schemes, we do not know of any partially non-interactive lattice-based threshold signatures. Note that in fact partially non-interactive lattice-based multi-signatures exist [17], inspired by the discrete-log based counterparts [59], but it is not clear how to turn these into threshold signatures, especially for the case $t < n$.

Table 1. Overview of lattice-based t -out-of- n threshold signatures for arbitrary thresholds with the number of online (message-dependent) and offline (message-independent) rounds, hardness assumptions, and notes on specific details in each construction.

Scheme	Offline	Online	Assumption	Notes
ASY [2]	0	1	SIS + LWE	FHE
GKS [45]	0	2	RSIS + RLWE	Trapdoor Commitments + NIZKs
TRaccoon [60]	0	3	MSIS + MLWE	
KRT [49]	2	3	MSIS + MLWE	Adaptive security
Pelican [36]	0	4	MLWE	Robustness when $t \leq n/3$
EKT [35]	1	1	AOM-MLWE	Non-standard assumption
Our work	1	1	MSIS	

OUR CONTRIBUTIONS. In this paper, we develop the first partially non-interactive lattice-based threshold signatures that do not rely on FHE or other heavy primitives like NIZKs and trapdoor commitments. The security of our scheme is based on standard lattice assumptions, in particular, we rely on the Module-SIS assumption.

To achieve 128-bit of security and allow for up to 2^{64} signatures to be generated with the same key, for the case $n = 5$, which is the same setting considered by [45], the signatures in our scheme have sizes roughly of 258.1 KB with the size of public keys 47.6 KB, and the communication complexity per signer is 1.5 MB. While our signature and public key sizes are larger than [45], we achieve better communication complexity.

Like other recent works [5, 7, 26, 60], we do not propose an explicit distributed key generation (DKG) protocol. (We can envision that keys are either set up manually, or that they are the output of a suitable generic MPC protocol.) We leave the design of suitable DKG protocols as an interesting open question.

OUR APPROACH. A common way to construct an efficient lattice-based primitive is to take an efficient construction based on pairing-free groups and translate it into a lattice-based scheme. However, one key barrier in translating ideas from FROST, the state-of-art group-based partially non-interactive threshold signature scheme, to the lattice setting is that the security analysis of FROST relies on the one-more discrete logarithm assumption, of which no analog is known in the lattice setting. A recent work by Tessaro and Zhu [68] proposes a variant of FROST based on linear hash functions (LHF) and gives a security reduction to the plain DL assumption. Inspired by the work of Hauck et al. [46], which turns a LHF-based blind signature scheme into a lattice-based one, our starting point is to translate the LHF-based threshold signatures into lattice-based threshold signatures. The main difficulty in this idea is that the lattice-based linear hash functions do not have the desirable algebraic properties as required in the original analysis from [68]. We refer to the technical overview below for the detailed issues and our solutions.

We also want to point out some caveats with our work:

- Our approach is very different from the aforementioned lattice-based threshold signature works, and techniques, such as modularising the proof by reducing to the self-target MSIS problem [34, 50] or the one from [55] for reducing the norm bound of secret keys, do not apply to our construction. We refer to the last paragraph of the technical overview for more details.
- Our solution requires stronger properties from the underlying secret sharing scheme, which are satisfied by the secret sharing scheme by Benaloh and Leichter [9]. However, this makes our secret key shares significantly large and we address this further in Sect. 4.4. We remark that Benaloh-Leichter and similar kinds of small-coefficient linear secret sharing have also been used before in the context of lattice-based threshold cryptography, e.g., in constructing universal thresholdizers [14, 15, 18, 21]. We further discuss in Sect. 3.3 the issues which make other secret sharing schemes, such as the one by Applebaum et al. [3], not applicable to our use case.

SIGNIFICANCE OF THE WORK. We emphasize that we see the primary value of our paper in showing the feasibility of constructing partially non-interactive threshold signatures based on standard lattice assumptions without using FHE and giving new techniques involved in transforming a DL-based schemes into a lattice-based one. Nonetheless, we note that the efficiency of our schemes is still within the practical realm and deserves further investigation.

CONCURRENT AND INDEPENDENT WORK. Espitau et al. [35] recently proposed a lattice-based two-round partially non-interactive threshold signatures. Their construction also follows the approach of instantiating FROST in the lattice-based settings. However, the difference is that their security analysis is based on a non-standard interactive assumption, the Algebraic One-More Module Learning with Error (AOM-MLWE) assumption, which is newly proposed in their paper.

OTHER RELATED WORKS. We discuss some additional related works we have not discussed above. An alternative approach to obtain threshold signatures is to leverage standard MPC techniques to evaluate (part of the) signing. For example, Bendlin *et al.* [10] use this approach to obtain a threshold version of GPV signatures [44]. More recently, Cozzo and Smart [24] considered more broadly MPC-based instantiations of NIST post-quantum signature candidates and concluded that they are unlikely to lead to practical solutions.

1.1 Technical Overview

Our starting point is a variant of FROST [51] proposed by [68] which gives a threshold signature scheme based solely on the DL assumption, instead of the stronger one-more DL assumption. The key idea is to replace the map $x \mapsto g^x$ (for a generator g) in FROST with a *compressing* and *collision resistant linear* map $F : \mathfrak{D} \rightarrow \mathfrak{R}$, referred to as a linear hash function (LHF), where \mathfrak{D} and \mathfrak{R} are two vector spaces over a scalar field \mathfrak{S} . The secret key of the scheme is a random element $\text{sk} \in \mathfrak{D}$ and the corresponding public key is $\text{pk} \leftarrow F(\text{sk})$. The secret key shares $\{\text{sk}_i\}_{i \in [n]}$ are generated using Shamir’s secret sharing. The signing protocol consists of one offline round and one online round.

- In the offline round, each signer i samples $r_{i,0}, r_{i,1} \in \mathfrak{D}$ and publishes a token $(R_{i,0}, R_{i,1}) \leftarrow (F(r_{i,0}), F(r_{i,1}))$.
- In the online round, to sign a message μ , the user selects a set of signers SS of size at least t and sends a request $lr \leftarrow (\mu, SS, \{R_{i,0}, R_{i,1}\}_{i \in SS})$ to each signer in SS . Each signer i sends $R \leftarrow \sum_{i' \in SS} (R_{i',0} + bR_{i',1})$ with $b \leftarrow H_1(\text{pk}, lr)$, and $z_i \leftarrow r_{i,0} + br_{i,1} + c\lambda_i^{SS}\text{sk}_i$ with $c \leftarrow H_2(\text{pk}, \mu, R)$ to the user, where H_1 and H_2 are two hash functions.
- Finally, the signature is computed as $(R, z = \sum_{i \in SS} z_i)$. To verify it, one checks whether $F(z) = R + c \cdot \text{pk}$ for $c = H_2(\text{pk}, \mu, R)$.

Here $H_1, H_2 : \{0, 1\}^* \rightarrow \mathfrak{S}$ are hash functions. We note that the underlying signature scheme can be viewed as a LHF-based analog of Schnorr signatures. Also, the required properties of F are:

- (i) Linearity: $F(a) + F(b) = F(a + b)$ holds for any $a, b \in \mathfrak{D}$.
- (ii) Collision resistance: it is hard to find $x \neq y \in \mathfrak{D}$ such that $F(x) = F(y)$ for a randomly sampled F .
- (iii) Compressing: the pre-image of any element in \mathfrak{R} under F contains multiple elements.

As observed by Hauck et al. [46], a natural candidate to instantiate LHF from lattices is $F(\mathbf{x}) = A\mathbf{x}$, where A is a uniformly random matrix $A \in R_q^{k \times m}$ for a prime q and the ring $R_q := \mathbb{Z}_q[X]/(X^N + 1)$, with $\mathfrak{D} = \{\mathbf{x} \in R_q^m \mid \|\mathbf{x}\|_\infty \leq \beta_x\}$, $\mathfrak{R} = R_q^k$, and $\mathfrak{S} = R_q$, where $\beta_x < q$ is a constant. It is clear that F is linear and compressing if $|\mathfrak{D}| = (2\beta_x)^{mN} \gg q^{kN} = |\mathfrak{R}|$. Also, F is collision resistance under the Module-SIS (MSIS) assumption, which guarantees that given a uniform matrix $A \in R_q^{k \times m}$, it must be infeasible to find a small-norm solution $\mathbf{x} \neq \mathbf{0}$ such that $A\mathbf{x} = \mathbf{0}$. If one can find $\mathbf{x}_1 \neq \mathbf{x}_2 \in \mathfrak{D}$ such that $F(\mathbf{x}_1) = F(\mathbf{x}_2)$, we have $A(\mathbf{x}_1 - \mathbf{x}_2) = \mathbf{0}$, which gives us a MSIS solution $(\mathbf{x}_1 - \mathbf{x}_2)$ for A with ℓ_∞ -norm bounded by $2\beta_x$.

Unfortunately, we cannot simply apply the analysis from [68] to the above lattice-based instantiation. A simple reason is that \mathfrak{D} as defined above is not a linear space,¹ which are required by the prior analysis. There are also more technical reasons why this does not work, and to see what they are, we now try to apply the prior analysis here.

REDUCTION IDEA FROM PRIOR WORK. The reduction idea is simple. Denote an adversary that breaks unforgeability of the threshold signature scheme as \mathcal{A} , which corrupts up to $t - 1$ signers, engages in an arbitrary number of signing sessions with honest signers, and forges a valid signature for a message that was not signed in any of the signing sessions. We construct a MSIS adversary \mathcal{B} as follows: (In the analysis, H_1 and H_2 are modeled as random oracles.) Initially, \mathcal{B} receives a MSIS challenge A . Then, \mathcal{B} runs \mathcal{A} by simulating the key generation, the signing sessions and the random oracles following the protocol by itself. If \mathcal{A} returns a valid message-signature pair $(\mu^*, \text{sig}^* = (\mathbf{R}^*, \mathbf{z}^*))$, \mathcal{B} rewinds \mathcal{A} to the

¹ This is because given $\mathbf{x}_1, \mathbf{x}_2$ with ℓ_∞ -norm bounded by β_x , $\|\mathbf{x}_1 + \mathbf{x}_2\|_\infty$ can exceed β_x .

step that the query $H_2(\text{pk}, \mu^*, \mathbf{R}^*)$ is made and runs \mathcal{A} again while answering its random oracle queries with refreshed randomness. If \mathcal{A} returns $(\bar{\mu}^*, \overline{sig}^* = (\bar{\mathbf{R}}^*, \bar{z}^*))^2$ with $(\mu^*, \mathbf{R}^*) = (\bar{\mu}^*, \bar{\mathbf{R}}^*)$, then we find a collision $F(\mathbf{z}^* - c \cdot \text{sk}) = \mathbf{R}^* = \bar{\mathbf{R}}^* = F(\bar{\mathbf{z}}^* - \bar{c} \cdot \text{sk})$, where c and \bar{c} are the outputs of $H_2(\text{pk}, \mu^*, \mathbf{R}^*)$ in the first and second execution respectively. Therefore, \mathcal{B} returns $(\mathbf{z}^* - \bar{\mathbf{z}}^* - (c - \bar{c}) \cdot \text{sk})$. Otherwise, \mathcal{B} aborts.

By the Forking Lemma, if \mathcal{A} breaks unforgeability with high probability, then we have that \mathcal{B} does not abort and $c \neq \bar{c}$ with high probability. The difficulty here is to ensure that we indeed find a MSIS solution, i.e. $\mathbf{z}^* - \bar{\mathbf{z}}^* - (c - \bar{c}) \cdot \text{sk} \neq 0$. The prior analysis from [68] shows that for any two secret keys $\text{sk} \neq \text{sk}'$ mapping to the same public key, there exists a bijection Φ that maps the randomness ρ of \mathcal{B} to another randomness ρ' such that ρ and ρ' lead to secret key sk and sk' respectively, and the view of \mathcal{A} given ρ is identical to that given ρ' . Therefore, \mathcal{A} outputs the same $(\mu^*, \mathbf{R}^*, \mathbf{z}^*, \bar{\mu}^*, \bar{\mathbf{R}}^*, \bar{z}^*)$ independent of whether \mathcal{B} is run with ρ or ρ' . Since $\text{sk} \neq \text{sk}'$ and $c \neq \bar{c}$, we have that $\mathbf{z}^* - \bar{\mathbf{z}}^* - (c - \bar{c}) \cdot \text{sk} \neq \mathbf{z}^* - \bar{\mathbf{z}}^* - (c - \bar{c}) \cdot \text{sk}'$, so \mathcal{B} wins in at least one of the cases. Hence, \mathcal{B} wins with at least half of the probability that \mathcal{B} does not abort.

CHALLENGES IN LATTICE INSTANTIATIONS. The main challenges lie in how to construct Φ . Note that given the secret key sk , the randomness ρ of \mathcal{B} consists of: an RO tape $\mathbf{h} = (h_{1,1}, h_{1,2}, \dots, h_{q_h,1}, h_{q_h,2}, \bar{h}_{1,1}, \dots, \bar{h}_{q_h,2})$, where $h_{i,j}$ is used to answer the i -th RO query to H_j in the first execution of \mathcal{A} and $\bar{h}_{i,j}$ is used after rewinding, the secret key shares $\{\text{sk}_i\}_{i \in [n]}$ of sk ,³ and the randomness $(\mathbf{r}_{i,0}, \mathbf{r}_{i,1})$ for generating the tokens of each signing session. Therefore, we only consider Φ defined over those variables. First of all, Φ maps \mathbf{h} to itself since \mathcal{A} can learn \mathbf{h} from RO queries. For the other two parts, Φ satisfies the following:

- (1) Φ maps $\{\text{sk}_i\}_{i \in [n]}$ to $\{\text{sk}'_i\}_{i \in [n]}$ such that $\{\text{sk}'_i\}_{i \in [n]}$ are the secret shares of sk' and $\text{sk}_i = \text{sk}'_i$ for any corrupted signer i .
- (2) For the interaction with signer i during signing, Φ maps $(\mathbf{r}_{i,0}, \mathbf{r}_{i,1})$ to $(\mathbf{r}'_{i,0}, \mathbf{r}'_{i,1})$ such that $F(\mathbf{r}_{i,0}) = F(\mathbf{r}'_{i,0})$, $F(\mathbf{r}_{i,1}) = F(\mathbf{r}'_{i,1})$, and

$$\begin{pmatrix} 1 & b \\ 1 & \bar{b} \end{pmatrix} \begin{pmatrix} \mathbf{r}_{i,0} \\ \mathbf{r}_{i,1} \end{pmatrix} + \begin{pmatrix} c\lambda_i^{SS} \text{sk}_i \\ \bar{c}\lambda_i^{SS} \text{sk}_i \end{pmatrix} = \begin{pmatrix} \mathbf{z}_i \\ \bar{\mathbf{z}}_i \end{pmatrix} = \begin{pmatrix} 1 & b \\ 1 & \bar{b} \end{pmatrix} \begin{pmatrix} \mathbf{r}'_{i,0} \\ \mathbf{r}'_{i,1} \end{pmatrix} + \begin{pmatrix} c\lambda_i^{SS} \text{sk}'_i \\ \bar{c}\lambda_i^{SS} \text{sk}'_i \end{pmatrix},$$

where we use $(\bar{\cdot})$ to denote the variables after rewinding. (It is possible that the adversary makes only one query or the same queries for the token during the two executions, but these cases are easier to deal with. Thus, we only discuss the above hardest case here.)

It is not hard to satisfy the first condition due to the privacy property of secret sharing. For the second condition, by the idea of prior work, if $b - \bar{b}$ is invertible, we can set $(\mathbf{r}'_{i,0}, \mathbf{r}'_{i,1}) = (\mathbf{r}_{i,0} + (c - b(b - \bar{b})^{-1} \Delta_c) \Delta_{\text{sk}}, \mathbf{r}_{i,1} + (b - \bar{b})^{-1} \Delta_c \Delta_{\text{sk}})$ to

² In this section, we will use the overline to denote values in the second run.

³ More accurately, it should be the randomness for generating the secret key shares, but for simplicity of explanation, we use the secret key shares instead.

make the condition hold, where $\Delta_c = c - \bar{c}$ and $\Delta_{sk} = \lambda_i^{SS}(\mathbf{sk}_i - \mathbf{sk}'_i)$. However, the problem is that the map is not a bijection since \mathfrak{D} is not a vector space. There is no guarantee that $(\mathbf{r}'_{i,0}, \mathbf{r}'_{i,1}) \in \mathfrak{D}$ for $\mathbf{r}_{i,0}, \mathbf{r}_{i,1} \in \mathfrak{D}$. A common solution, which was also used by Hauck et al. [46], is to enlarge \mathfrak{D} (by increasing β_x) such that $(\mathbf{r}'_{i,0}, \dots, \mathbf{r}'_{i,\ell}) \in \mathfrak{D}$ except for a negligible fraction of $(\mathbf{r}_{i,0}, \dots, \mathbf{r}_{i,\ell})$. Still, there are two issues we need to address: (a) We need to show that the shift $(b - \bar{b})^{-1} \Delta_c \Delta_{sk}$ is small; (b) To make the fraction of bad randomness negligible, we have to set $\beta_x = \Omega(2^\kappa \|(b - \bar{b})^{-1} \Delta_c \Delta_{sk}\|)$, where κ denotes the security parameter. This would lead to a very large modulus.

OUR SOLUTION. For issue (a), we need to show that all of the three parts, i.e., $(b - \bar{b})^{-1}$, Δ_c , and Δ_{sk} , are small. To make sure that the inverse of $(b - \bar{b})^{-1}$ is small, we restrict the range of H_1 to be $\{0, 1\}$. As a result, with $1/2$ probability, $b - \bar{b} \in \{1, -1\}$ and thus its inverse is small (either 1 or -1). Then, we boost the probability to $1 - 2^{-2\kappa}$ by increasing the number of nonces and the range of H_1 to be $\{0, 1\}^{2\kappa}$. More precisely, in the offline round, each signer i samples $\mathbf{r}_{i,0}, \mathbf{r}_{i,1}, \dots, \mathbf{r}_{i,\ell}$ for $\ell = 2\kappa$. In the online round, signer i returns $\mathbf{z}_i \leftarrow \mathbf{r}_{i,0} + \sum_{j \in [\ell]} b_j \mathbf{r}_{i,j} + c \lambda_i^{SS} \mathbf{sk}_i$, where $(b_1, \dots, b_\ell) \in \{0, 1\}^\ell$ are computed from H_1 . Also, Φ maps $(\mathbf{r}_{i,0}, \dots, \mathbf{r}_{i,\ell})$ to $(\mathbf{r}'_{i,0}, \dots, \mathbf{r}'_{i,\ell}) = (\mathbf{r}_{i,0} + (c - b_j(b_j - \bar{b}_j)^{-1} \Delta_c) \Delta_{sk}, \dots, \mathbf{r}_{i,j-1}, \mathbf{r}_{i,j} + (b_j - \bar{b}_j)^{-1} \Delta_c \Delta_{sk}, \mathbf{r}_{i,j+1}, \dots, \mathbf{r}_{i,\ell})$, where j is the first index with $b_j \neq \bar{b}_j$.

For Δ_c , it is a common practice to sample c with small ℓ_1 -norm, which implies that the norm of Δ_c is small. Lastly, we have to ensure that the norm of Δ_{sk} is small. This imposes an additional requirement on the secret sharing scheme. Namely, it requires that there exists a map Φ satisfying the aforementioned condition (1) and in addition, restricting $\|\mathbf{sk}_i - \mathbf{sk}'_i\|_\infty$ to be small. We show that a special class of secret sharing schemes, referred to as linear secret sharing schemes with small coefficients, satisfies the requirement. We refer to Sect. 3 for the detailed definition and instantiation.

To address issue (b), we sample each $\mathbf{r}_{i,j}$ from an m -dimensional discrete Gaussian distribution centered at the origin with variance σ_r . Intuitively, \mathfrak{D} becomes a probability distribution instead of a set, and we can show that \mathcal{B} wins with high probability as long as the ratio $\alpha = \frac{\Pr[(\mathbf{r}_{i,0}, \dots, \mathbf{r}_{i,\ell})]}{\Pr[\Phi(\mathbf{r}_{i,0}, \dots, \mathbf{r}_{i,\ell})]}$ is close to 1 except for a negligible fraction of $(\mathbf{r}_{i,0}, \dots, \mathbf{r}_{i,\ell})$. More precisely, we need to show $\alpha^{q_s} \in (1 - \varepsilon, 1 + \varepsilon)$ for some constant ε , where q_s denotes the number of signing sessions. Since the map only shifts $\mathbf{r}_{i,0}$ and $\mathbf{r}_{i,j}$ by roughly $\Delta = \Delta_c \Delta_{sk}$, the ratio is roughly $\exp\left(\frac{\|\Delta\|^2 + 2\|\Delta\| \cdot \|(\mathbf{r}_{i,0}, \dots, \mathbf{r}_{i,\ell})\|}{\sigma_r^2}\right)$, and we can achieve the desired bound by setting $\sigma_r = \Omega(q_s \|\Delta_c \Delta_{sk}\|)$.

We now discuss the two optimizations made to improve the efficiency of our protocol in the following paragraphs.

DECREASING THE NUMBER OF NONCES. In the above protocol, the number of nonces generated is equal to the security parameter, resulting in significant overhead in communication complexity. To decrease the number of nonces ℓ , the key observation is that we can extend the domain of b to the set of signed

monomials, $\mathcal{S}_b := \{\pm 1, \pm X, \dots, \pm X^{N-1}\} \subseteq R_q$. (This has also been considered in other works, e.g., [11].) Although for any $b \neq \bar{b} \in \mathcal{S}_b$, $(b - \bar{b})$ does not necessarily have a small inverse, we can show that there exists $v_{b-\bar{b}} \in R$ such that $v_{b-\bar{b}}(b - \bar{b}) = 2$ and $\|v_{b-\bar{b}}\|_\infty \leq 1$. Therefore, we let each signer compute $\mathbf{z}_i \leftarrow \mathbf{r}_{i,0} + \sum_{j \in [\ell]} b_j \mathbf{r}_{i,j} + 2c \cdot \lambda_i^{\text{SS}} \text{sk}_i$, and, then we can structure the map Φ following the method described above except that we replace $(b - \bar{b})^{-1}$ with $v_{b-\bar{b}}$. As a result, we just need to set $\ell = 2\kappa / \log(2N)$, which is 10 times smaller for $N = 512$ used in our concrete efficiency analysis.

IMPROVING MODULUS SIZE USING THE RÉNYI DIVERGENCE. Another main efficiency problem is that the modulus size depends linearly on \mathbf{q}_s , which is implied by how we set σ_r . To address this, we observe that the ratio $\frac{\Pr[(\mathbf{r}_{i,0}, \dots, \mathbf{r}_{i,\ell})]}{\Pr[(\mathbf{r}'_{i,0}, \dots, \mathbf{r}'_{i,\ell})]}$ is not evenly distributed. It gets larger as the norm of $\mathbf{r}_{i,j}$ becomes larger. However, as the norm of $\mathbf{r}_{i,j}$ becomes larger, its probability of being sampled becomes exponentially small. As a result, there are only a small fraction of points with ratios close to the ratio bound, while a large proportion of points have much smaller ratios. Therefore, we try to use the Rényi divergence, which computes the *average* of the probability ratio of two distributions. More precisely, instead of considering the probability that a particular random value $(\mathbf{r}_{i,0}, \dots, \mathbf{r}_{i,\ell})$ is sampled, we consider the distribution of the view of \mathcal{A} conditioning on sk (denoted by T_{sk}) directly. We show that \mathcal{B} wins with high probability as long as the Rényi divergence $R_\alpha(T_{\text{sk}} \| T_{\text{sk}'})$ is close to 1. Then, we observe that the Rényi divergence of the view of \mathcal{A} in a *single* signing session given sk from that given sk' is roughly the Rényi divergence of two discrete Gaussian distributions both with variance $O(\sigma_r)$ and with distance $\|\Delta_c \Delta_{\text{sk}}\|$ between their centers. Thus, considering all signing sessions (both before and after the rewinding), $R_\alpha(T_{\text{sk}} \| T_{\text{sk}'})$ is roughly $\exp(\mathbf{q}_s \|\Delta_c \Delta_{\text{sk}}\|^2 / \sigma_r^2)$, omitting the constants and unimportant factors. Therefore, we can set $\sigma_r = \Omega(\sqrt{\mathbf{q}_s} \|\Delta_c \Delta_{\text{sk}}\|)$, improving the modulus size by a factor of $\sqrt{\mathbf{q}_s}$. We note that similar techniques have also been used by Agrawal et al. [2] to improve the modulus size of their FHE-based threshold signature.

TECHNICAL DISTINCTIONS FROM OTHER WORKS. We emphasize that our proof framework is very different from other recent lattice-based threshold and multi-signature works [17, 20, 29, 35, 60] without using homomorphic encryptions, where unforgeability of the protocols is reduced to the key-only security of the underlying signature schemes by showing that the signing oracles can be simulated given only the public key. In contrast, our proof framework directly simulates the unforgeability game using the honestly sampled secret key and then extract an MSIS solution by rewinding. The benefit here is that we do not need to rely on more advanced assumptions [35], introduce an additional online round [20, 60], or use hard cryptographic primitives, such as a homomorphic trapdoor commitment scheme [17, 29]. However, the downside is that techniques, such as reducing to the self-target MSIS problem [34, 50] or the common trick to reduce the norm bound of secret keys using MLWE [17, 29, 55], do not apply to our construction.

2 Preliminaries

2.1 Notations

For any integers $0 \leq k < n$, $[n]$ denotes $\{1, \dots, n\}$, and $[k..n]$ denotes $\{k, \dots, n\}$. We use κ to denote the security parameter. For a finite set S , $|S|$ denotes the size of S , and $x \leftarrow_s S$ denotes sampling an element uniformly from S and assigning it to x . For a distribution \mathcal{D} , $x \leftarrow_s \mathcal{D}$ denotes sampling x according to \mathcal{D} . For a sequence of variables x_1, \dots, x_ℓ , we use $x_{[i..j]}$ to denote (x_i, \dots, x_j) . For any vector space V over a field F and a set $S \in V$, we denote $\text{Span}_F(S)$ as the F -span of S , which is the smallest F -subspace of V that contains S . In particular, we omit F from the subscript if $F = \mathbb{R}$. For a finite set $S = \{v_1, \dots, v_n\} \subseteq V$, we say S is F -linearly independent if and only if for any non-zero $(a_1, \dots, a_n) \in F^n$, $\sum_{i \in [n]} a_i v_i \neq 0$. We say S is a F -basis of V if and only if S is F -linearly independent and $\text{Span}_F(S) = V$. When F is not specified, we assume $F = \mathbb{R}$. The dimension of V is equal to the size of S .

2.2 Polynomial Rings

Let q be an odd prime and N be a power of 2. We denote the ring $R := \mathbb{Z}[X]/(X^N + 1)$, contained in the cyclotomic field $K := \mathbb{Q}[X]/(X^N + 1)$, and let $R_q := R/qR \cong \mathbb{Z}_q[X]/(X^N + 1)$. Denote $K_{\mathbb{R}} := \mathbb{R} \otimes K \cong \mathbb{R}[X]/(X^N + 1)$. For an element $v \in K_{\mathbb{R}}$, where $v = \sum_{i=0}^{N-1} v_i X^i$, we denote its conjugate as $v^* = \sum_{i=0}^{N-1} -v_i X^{N-i}$. We use ϕ to denote the coefficient embedding that embeds $K_{\mathbb{R}}$ in \mathbb{R}^N , and ϕ maps v to vector $(v_0, \dots, v_{N-1}) \in \mathbb{R}^N$. When applying ϕ to a vector $\mathbf{v} \in K_{\mathbb{R}}^m$, ϕ maps \mathbf{v} to a vector in \mathbb{R}^{mN} by applying ϕ to each entry of \mathbf{v} . The map ϕ is a bijection, and we denote its inverse by ϕ^{-1} . An ℓ_p -norm of $\mathbf{v} \in K_{\mathbb{R}}^m$ is given by $\|\mathbf{v}\|_p := \|\phi(\mathbf{v})\|_p = \left(\sum_{i=1}^m \sum_{j=0}^{N-1} |v_{i,j}|^p \right)^{\frac{1}{p}}$, where $v_{i,j}$ denotes the coefficient of X^j of the i -th entry of \mathbf{v} . Additionally, the ℓ_∞ -norm of \mathbf{v} is defined as $\|\mathbf{v}\|_\infty := \max_{i \in [m], j \in [0..N-1]} |v_{i,j}|$. For the ℓ_2 -norm, we omit the subscript and denote $\|\mathbf{v}\|$ as the ℓ_2 -norm of \mathbf{v} . Denote the conjugate transpose of $\mathbf{v} \in K_{\mathbb{R}}^m$ as $\mathbf{v}^\dagger := (\mathbf{v}^*)^T$. We define the inner product of two vectors $\mathbf{v}, \mathbf{v}' \in K_{\mathbb{R}}^m$ as $\langle \mathbf{v}, \mathbf{v}' \rangle := \phi(\mathbf{v})^T \phi(\mathbf{v}') = \langle \phi(\mathbf{v}), \phi(\mathbf{v}') \rangle$. We have $\|\mathbf{v}\| = \langle \mathbf{v}, \mathbf{v} \rangle$. We say \mathbf{v} is a unit vector if $\|\mathbf{v}\| = 1$.

Also, we define a map ϕ_M that maps each element in $K_{\mathbb{R}}$ to a matrix in $\mathbb{R}^{N \times N}$ as follows. Let $M_X := \begin{pmatrix} \mathbf{0} & -1 \\ I_{N-1} & \mathbf{0} \end{pmatrix} \in \mathbb{R}^N$, where I_{N-1} is the identity matrix in \mathbb{R}^{N-1} . For each $v \in K_{\mathbb{R}}$, $\phi_M(v) := \sum_{i=0}^{N-1} v_i M_X^i$, which can be viewed as the matrix representation of v . In particular, for ϕ and ϕ_M , the following properties hold: for any $v, v' \in K_{\mathbb{R}}$, $\phi_M(v^*) = \phi_M(v)^T$, $\phi_M(vv') = \phi_M(v)\phi_M(v')$ and $\phi_M(v)\phi(v') = \phi(vv')$. We extend the above definitions to R_q by representing each $v \in R_q$ as $v = \sum_{i=0}^{N-1} v_i X^i$, where $v_i \in \{-(q-1)/2, \dots, (q-1)/2\}$.

For a matrix $M \in K_{\mathbb{R}}^{m \times m}$, we denote its conjugate transpose as $M^\dagger = (M^*)^T$, and we say M is *hermitian* if $M = M^\dagger$. We say M is *positive definite* if and

only if M is hermitian and for all $\mathbf{x} \in K_{\mathbb{R}}^m \setminus \{\mathbf{0}\}$, $\langle \mathbf{x}, M\mathbf{x} \rangle > 0$, or equivalently, $\phi_M(M)$ is positive definite. Also, denote $\sigma_{\min}(M) := \inf_{\mathbf{x} \in K_{\mathbb{R}}^m, \|\mathbf{x}\|=1} \langle \mathbf{x}, M\mathbf{x} \rangle$ as the smallest singular value of M and $\sigma_{\max}(M) := \sup_{\mathbf{x} \in K_{\mathbb{R}}^m, \|\mathbf{x}\|=1} \langle \mathbf{x}, M\mathbf{x} \rangle$ as the largest singular value of M .

We state the following lemma establishing the property of the set of signed monomials $\mathcal{S}_b := \{\pm 1, \dots, \pm X^{N-1}\} \subseteq R_q$, used in the security analysis.

Lemma 1. *Let $\mathcal{S}_b := \{\pm 1, \dots, \pm X^{N-1}\} \subseteq R_q$. For any $b, \bar{b} \in \mathcal{S}_b$ such that $b \neq \bar{b}$, the ideal generated by $b - \bar{b}$ contains 2.*

Proof. Let $b = sX^a, \bar{b} = \bar{s}X^{\bar{a}}$ for $a, \bar{a} \in [0..N-1]$ and $s, \bar{s} \in \{-1, 1\}$. Consider two cases:

- $a = \bar{a}$: Then, $b - \bar{b} = 2X^a$ or $-2X^a$. It is easy to see that the statement holds as $(b - \bar{b})X^{N-a} = 2$ or -2 .
- $a \neq \bar{a}$: W.l.o.g. assume $a > \bar{a}$. Then, $b - \bar{b}$ generates $X^{a-\bar{a}} - s\bar{s}$ since $(b - \bar{b}) \cdot (-sX^{N-\bar{a}}) = X^{a-\bar{a}} + s\bar{s}$. We can see that this generates $X^{2^e(a-\bar{a})} - 1$ for any $e \geq 1$, since $(x-1)(x+1) = x^2 - 1$. Since $a - \bar{a} < N$ and N is a power of 2, there exists e such that $N \mid 2^e(a-\bar{a})$ but $N \nmid 2^{e-1}(a-\bar{a})$. Then, $2^e(a-\bar{a}) = Na'$ for some odd a' , and thus $b - \bar{b}$ generates $X^{Na'} - 1 = (-1)^{a'} - 1 = -2$, which implies the statement. \square

2.3 Lattices and Discrete Gaussian Distributions

In this subsection, we give definitions for lattices and discrete Gaussian distributions over \mathbb{R} and $K_{\mathbb{R}}$. An m -dimensional lattice Λ over \mathbb{Z} (resp. R) is a discrete additive subgroup of \mathbb{Z} (resp. R). Equivalently, $\Lambda = \mathcal{L}(\{\mathbf{b}_1, \dots, \mathbf{b}_k\}) := \{\sum_{i \in [k]} x_i \mathbf{b}_i : x_i \in \mathbb{Z}\}$ for a set of linearly independent vectors $\mathbf{b}_1, \dots, \mathbf{b}_k \in \mathbb{Z}^m$ (resp. R^m), which is referred to as a basis of Λ . The size k is the *rank* of the lattice Λ . We say Λ is a *full rank* lattice if $k = m$ (resp. $k = mN$ for Λ over R). For any $a \in \mathbb{Z}^m$ (resp. R^m), $\Lambda + a$ is a *coset* of Λ . The *dual lattice* of Λ is denoted as $\Lambda^* = \{\mathbf{x} \in \text{Span}(\Lambda) : \forall \mathbf{y} \in \Lambda, \langle \mathbf{x}, \mathbf{y} \rangle \in \mathbb{Z}\}$. A Λ -subspace is the linear span of some subset of Λ , i.e., a subspace S such that $S = \text{Span}(S \cap \Lambda)$. For any two vectors $\mathbf{v} \in \mathbb{Z}^m$ (resp. R^m) and $\mathbf{u} \in \mathbb{Z}^n$ (resp. R^n), denote $\mathbf{v} \otimes \mathbf{u} := (v_1 u_1, \dots, v_1 u_n, \dots, v_m u_1, \dots, v_m u_n) \in \mathbb{Z}^{mn}$ (resp. R^{mn}). For any two lattices $\Lambda \subseteq \mathbb{Z}^m$ (resp. R^m) and $\Lambda' \subseteq \mathbb{Z}^n$ (resp. R^n), denote their tensor product as $\Lambda \otimes \Lambda'$, which is the smallest lattice over \mathbb{Z}^{mn} (resp. R^{mn}) that contains $\{\mathbf{x} \otimes \mathbf{y} : \mathbf{x} \in \Lambda, \mathbf{y} \in \Lambda'\}$.

Further, for a lattice $\Lambda \subseteq R^m$, we say Λ is a R -lattice if and only if Λ is a R -module, or equivalently, $r\mathbf{x} \in \Lambda$ for any $r \in R$ and $\mathbf{x} \in \Lambda$. For a matrix $A \in R_q^{k \times m}$, we define the R -lattice $\Lambda_q^\perp(A) \subseteq R^m$ as $\Lambda_q^\perp(A) := \{\mathbf{x} \in R^m : A\mathbf{x} = \mathbf{0} \pmod{q}\}$, which is full-rank since $qR^m \subseteq \Lambda_q^\perp(A)$.

For a positive definite matrix $\Sigma \in \mathbb{R}^{m \times m}$ (resp. an invertible positive definite matrix $\Sigma \in K_{\mathbb{R}}^{m \times m}$) and a vector $\mathbf{c} \in \mathbb{R}^n$ (resp. $K_{\mathbb{R}}^m$), we define the function $\rho_{\Sigma, \mathbf{c}}$ over \mathbb{R}^m (resp. $K_{\mathbb{R}}^m$) as $\rho_{\Sigma, \mathbf{c}}(\mathbf{x}) := \exp(-\pi \langle \mathbf{x} - \mathbf{c}, \Sigma^{-1}(\mathbf{x} - \mathbf{c}) \rangle)$.

$$\mathcal{D}_{\Lambda + \mathbf{a}, \Sigma, \mathbf{c}}^m(\mathbf{x}) := \Pr[\mathbf{x} \leftarrow \mathcal{D}_{\Lambda + \mathbf{a}, \Sigma, \mathbf{c}}^m] = \frac{\rho_{\Sigma, \mathbf{c}}(\mathbf{x})}{\rho_{\Sigma, \mathbf{c}}(\Lambda + \mathbf{a})}$$

where $\rho_{\Sigma, \mathbf{c}}(\Lambda + \mathbf{a}) = \sum_{\mathbf{x} \in \Lambda + \mathbf{a}} \rho_{\Sigma, \mathbf{c}}(\mathbf{x})$. For $\Lambda + \mathbf{a} \subseteq R^m$, we denote $\mathcal{D}_{\Lambda + \mathbf{a}, \Sigma, \mathbf{c}}^{m, \text{mod } q}(\mathbf{x})$ as the distribution of $(\mathbf{x} \bmod q) \in R_q^m$ for \mathbf{x} sampled from $\mathcal{D}_{\Lambda + \mathbf{a}, \Sigma, \mathbf{c}}^m$.

The following lemma shows that a discrete Gaussian distribution over $K_{\mathbb{R}}$ can be viewed as a discrete Gaussian distribution over \mathbb{R} via the coefficient embedding ϕ .

Lemma 2. *For a random variable $\mathbf{x} \in K_{\mathbb{R}}^m$, the distribution of \mathbf{x} is $\mathcal{D}_{\Lambda + \mathbf{a}, \Sigma, \mathbf{c}}^m$ for some lattice coset $\Lambda + \mathbf{a} \subseteq R^m$, an invertible positive definite matrix $\Sigma \in K_{\mathbb{R}}^{m \times m}$, and vector $\mathbf{c} \in K_{\mathbb{R}}^m$ if and only if the distribution of $\phi(\mathbf{x})$ is $\mathcal{D}_{\phi(\Lambda + \mathbf{a}), \phi_M(\Sigma), \phi(\mathbf{c})}^{mN}$.*

Proof. For any $\mathbf{v} \in K_{\mathbb{R}}^m$,

$$\begin{aligned} \rho_{\phi_M(\Sigma), \phi(\mathbf{c})}(\phi(\mathbf{v})) &= \exp(-\pi \langle \phi(\mathbf{v} - \mathbf{c}), \phi_M(\Sigma)^{-1} \phi(\mathbf{v} - \mathbf{c}) \rangle) \\ &= \exp(-\pi \langle \mathbf{v} - \mathbf{c}, \Sigma^{-1}(\mathbf{v} - \mathbf{c}) \rangle) \\ &= \rho_{\Sigma, \mathbf{c}}(\mathbf{v}). \end{aligned}$$

Therefore, for any $\mathbf{x} \in \Lambda + \mathbf{a}$, $\mathcal{D}_{\Lambda + \mathbf{a}, \Sigma, \mathbf{c}}^m(\mathbf{x}) = \mathcal{D}_{\phi(\Lambda + \mathbf{a}), \phi_M(\Sigma), \phi(\mathbf{c})}^{mN}(\phi(\mathbf{x}))$. □

Also, we make some remarks about the notations we will use throughout the paper. When $\Sigma = \sigma^2 \mathbb{I}_m$ for $\sigma \in \mathbb{R}$, we will use $\rho_{\sigma, \mathbf{c}}$ and $\mathcal{D}_{\Lambda + \mathbf{a}, \sigma, \mathbf{c}}^m$ as $\rho_{\Sigma, \mathbf{c}}$ and $\mathcal{D}_{\Lambda + \mathbf{a}, \Sigma, \mathbf{c}}^m$, respectively. If the center $\mathbf{c} = 0$, then we omit the subscript \mathbf{c} from $\rho_{\Sigma, \mathbf{c}}$ and $\mathcal{D}_{\Lambda + \mathbf{a}, \Sigma, \mathbf{c}}^m$. Moreover, when $\Lambda + \mathbf{a} = \mathbb{Z}^m$ (resp. $\Lambda + \mathbf{a} = R^m$), we omit $\Lambda + \mathbf{a}$ from the subscript of $\mathcal{D}_{\Lambda + \mathbf{a}, \Sigma, \mathbf{c}}^m$.

The smoothing parameter of a lattice Λ with respect to $\varepsilon > 0$, denoted by $\eta_{\varepsilon}(\Lambda)$, is the smallest $s > 0$ such that $\rho_{1/s}(\Lambda^* \setminus \{0\}) \leq \varepsilon$. Throughout the paper, we set $\varepsilon = 2^{-2\kappa}$.

We borrow the following lemma from [1] that bounds the ℓ_2 -norm of discrete Gaussian random variables and adapt it to lattices over $K_{\mathbb{R}}$ by Lemma 2.

Lemma 3 (Lemma 3 of [1] adapted to $K_{\mathbb{R}}$). *For any $\varepsilon \in (0, 1)$, a lattice $\Lambda \subseteq R^m$, $\mathbf{c} \in K_{\mathbb{R}}^m$, and $\sigma \geq \eta_{\varepsilon}(\Lambda)$, then $\Pr[\|\mathbf{x} - \mathbf{c}\| \geq \sigma\sqrt{mN} : \mathbf{x} \leftarrow \mathcal{D}_{\Lambda, \sigma, \mathbf{c}}] \leq \frac{1+\varepsilon}{1-\varepsilon} \cdot 2^{-mN}$.*

We also borrow the following lemma from [17] to bound the smoothing parameters of $\Lambda_q^{\perp}(A)$ for a randomly sampled A .

Lemma 4 (Lemma 2.5 of [17]). *Let q be an odd integer and A a uniformly random matrix in $R_q^{k \times m}$, $k < m$. Then, for any $\varepsilon > 0$, except with probability at most 2^{-N} on the choice of A , we have $\eta_{\varepsilon}(\Lambda_q^{\perp}(A)) \leq \frac{8q \frac{k}{m}}{\sqrt{\pi}} \sqrt{N \log(2mN(1 + 1/\varepsilon))}$.*

2.4 Assumptions

We recall the module short integer solution (MSIS) problem (defined in Fig. 1). The advantage of \mathcal{A} for the MSIS problem is defined as $\text{Adv}_{q, k, m, \beta}^{\text{msis}}(\mathcal{A}) := \Pr[\text{MSIS}_{q, k, m, \beta}^{\mathcal{A}} = 1]$.

Game $\text{MSIS}_{q,k,m,\beta}^A$: <hr style="border: 0; border-top: 1px solid black; margin: 2px 0;"/> $A \leftarrow_{\$} R_q^{k \times m}$; $\mathbf{x} \leftarrow \mathcal{A}(A)$ Return $(\mathbf{x} \neq 0 \wedge \ \mathbf{x}\ \leq \beta \wedge A\mathbf{x} = 0)$
--

Fig. 1. The module-SIS problem.

2.5 Rényi Divergence

We define the notion of Rényi Divergence [62] between two distributions P, Q which we will use in the analysis of the scheme. For a discrete distribution P , we denote the support of P as $\text{Supp}(P) := \{x : P(x) > 0\}$.

Definition 1 (Rényi Divergence). *Let P, Q be two discrete probability distributions such that $\text{Supp}(P) \subseteq \text{Supp}(Q)$ and $\alpha \in [1, +\infty]$. We define the Rényi*

Divergence of order α , for $\alpha \in (1, \infty)$ as $R_\alpha(P\|Q) := \left(\sum_{x \in \text{Supp}(P)} \frac{P(x)^\alpha}{Q(x)^{\alpha-1}} \right)^{\frac{1}{\alpha-1}}$.

For $\alpha = 1$ and $\alpha = \infty$, we define $R_1(P\|Q) := \exp\left(\sum_{x \in \text{Supp}(P)} P(x) \log \frac{P(x)}{Q(x)}\right)$,

$R_\infty(P\|Q) := \max_{x \in \text{Supp}(P)} \frac{P(x)}{Q(x)}$.

The two following lemmas, from [2] and [64] respectively, give basic properties of the Rényi Divergence.

Lemma 5 (Lemma 2.27 of [2]). *Let $\alpha \in [1, \infty]$ and P, Q be discrete probability distributions with $\text{Supp}(P) \subseteq \text{Supp}(Q)$. Then, the following properties hold:*

- **Log Positivity:** $R_\alpha(P\|Q) \geq R_\alpha(P\|P) = 1$.
- **Data Processing Inequality:** $R_\alpha(P^f\|Q^f) \leq R_\alpha(P\|Q)$ for any function f , where P^f (and Q^f) denotes the distribution which samples $x \leftarrow_{\$} P$ ($x \leftarrow_{\$} Q$) and outputs $f(x)$.
- **Probability Preservation:** Let $E \subseteq \text{Supp}(Q)$ be an arbitrary event. Then, for $\alpha \in (1, \infty)$, $\Pr_{x \leftarrow_{\$} Q}[E] \geq \Pr_{x \leftarrow_{\$} P}[E]^{\alpha/(\alpha-1)} / R_\alpha(P\|Q)$.
- **Weak Triangle Inequality:** Let P_1, P_2, P_3 be three probability distributions where $\text{Supp}(P_1) \subseteq \text{Supp}(P_2) \subseteq \text{Supp}(P_3)$. Then, we have

$$R_\alpha(P_1\|P_3) \leq \begin{cases} R_\alpha(P_1\|P_2) \cdot R_\infty(P_2\|P_3) \\ R_\infty(P_1\|P_2)^{\frac{\alpha}{\alpha-1}} \cdot R_\alpha(P_2\|P_3) & \text{if } \alpha \in (1, \infty) \end{cases}$$

Lemma 6 (Proposition 2 of [64]). *Let P and Q denote two distributions of a sequence of random variables (X_1, \dots, X_n) . For $1 \leq i \leq n$, denote $P_{i|x_{[i-1]}}$ (resp. $Q_{i|x_{[i-1]}}$) as the conditional distribution of X_i given $X_{[i-1]} = x_{[i-1]}$. Then, for any $\alpha > 1$, $R_\alpha(P\|Q) \leq \prod_{i \in [n]} \max_{x_{[i-1]}} R_\alpha(P_{i|x_{[i-1]}}\|Q_{i|x_{[i-1]}})$.*

The following lemma from [67] upperbounds the Rényi Divergence between two discrete Gaussian distributions with different centers.

Lemma 7 (Lemma 5 of [67]). For any m -dimensional lattice $\Lambda \subseteq \mathbb{Z}^m$, a positive definite $\Sigma \in \mathbb{R}^{m \times m}$, and two vectors $\mathbf{c}, \mathbf{c}' \in \mathbb{R}^m$, let $P = \mathcal{D}_{\Lambda, \Sigma, \mathbf{c}}^m$ and $Q = \mathcal{D}_{\Lambda, \Sigma, \mathbf{c}'}^m$. If $\mathbf{c}, \mathbf{c}' \in \Lambda$, set $\varepsilon = 0$. Otherwise, fix $\varepsilon \in (0, 1)$ and assume $\sqrt{\sigma_{\min}(\Sigma)} \geq \eta_\varepsilon(\Lambda)$ with $\sigma_{\min}(\Sigma) := \inf_{x \in \mathbb{R}^m, \|x\|=1} \|\Sigma x\|$ denoting the smallest singular value of Σ . Then, $R_\alpha(P\|Q) \leq \left(\frac{1+\varepsilon}{1-\varepsilon}\right)^{\frac{\alpha}{\alpha-1}} \exp\left(\alpha\pi \frac{\|\mathbf{c}-\mathbf{c}'\|^2}{\sigma_{\min}(\Sigma)}\right)$.

By Lemma 2, we derive the following lemma, which adapts the above to lattices over rings. The proof is deferred to the full version of the paper.

Lemma 8. For any m -dimensional lattice coset $\Lambda + \mathbf{a} \subseteq R^m$ and any integer $q > 0$, an invertible positive definite $\Sigma \in K_{\mathbb{R}}^{m \times m}$, and two vectors $\mathbf{c}, \mathbf{c}' \in R_q^m$, let $P = \mathcal{D}_{\Lambda+\mathbf{a}, \Sigma, \mathbf{c}}^{m, \text{mod } q}$ and $Q = \mathcal{D}_{\Lambda+\mathbf{a}, \Sigma, \mathbf{c}'}^{m, \text{mod } q}$. If $\mathbf{c}, \mathbf{c}' \in \Lambda + \mathbf{a}$, set $\varepsilon = 0$. Otherwise, fix $\varepsilon \in (0, 1)$ and assume $\sqrt{\sigma_{\min}(\Sigma)} \geq \eta_\varepsilon(\Lambda)$. Then,

$$R_\alpha(P\|Q) \leq \left(\frac{1+\varepsilon}{1-\varepsilon}\right)^{\frac{\alpha}{\alpha-1}} \exp\left(\alpha\pi \frac{\|\mathbf{c}-\mathbf{c}'\|^2}{\sigma_{\min}(\Sigma)}\right).$$

2.6 Linear Transformations of Discrete Gaussian Random Variables

We adopt the notation $P \stackrel{\varepsilon}{\approx} Q$ from [37]: for any two distributions P, Q with the same support and $\varepsilon > 0$, we say that $P \stackrel{\varepsilon}{\approx} Q$ if and only if $\max_{x \in \text{Supp}(P)} |\log P(x) - \log Q(x)| \leq \log(1 + \varepsilon)$, or equivalently, $\max(R_\infty(P\|Q), R_\infty(Q\|P)) \leq 1 + \varepsilon$. Note that if $P \stackrel{\varepsilon}{\approx} Q$, then the statistical distance between P and Q is bounded by $\varepsilon/2$, i.e., $\frac{1}{2} \sum_{x \in \text{Supp}(P)} |P(x) - Q(x)| \leq \varepsilon/2$. We state the following lemma used later in our security proof, which characterizes the distribution of a linear transformation of independent discrete Gaussian random variables. The full proof is deferred to the full version of the paper.

Lemma 9. For any constant $\varepsilon \in (0, 1)$, $\sigma_0 > 0$, full-rank R -lattice $\Lambda \subseteq R^m$ with $\eta_\varepsilon(\Lambda) \leq \sigma_0/(2\sqrt{3mN})$, arbitrary elements $\mathbf{s}_0, \mathbf{s}_1, \dots, \mathbf{s}_\ell \in R^m$ and $b_1, \bar{b}_1, \dots, b_\ell, \bar{b}_\ell \in \mathcal{S}_b$ (defined in Lemma 1) such that $(b_1, \dots, b_\ell) \neq (\bar{b}_1, \dots, \bar{b}_\ell)$, let $\mathbf{r}_0, \mathbf{r}_1, \dots, \mathbf{r}_\ell$ be independent samples with $\mathbf{r}_i \leftarrow_s \mathcal{D}_{\Lambda+\mathbf{s}_i, \sigma_0}^m$, and $T = \begin{pmatrix} 1 & b_1 & \dots & b_\ell \\ 1 & \bar{b}_1 & \dots & \bar{b}_\ell \end{pmatrix}$ and $(\mathbf{y}, \bar{\mathbf{y}}) = (T \otimes \mathbb{I}_m) \cdot (\mathbf{r}_0, \dots, \mathbf{r}_\ell) \in R^{2m}$. Denote the joint distribution of $(\mathbf{y}, \bar{\mathbf{y}})$ as D . Then,

$$D \stackrel{\varepsilon'}{\approx} \mathcal{D}_{(T \otimes \mathbb{I}_m)\Lambda^{\ell+1} + (\mathbf{S}, \bar{\mathbf{S}}), \Sigma \otimes \mathbb{I}_m}^{2m},$$

where $\varepsilon' = \frac{2((1+\varepsilon)^\ell - 1)}{2 - (1+\varepsilon)^\ell}$, $\Lambda^{\ell+1} := \{(\mathbf{x}_0, \dots, \mathbf{x}_\ell) : \forall i \in [0..\ell], \mathbf{x}_i \in \Lambda\}$, which is a $(\ell + 1)m$ -dimensional lattice over R , $(\mathbf{S}, \bar{\mathbf{S}}) = (T \otimes \mathbb{I}_m) \cdot (\mathbf{s}_0, \dots, \mathbf{s}_\ell)$, and $\Sigma = \sigma_0^2 T T^\dagger \in K_{\mathbb{R}}^{2 \times 2}$ is invertible and positive definite.

Moreover, denote D_1 as the marginal distribution of \mathbf{y} and $D_{2|\mathbf{y}_0}$ as the distribution of $\bar{\mathbf{y}}$ conditioning on $\mathbf{y} = \mathbf{y}_0$ for any $\mathbf{y}_0 \in \Lambda + \mathbf{S}$, and we have

$$D_1 \stackrel{\varepsilon'}{\approx} \mathcal{D}_{\Lambda+\mathbf{S}, \sigma=\sqrt{\Sigma_{11}}}^m, \quad D_{2|\mathbf{y}_0} \stackrel{\varepsilon'}{\approx} \mathcal{D}_{T \otimes \Lambda + \mathbf{y}_0 + \bar{\mathbf{S}} - \mathbf{S}, \frac{\Delta(\Sigma)}{\Sigma_{11}} \cdot \mathbb{I}_m, \frac{\Sigma_{12}}{\Sigma_{11}} \mathbf{y}_0}^m,$$

where $\mathcal{I} \subseteq R$ is the ideal generated by $b_1 - \bar{b}_1, \dots, b_\ell - \bar{b}_\ell$, Σ_{ij} denotes the entry in the i -th row and j -th column of Σ , $\Delta(\Sigma) = \Sigma_{11}\Sigma_{22} - \Sigma_{12}\Sigma_{21}$ denotes the determinant of Σ , and $\Sigma_{11} = \Sigma_{22} = \sigma_0^2(1 + \ell)$.

2.7 Other Useful Lemmas

We also show the following useful lemmas.

Lemma 10. *For any integer $n > 0$ and any $v \in R$ such that $\|v\|_1 \leq n$ and $v \neq n$,*

$$\sigma_{\max}((1 + v)^*(1 + v)) \leq (n + 1)^2 - 2n/N^2 .$$

The proof of the above lemma is deferred to the full version of the paper.

Lemma 11. *For any $a, b \geq 0$ such that $a + b \leq 1$ and $\alpha \geq 1$, we have $a + b^\alpha \geq \frac{1}{\alpha}(a + b)^\alpha$.*

Proof. Let $f(x) = x + b^\alpha$ and $g(x) = \frac{1}{\alpha}(x + b)^\alpha$. Since $f(0) = b^\alpha \geq \frac{1}{\alpha}b^\alpha = g(0)$ and $f'(x) = 1 \geq (x + b)^{\alpha-1} = g'(x)$ for $x \geq 0$, we know $f(x) \geq g(x)$ for $0 \leq x \leq 1 - b$, we have $f(x) \geq g(x)$ for $0 \leq x \leq 1 - b$, which shows the statement. \square

3 Linear Secret Sharing Schemes with Small Coefficients

In this section, we first define, in Sect. 3.1, the notion of linear threshold secret sharing schemes with small coefficients for an abelian group \mathbb{G} (which for our threshold signature $\mathbb{G} = R_q^m$ with its additions as the group operations) and discuss the properties required by our construction. Then, we consider a secret sharing scheme which satisfies the desired properties in Sect. 3.2 and discuss why other secret sharing schemes do not apply to our case in Sect. 3.3.

3.1 Definitions

We first give a brief explanation on the notations used in this section. We consider the group \mathbb{G} as a \mathbb{Z} -module and adopt the additive notation with 0 as the neutral element. Additionally, for a vector $\mathbf{g} \in \mathbb{G}^K$ and a matrix $M \in \mathbb{Z}^{L \times K}$, $M\mathbf{g}$ denotes $(\sum_{j=1}^K M_{1,j} \cdot g_j, \dots, \sum_{j=1}^K M_{L,j} \cdot g_j)^T \in \mathbb{G}^L$, and for $g \in \mathbb{G}$ and a vector $\mathbf{u} \in \mathbb{Z}^K$, $\mathbf{u} \cdot g$ denotes $(u_1 \cdot g, \dots, u_K \cdot g)^T$. Now, we give the following definition for linear threshold secret sharing schemes with small coefficients.

Definition 2 (Linear Threshold Secret Sharing with Small Coefficients). *Let $1 < t \leq n, L$, and K be positive integers and \mathbb{G} be an abelian group. A t -out-of- n linear threshold secret sharing scheme $\text{SecSha}_{t,n}$ for \mathbb{G} consists of two algorithms (Share, Recon) with the following syntax:*

- $\text{Share}(s \in \mathbb{G}; \boldsymbol{\rho} \in \mathbb{G}^K) \Rightarrow (\text{ss}_j)_{j \in [L]} \in \mathbb{G}^L$: takes as input a secret $s \in \mathbb{G}$ and a randomness vector $\boldsymbol{\rho} \in \mathbb{G}^K$ (sampled uniformly from \mathbb{G}^K), and returns the secret shares $(\text{ss}_j)_{j \in [L]}$. We note that each party $i \in [n]$ has a subset of indices $T_i \subseteq [L]$ such that the share of party i is $(\text{ss}_j)_{j \in T_i}$. We say that the individual share size of party i is $|T_i|$, the total share size is L , and the randomness size is K .
- $\text{Recon}(U, (\text{ss}_j)_{j \in \bigcup_{i \in U} T_i}) \Rightarrow s \in \mathbb{G}$: takes as input a set $U \subseteq [n]$ with $|U| \geq t$ and the secret shares corresponding to each party in U , and returns the reconstructed secret s .

We require that $\text{SecSha}_{t,n}$ satisfies the following properties:

- **Linearity:** The sharing algorithm Share can be written as an integer matrix $M \in \mathbb{Z}^{L \times (K+1)}$ mapping a vector $\mathbf{v} = (s, \rho_1, \dots, \rho_K)^T \in \mathbb{G}^{K+1}$ to $M\mathbf{v} \in \mathbb{G}^L$. Moreover, for any $U \subseteq [n]$ denote M_U as the matrix M restricted to the rows indexed with $\bigcup_{i \in U} T_i$, the following is also true:
 - For any $U \subseteq [n], |U| \geq t$, there exists a **reconstruction coefficient** vector $\boldsymbol{\lambda}^U \in \mathbb{Z}^L$ such that $\lambda_j^U = 0$ for $j \notin \bigcup_{i \in U} T_i$ and $(\boldsymbol{\lambda}^U)^T M = (1, 0, \dots, 0)$. Then, the output of $\text{Recon}(U, \cdot)$ on input $(\text{ss}_j)_{j \in \bigcup_{i \in U} T_i}$ can be written as $\sum_{i \in U} \sum_{j \in T_i} \lambda_j^U \text{ss}_j$. Hence, for $(\text{ss}_j)_{j \in [L]} \leftarrow \text{Share}(s; \boldsymbol{\rho})$ for any $s \in \mathbb{G}$ and $\boldsymbol{\rho} \in \mathbb{G}^K$, we have that $\sum_{i \in U} \sum_{j \in T_i} \lambda_j^U \text{ss}_j = s$.
 - For any $U \subseteq [n]$ with $|U| < t$, there exists a vector $\mathbf{u} \in \mathbb{Z}^{K+1}$ such that $u_1 = 1$ and $M_U \mathbf{u} = \mathbf{0}$. We call such \mathbf{u} the **sweeping vector** of M_U .
- **Small Coefficients:** For the sharing matrix M , its entries are bounded by β_M and the number of non-zero entries in each row is bounded by β_{row} . For any $U \subseteq [n]$ and $|U| \geq t$, the reconstruction coefficient vector $\boldsymbol{\lambda}^U$ has $\|\boldsymbol{\lambda}^U\|_\infty \leq \beta_\lambda$. For any $U \subseteq [n]$ and $|U| < t$, there exists a sweeping vector \mathbf{u} of M_U such that $\|\mathbf{u}\|_\infty \leq \beta_u$.

We point out that our definition differs from prior works in that we did not explicitly define correctness and privacy properties (since we will not use them in the proofs of our construction), and instead give two properties: linearity and small coefficients. The linearity property already implies correctness and privacy, as shown in prior works [6, 25, 48] which showed relations between linear secret sharing schemes and span programs. In particular, the first bullet point of linearity implies correctness, while the second bullet point implies privacy.

The small coefficients property is required by the following lemma, which establishes a crucial property used in the security proof of our threshold signature. Notably, fixing two secret keys $\text{sk}, \text{sk}' \in R_q^m$ with bounded norms and a corrupted subset $U \subseteq [n]$ with $|U| < t$, one can construct a bijection $\Phi_{\text{sk}, \text{sk}', U}$ between the set of the randomness used to generate the secret shares of sk and sk' such that: the secret shares given to the corrupted parties is unchanged (item (1)), and the distance between the reconstructed shares for any party is bounded (item (2)).

Lemma 12. *Let $(\text{Share}, \text{Recon})$ be a t -out-of- n linear threshold secret sharing with small coefficients for $\mathbb{G} = R_q^m$. In particular, let $M \in \mathbb{Z}^{L \times (K+1)}$ be the sharing matrix, and $\beta_M, \beta_{\text{row}}, \beta_\lambda, \beta_u$ be the bounds for the small coefficients property.*

Fix any $U \subseteq [n]$ with $|U| < t$, a matrix $A \in R_q^{k \times m}$ and any $\text{sk}, \text{sk}' \in R_q^m$ such that $A\text{sk} = A\text{sk}'$ and $\|\text{sk}\|_\infty, \|\text{sk}'\|_\infty \leq \beta_{\text{sk}}$. Then, there exists a bijection $\Phi_{\text{sk}, \text{sk}', U} : (R_q^m)^K \rightarrow (R_q^m)^K$, such that for any $\rho \in (R_q^m)^K$ and $\rho' = \Phi_{\text{sk}, \text{sk}', U}(\rho)$, the secret shares $(\text{ss}_j)_{j \in [L]} \leftarrow \text{Share}(\text{sk}; \rho)$ and $(\text{ss}'_j)_{j \in [L]} \leftarrow \text{Share}(\text{sk}'; \rho')$ satisfy:

- (1) $(\text{ss}_j)_{j \in \cup_{i \in U} T_i} = (\text{ss}'_j)_{j \in \cup_{i \in U} T_i}$
- (2) For any $S \subseteq [n]$ with $|S| \geq t$, let $\lambda^S \in \mathbb{Z}^L$ be the reconstruction coefficients for $\text{Recon}(S, \cdot)$. Also, for $i \in S$, define $\mathbf{v}_i = \sum_{j \in T_i} \lambda_j^S \text{ss}_j$ and $\mathbf{v}'_i = \sum_{j \in T_i} \lambda_j^S \text{ss}'_j$, we have that $A\mathbf{v}_i = A\mathbf{v}'_i$, and $\|\mathbf{v}_i - \mathbf{v}'_i\|_\infty \leq \beta_{\text{ss}}\beta_{\text{sk}}$, where $\beta_{\text{ss}} = 2|T_i|\beta_M\beta_{\text{row}}\beta_u\beta_\lambda$.

Proof. Let $\mathbf{u} \in \mathbb{Z}^{K+1}$ be a sweeping vector for M_U such that $\|\mathbf{u}\|_\infty \leq \beta_u$ which exists due to our secret sharing definition. Consider the map $\Phi_{\text{sk}, \text{sk}', U}$ defined as $\Phi_{\text{sk}, \text{sk}', U}(\rho) = \rho + (u_2, \dots, u_{K+1})^T \cdot (\text{sk}' - \text{sk})$, which we can see is a bijection on $(R_q^m)^K$ as it only shifts ρ by some fixed amount. Now, fix a $\rho \in (R_q^m)^K$ and $\rho' = \Phi_{\text{sk}, \text{sk}', U}(\rho)$. Consider the secret shares generated using these two randomness. For any $j \in [L]$, denote M_j as the j -th row of M , then $\text{ss}'_j - \text{ss}_j = M_j(\text{sk}', \rho'^T)^T - M_j(\text{sk}, \rho^T)^T = (M_j\mathbf{u}) \cdot (\text{sk}' - \text{sk})$.

Then, since $M_U\mathbf{u} = \mathbf{0}$, we have $(\text{ss}'_j)_{j \in \cup_{i \in U} T_i} = (\text{ss}_j)_{j \in \cup_{i \in U} T_i} + (M_U\mathbf{u}) \cdot (\text{sk}' - \text{sk}) = (\text{ss}_j)_{j \in \cup_{i \in U} T_i}$, proving (1). To show (2), for $i \in [n]$, consider \mathbf{v}_i and \mathbf{v}'_i as defined in the lemma statement. Then,

$$\begin{aligned} \mathbf{v}'_i - \mathbf{v}_i &= \sum_{j \in T_i} \lambda_j^S (\text{ss}'_j - \text{ss}_j) = \sum_{j \in T_i} \lambda_j^S (M_j\mathbf{u}) \cdot (\text{sk}' - \text{sk}). \end{aligned}$$

Since $\sum_{j \in T_i} \lambda_j^S (M_j\mathbf{u}) \in \mathbb{Z}$, we have that $A\mathbf{v}'_i - A\mathbf{v}_i = \left(\sum_{j \in T_i} \lambda_j^S (M_j\mathbf{u}) \right) \cdot (A\text{sk}' - A\text{sk}) = \mathbf{0} \in R_q^k$, so $A\mathbf{v}'_i = A\mathbf{v}_i$. Moreover, with $\beta_{\text{ss}} = 2|T_i|\beta_M\beta_{\text{row}}\beta_u\beta_\lambda$, $\|\mathbf{v}_i - \mathbf{v}'_i\|_\infty \leq \left\| \sum_{j \in T_i} \lambda_j^S (M_j\mathbf{u}) (\text{sk}' - \text{sk}) \right\|_\infty \leq \beta_{\text{ss}}\beta_{\text{sk}}$. \square

3.2 Instantiation

One secret sharing scheme satisfying Definition 2 is the generic construction from Benaloh and Leichter [9] which derives a linear secret sharing scheme for any monotone access structure (i.e., for any set S of parties that can recover the secret, any set that contains S can also recover the secret) from a monotone Boolean formula (i.e., a Boolean circuit with only AND and OR gates of fan-in 2 and fan-out 1, but the input wires may have multiple fan-out) f computing such access structure. Damgård and Thorbek [30] showed that Benaloh-Leichter secret sharing satisfies the following properties:

- (1) Both the number of randomness K and total share size L are at most the size of the formula f .
- (2) The sharing matrix M has binary entries, and the number of 1's in each row is at most the depth of f .
- (3) The reconstruction coefficients are in $\{-1, 0, 1\}$.
- (4) For any $U \subseteq [n]$ with $|U| < t$, the sweeping vector \mathbf{u} of M_U has entries in $\{-1, 0, 1\}$.

Regarding the formula computing threshold access structure, a seminal work by Valiant [69] gave a probabilistic construction of a monotone formula for majority function ($(n/2, n)$ -threshold function) of size $O(n^{5.3})$ and depth $5.3 \log n + O(1)$. Then, Boppana [16] generalized this result to a monotone formula for (t, n) -threshold function of size $O(t'^{4.3} n \log n)$ and depth $\log n + 4.3 \log t' + \log \log n + O(1)$ where $t' = \min(t, n - t)$. Hoory, Magen, and Pitassi [47] improved this to a monotone circuit of size $O(t'^2 n \log n)$ and depth $O(\log n)$. However, as pointed out in [18], this construction is not a formula (namely, the gates in this circuit have multiple fan-out), so it does not imply a linear secret sharing scheme. Also, it is worth noting that these are probabilistic constructions with success probability $1/2$ of realizing the threshold functions. Still, for small n (e.g., $n = 5, 32$ as we consider in this work), we can exhaustively check if a constructed formula correctly computes the threshold function on all inputs.

The following lemma then formalizes the existence of a secret sharing scheme constructed by applying Benaloh and Leichter’s construction to Boppana’s monotone formula for threshold function.

Lemma 13. *There exists a t -out-of- n linear threshold secret sharing with small coefficients with total share size $L = O(t'^{4.3} n \log n)$ making the individual share size $|T_i| \leq O(t'^{4.3} n \log n)$ for $t' = \min(t, n - t)$ and the small coefficient bounds $\beta_M = \beta_\lambda = \beta_u = 1$ and $\beta_{\text{row}} = \log n + 4.3 \log t' + \log \log n + O(1)$, which result in the bound β_{ss} from Lemma 12 of $\beta_{\text{ss}} = O(t'^{4.3} n (\log n)^2)$.*

3.3 Discussion on Other Secret Sharing Schemes

In this section, we discuss whether other linear secret sharing schemes, such as a recent ramp/near-threshold secret sharing scheme [3] and the tree secret sharing scheme [21], apply to our case.

Applebaum, Nir, and Pinkas [3] recently proposed a ramp/near-threshold black-box secret sharing scheme where a set of at least $t_c n$ parties is guaranteed to recover a secret, while privacy is guaranteed for any set of less than $t_p n$ corrupted parties with $0 < t_p < t_c < 1$. Their secret sharing scheme has the sharing matrix M of the form

$$M = \begin{pmatrix} 0^{L-1} & G \\ 1 & \mathbf{a}^T \end{pmatrix} \in \mathbb{Z}^{L \times (K+1)},$$

where $L, K = O(n)$, $G \in \mathbb{Z}^{(L-1) \times K}$ is a matrix with small entries, and each entry of $\mathbf{a} \in \mathbb{Z}^K$ is bounded by some constant c . We also remark that the secret share corresponding to the last row of M is public in their scheme. Their reconstruction can be modeled as a $O(n)$ -size $O(\log n)$ -depth addition circuit, translating to a bound of $\text{poly}(n)$ on the reconstruction coefficients.

For the sweeping vector, fixing a subset $U \subseteq [n]$ where $|U| < t_p n$ and letting M_U and G_U denote the rows of the matrices M and G of which the shares are known to U , they showed that there exists a vector $\mathbf{u}' \in \mathbb{Z}^K$ with each entry bounded by some constant b where $G_U \mathbf{u}' = \mathbf{0}$ and $v = \mathbf{a}^T \cdot \mathbf{u}' \neq 0 \pmod q$ for any prime $q > 2bcK$ (see Claim 4.1 of [3]). This gives us a vector $(v, -\mathbf{u}'^T)^T$

with $|v| \leq bcK$ such that $M_U(v, -\mathbf{u}^T)^T = 0$. However, since v is not necessarily 1, we only get a sweeping vector $\mathbf{u} = v^{-1}(v, -\mathbf{u}^T)^T \pmod q$ of which the entries are not guaranteed to be bounded, because v^{-1} can be large in \mathbb{Z}_q . To account for division by any v , one can scale up the secret by the least common multiple of $(1, \dots, bcK)$, but this scaling is estimated to be $2^{O(n)}$.⁴

Tree Secret Sharing Scheme is proposed by Cheon, Cho, and Kim [21] in the context of improving universal thresholdizer. They constructed a $(n + 1)/2$ -out-of- n linear secret sharing by repeatedly applying η -out-of- $(2\eta - 1)$ Shamir’s secret sharing, for any integer $\eta \geq 2$, in a tree structure. The tree is of depth $d \geq \log_{c_\eta} n + \log_\eta n + O(1)$ with $c_\eta = \frac{2\eta-2}{2^{2\eta-2}} \cdot \binom{2\eta-2}{s-1}$, and the total share size is $O(n^{\log_{c_\eta}(2\eta-1) + \log_\eta(2\eta-1)})$. They showed that their reconstruction coefficients are bounded by $((2\eta - 1)!)^{2d}$, amounting to $O(n^{2(\log_{3/2} 6 + \log_2 6)}) \approx O(n^{14})$ for $\eta = 2$. This already exceeds β_{ss} from Lemma 13, so we do not consider their secret sharing as an instantiation.

4 Threshold Signatures

In this section, we first give formal syntax and security definitions for threshold signatures, then present our construction and the security analysis, and finally discuss the concrete parameters and efficiency.

4.1 Syntax and Security

We use the formalization proposed by Bellare et al. [7], which is also used in [68].

SYNTAX. A (partially) non-interactive threshold signature schemes for n signers and threshold t is a tuple of efficient (randomized) algorithms $\text{TS} = (\text{Setup}, \text{KeyGen}, \text{SPP}, \text{LPP}, \text{LR}, \text{PS}, \text{Agg}, \text{Vf})$ that behave as follows. Signers involved are a leader and n signers. In real-world scenarios, the leader can be one of the signers. The setup algorithm $\text{Setup}(1^\kappa)$ initializes the state st_i for each signer $i \in [n]$ and st_0 for the leader and returns a system parameter par . We assume par is given to all other algorithms implicitly. The key generation algorithm $\text{KeyGen}()$ returns a public verification key pk , and a secret key sk_i for each signer i .

The signing protocol consists of two rounds: a message-independent offline round and an online signing round. In the offline round, any signer i can run $\text{SPP}(\text{st}_i)$ to generate a pre-processing token pp , which is sent to the leader, and the leader runs $\text{LPP}(i, pp, \text{st}_0)$ to update its state st_0 to incorporate token pp . In the online round, for any signer set $SS \subseteq [n]$ with size t and message $\mu \in \{0, 1\}^*$, the leader runs $\text{LR}(\mu, SS, \text{st}_0)$ to generate a leader request lr with $lr.\text{msg} = \mu$ and $lr.SS = SS$ and sends lr to each signer $i \in SS$. Then, each signer i runs $\text{PS}(lr, i, \text{st}_i)$ to generate its partial signature $psig_i$. Finally, the leader computes a signature sig for μ by running $\text{Agg}(\{psig_i\}_{i \in SS})$. The (deterministic)

⁴ The natural logarithm of $\text{LCM}(1, \dots, x)$ is the second Chebyshev’s function which is bounded by $1.03883x$ [63].

<p>Game $\text{TS-COR}_{\text{TS}}^A(\kappa)$:</p> <p>$par \leftarrow \text{Setup}(1^\kappa)$; $(pk, \{sk_i\}_{i \in [n]}) \leftarrow \text{KeyGen}()$ For $i \in [n]$ do $st_i.sk \leftarrow sk_i$; $st_i.pk \leftarrow pk$ $(\mu, SS) \leftarrow \mathcal{A}(par, pk, \{sk_i\}_{i \in [n]})$ If $SS \not\subseteq [n]$ or $SS < t$ then return 0 For $i \in SS$ do $(pp_i, st_i) \leftarrow \text{SPP}(st_i)$; $st_0 \leftarrow \text{LPP}(i, pp_i, st_0)$ $(lr, st_0) \leftarrow \text{LR}(\mu, SS, st_0)$ For $i \in SS$ do $(psig_i, st_i) \leftarrow \text{PS}(lr, i, st_i)$ $sig \leftarrow \text{Agg}(\{psig_i\}_{i \in SS})$ Return $\text{Vf}(pk, \mu, sig) = 0$</p>
--

Fig. 2. The TS-COR game for a threshold signature scheme TS with threshold t .

<p>Game $\text{TS-UF-0}_{\text{TS}}^A(\kappa)$:</p> <p>$par \leftarrow \text{Setup}(1^\kappa)$; $H \leftarrow \text{TS.HF}$; $S \leftarrow \emptyset$ $(\mu, sig) \leftarrow \mathcal{A}^{\text{INIT}, \text{PPO}, \text{PSIGNO}, \text{RO}}(par)$ Return $(\mu \notin S \wedge \text{Vf}(pk, \mu, sig) = 1)$</p> <p><u>Oracle INIT($CS$) :</u> Require: $CS \subseteq [n]$ and $CS < t$ $HS \leftarrow [n] \setminus CS$ $(pk, sk_1, \dots, sk_n) \leftarrow \text{KeyGen}()$ For $i \in HS$ do $st_i.sk \leftarrow sk_i$; $st_i.pk \leftarrow pk$ Return $(pk, \{sk_i\}_{i \in CS})$</p>	<p><u>Oracle PPO(i) :</u> Require: $i \in HS$ $(pp, st_i) \leftarrow \text{SPP}(st_i)$; $PP_i \leftarrow PP_i \cup \{pp\}$ Return pp</p> <p><u>Oracle PSIGNO(i, lr) :</u> Require: $lr.SS \subseteq [n]$ and $i \in HS$ $\mu \leftarrow lr.msg$; $S \leftarrow S \cup \{\mu\}$ $(psig, st_i) \leftarrow \text{PS}(lr, i, st_i)$ Return $psig$</p> <p><u>Oracle RO(x) :</u> Return $H(x)$</p>
---	--

Fig. 3. The TS-UF-0 game for a threshold signature scheme TS.

verification algorithm $\text{Vf}(pk, \mu, sig)$ outputs a bit that indicates whether sig is valid for (pk, μ) .

In summary, an honest execution of the signing protocol between signers in SS and the leader to sign a message $\mu \in \{0, 1\}^*$ is represented in the game TS-COR (defined in Fig. 2), and we say that TS is *correct* with correctness error δ if for any adversary \mathcal{A} for the game TS-COR, we have $\Pr[\text{TS-COR}_{\text{TS}}^A(\kappa) = 1] \leq \delta$.

SECURITY. A hierarchy for security notions of threshold signatures is proposed in [7]. In this paper, we consider TS-UF-0, which guarantees that an adversary can generate a valid signature sig for μ only if it receives partial signatures from at least one honest signer for μ . We also note that the same security notion is also used in all the prior lattice-based works, such as [45, 60]. Formally, the TS-UF-0 game is defined in Fig. 3, where TS.HF denotes the space of the hash functions used in TS from which the random oracle is drawn. The advantage of \mathcal{A} for the TS-UF-0 game is defined as $\text{Adv}_{\text{TS}}^{\text{ts-uf-0}}(\mathcal{A}, \kappa) := \Pr[\text{TS-UF-0}_{\text{TS}}^A(\kappa) = 1]$.

4.2 Construction

Our threshold signature scheme TSL[SecSha] is shown in Fig. 4, where SecSha is a linear secret sharing scheme with small coefficients (see Definition 2), which can be instantiated from Benaloh and Leichter’s secret sharing scheme as discussed

<p>Setup(1^ℓ) :</p> <p>$A \leftarrow \\$ R_q^{k \times m}$; $par \leftarrow A$</p> <p>For $i \in [n]$ do</p> <p style="padding-left: 20px;">$st_0.curPP_i \leftarrow \emptyset$</p> <p style="padding-left: 20px;">$st_i.mapPP \leftarrow ()$</p> <p>Return par</p> <p>KeyGen() :</p> <p>$sk \leftarrow \\$ \mathcal{B}_{\beta_{sk}}^m$; $pk \leftarrow Ask \text{ mod } q$</p> <p>$\{ss_j\}_{j \in [L]} \leftarrow \\$ SecSha.Share(sk)$</p> <p>For $i \in [n]$ do $sk_i \leftarrow \{ss_j\}_{j \in T_i}$</p> <p>Return $(pk, \{sk_i\}_{i \in [n]})$</p> <p>SPP($st_i$) :</p> <p>For $j \in [0..\ell]$ do $r_j \leftarrow \\$ \mathcal{D}_{\sigma_r}^m$</p> <p>For $j \in [0..\ell]$ do $R_j \leftarrow Ar_j \text{ mod } q$</p> <p>$pp \leftarrow \{R_j\}_{j \in [0..\ell]}$</p> <p>$st_i.mapPP(pp) \leftarrow \{r_j\}_{j \in [0..\ell]}$</p> <p>Return (pp, st_i)</p> <p>LPP(i, pp, st_0) :</p> <p>$st_0.curPP_i \leftarrow st_0.curPP_i \cup \{pp\}$</p> <p>Return st_0</p> <p>LR(μ, SS, st_0) :</p> <p>If $\exists i \in SS : st_0.curPP_i = \emptyset$ then</p> <p style="padding-left: 20px;">Return \perp</p> <p>$lr.msg \leftarrow \mu$; $lr.SS \leftarrow SS$</p> <p>For $i \in SS$ do</p> <p style="padding-left: 20px;">Pick pp_i from $st_0.curPP_i$</p> <p style="padding-left: 20px;">$lr.PP(i) \leftarrow pp_i$</p> <p style="padding-left: 20px;">$st_0.curPP_i \leftarrow st_0.curPP_i \setminus \{pp_i\}$</p> <p>Return (lr, st_0)</p>	<p>CompPar(pk, lr) :</p> <p>$\mu \leftarrow lr.msg$</p> <p>For $i \in lr.SS$ do</p> <p style="padding-left: 20px;">$\{b_j\}_{j \in [\ell]} \leftarrow H_1(pk, lr)$</p> <p style="padding-left: 20px;">$\{R_{i,j}\}_{j \in [0..\ell]} \leftarrow lr.PP(i)$</p> <p>$R \leftarrow \sum_{i \in lr.SS} (R_{i,0} + \sum_{j \in [\ell]} b_j R_{i,j})$</p> <p>$c \leftarrow H_2(pk, \mu, R)$</p> <p>Return $(R, c, \{b_j\}_{j \in [\ell]})$</p> <p>PS($lr, i, st_i$) :</p> <p>$pp_i \leftarrow lr.PP(i)$</p> <p>If $st_i.mapPP(pp_i) = \perp$ then return (\perp, st_i)</p> <p>$\{r_j\}_{j \in [0..\ell]} \leftarrow st_i.mapPP(pp_i)$</p> <p>$st_i.mapPP(pp_i) \leftarrow \perp$</p> <p>$(R, c, \{b_j\}_{j \in [\ell]}) \leftarrow CompPar(st_i.pk, lr)$</p> <p>$\{ss_j\}_{j \in T_i} \leftarrow st_i.sk$</p> <p>$z \leftarrow r_0 + \sum_{j \in [\ell]} b_j \cdot r_j$</p> <p style="padding-left: 40px;">$+ 2c \cdot \sum_{j \in T_i} \lambda_j^{lr.SS} ss_j \text{ mod } q$</p> <p>Return $((R, z), st_i)$</p> <p>Agg(PS, st_0) :</p> <p>$R \leftarrow \perp$; $z \leftarrow 0$</p> <p>For $(R', z') \in PS$ do</p> <p style="padding-left: 20px;">If $R = \perp$ then $R \leftarrow R'$</p> <p style="padding-left: 20px;">If $R \neq R'$ then return (\perp, st_0)</p> <p style="padding-left: 20px;">$z \leftarrow z + z'$</p> <p>Return $((R, z), st_0)$</p> <p>Vf(pk, μ, sig) :</p> <p>$(R, z) \leftarrow sig$</p> <p>If $\ z\ > \beta_z$ then return 0</p> <p>$c \leftarrow H_2(pk, \mu, R)$</p> <p>Return $(Az = R + 2c \cdot pk \text{ mod } q)$</p>
---	---

Fig. 4. Lattice-based t -out-of- n threshold signatures TSL[SecSha], where SecSha is a linear secret sharing scheme with small coefficients (see Definition 2). Here, $H_1 : \{0, 1\}^* \rightarrow \mathcal{S}_b^\ell$ and $H_2 : \{0, 1\}^* \rightarrow \mathcal{S}_c$. Also, T_i denotes the set of shares of party i and $\lambda_j^{lr.SS}$ denotes the reconstruction coefficient. Also, we remark that, as stated earlier, the system parameter par is implicitly given to all algorithms except Setup.

in Sect. 3.2. Each T_i and $\lambda_j^{lr.SS}$ are defined by the scheme SecSha. In particular, the secret key $sk \in R_q^m$ is shared into L secret shares $\{ss_j\}_{j \in [L]}$, and for each party $i \in [n]$, its secret key share is $\{ss_j\}_{j \in T_i}$. For a signer set SS where $|SS| \geq t$, by the linearity property of SecSha, the secret key can be reconstructed as $sk \leftarrow \sum_{i \in SS} \sum_{j \in T_i} \lambda_j^{SS} ss_j$. Although we do not explicitly provide a DKG protocol, we do not see our use of Benaloh and Leichter's secret sharing in place of Shamir's secret sharing to be a barrier in constructing a DKG. For instance, the DKG protocol given in [45] is a possible candidate as it only utilizes the linearity of the secret sharing schemes, which is satisfied by both Shamir's and Benaloh and Leichter's secret sharing.

For the signing protocol, in the offline round, each signer generates $\ell + 1$ nonces $\{R_j\}_{j \in [0..\ell]}$ as a pre-processing token, where $R_j \leftarrow Ar_j$ for a uniformly sampled $A \in R_q^{k \times m}$ generated during the setup phase and r_j sampled from the discrete Gaussian distribution $\mathcal{D}_{\sigma_r}^m$. In the online round, given a leader request

Parameter	Description
κ	Security parameter
n	Number of parties
t	Threshold for signing
L	Total size of the secret shares
$N \geq 2\kappa$	A power of two integer
q	Prime modulus
$R = \mathbb{Z}[X]/(X^N + 1)$	Cyclotomic Ring
$R_q = \mathbb{Z}_q[X]/(X^N + 1)$	Ring
β_{sk}	The ℓ_∞ -norm bound of the secret key sk
k	Number of rows of A
$m = (2\kappa/N + k \log q) / \log(2\beta_{sk})$	Number of columns of A
$\ell + 1 = 2\kappa / \log(2N) + 1$	Number of nonces for each signer
$S_b = \{\pm 1, \pm X, \dots, \pm X^{N-1}\}$	Set for the aggregating coefficients b_j
β_c chosen such that $2^{\beta_c} \binom{N}{\beta_c} \geq 2^{2\kappa}$	The ℓ_1 -norm of the challenge c
$S_c = \{c \in R : \ c\ _\infty = 1, \ c\ _1 = \beta_c\}$	Set of the challenges c
$\mathcal{B}_{\beta_{sk}} = \{s \in R : \ s\ _\infty \leq \beta_{sk}\}$	Set of elements with bounded ℓ_∞ -norm
β_{ss}	Lemma 12's ℓ_∞ -norm bound of SecSha
$\sigma_r = \max\{N\sqrt{32\pi q_s m N} \beta_c \beta_{ss} \beta_{sk}, \frac{16N\sqrt{3m}}{\sqrt{\pi}} q^{\frac{k}{m}} \sqrt{N(\log(2mN) + 2\kappa)}\}$	Standard deviation of the preimages $r_{i,j}$ for $j \in [0..\ell]$
$\beta_z = \sqrt{mN}(\sqrt{t(\ell + 1)}\sigma_r + 2\beta_c\beta_{sk})$	The ℓ_2 -norm bound for the aggregated z

Fig. 5. Table showing the parameters for the scheme TSL.

lr , each signer computes an aggregated nonce \mathbf{R} from a list of tokens generated by signers in $lr.SS$ using coefficients $\{b_j \in R\}_{j \in [\ell]}$ output from a hash function H_1 and computes a challenge $c \in R$ from another hash function H_2 as described in the algorithm CompPar . Each signer then returns its partial signature (\mathbf{R}, z) . It is worth noting that we include \mathbf{R} in partial signatures for the simplicity of presenting our protocol. In actual implementations, each signer only needs to send back z since the leader can compute \mathbf{R} from lr by itself.

We note that our protocol does not achieve identifiable abort, i.e., one cannot identify misbehaving signers if the final signature obtained from aggregating the partial signatures is not valid. However, we point out that one possible fix is to let KeyGen additionally output the commitments of the secret key shares as public information and, during signing, have each signer send an NIZK along with their partial signature proving that the partial signature is computed honestly with respect to the committed key shares and the first round nonces, which allows the leader to verify the correctness of the partial signature. A similar approach can be found in [45]. We also note that, except for the recent work [36], which uses 4 online rounds to achieve robustness, other prior works satisfying robustness/identifiable abort rely on advanced primitives, such as NIZKs [14, 21, 45] or homomorphic signatures [2].

PARAMETERS. In Fig. 5, we give the description of the parameters used in the protocol. We set ℓ and β_c such that the sizes of S_b^ℓ and S_c are at least $2^{2\kappa}$. We set m such that except for a negligible probability, for a secret key uniformly

sampled from $\mathcal{B}_{\beta_{sk}}^m$, there exists another secret key in $\mathcal{B}_{\beta_{sk}}^m$ such that their corresponding public keys are the same. We set β_z according to the correctness proof and σ_r according to the unforgeability proof.

CORRECTNESS AND UNFORGEABILITY. The following theorems establish the correctness and unforgeability of TSL. We show correctness in the full version of the paper, while we show TS-UF-0 under the MSIS assumption in the random oracle model below.

Theorem 1 (Correctness of TSL). *The threshold signature scheme TSL is correct with correctness error $\delta = (2 + 4t(\ell + 1)) \cdot 2^{-2\kappa}$.*

Theorem 2 (TS-UF-0 of TSL). *For any integers $q = q(\kappa), k = k(\kappa), m = m(\kappa)$ and any TS-UF-0 adversary \mathcal{A} making at most $q_s = q_s(\kappa)$ queries to PPO and $q_h = q_h(\kappa)$ queries to RO, there exists an MSIS adversary \mathcal{B} running in time roughly two times that of \mathcal{A} such that, for any $\alpha \geq 2$,*

$$\text{Adv}_{\text{TSL}}^{\text{ts-uf-0}}(\mathcal{A}, \kappa) \leq \sqrt{q \left(2\alpha \delta_\alpha \text{Adv}_{q,k,m,\beta}^{\text{msis}}(\mathcal{B}, \kappa) \right)^{1 - \frac{1}{\alpha}} + q(2 + 8q^2)2^{-2\kappa}}.$$

where $q = q_h + q_s + 1$, $\beta = 2\beta_z + 4\sqrt{mN}\beta_c\beta_{sk}$, and $\delta_\alpha = (1 + 160\ell q \cdot 2^{-2\kappa}) \cdot e^\alpha$.

To prove the above theorem, we use the following variant of the forking lemma from [8], which is proved in the full version of the paper. The only difference is that here each h_i might be sampled independently from a different distribution. We require it in our proof since the ranges of H_1 and H_2 are different.

Lemma 14. *Let $q \geq 1$ be an integer, $S \subseteq [1..q]$ be a set, and HG be an algorithm that outputs h_1, \dots, h_q where each h_i is independently sampled. Let \mathcal{A} be a randomized algorithm that on input x, h_1, \dots, h_q outputs a pair (I, Out) , where $I \in \{\perp\} \cup S$ and Out is a side output. Let IG be a randomized algorithm that generates x . The accepting probability of \mathcal{A} is defined as*

$$\text{acc}(\mathcal{A}) = \Pr_{x \leftarrow \mathcal{IG}, h_1, \dots, h_q \leftarrow \mathcal{HG}}[(I, \text{Out}) \leftarrow \mathcal{A}(x, h_1, \dots, h_q) : I \neq \perp].$$

Consider algorithm $\text{Fork}^{\mathcal{A}}$ described in Fig. 6. The accepting probability of $\text{Fork}^{\mathcal{A}}$ is defined as $\text{acc}(\text{Fork}^{\mathcal{A}}) = \Pr_{x \leftarrow \mathcal{IG}}[\alpha \leftarrow \mathcal{A}(x) : \alpha \neq \perp]$. Then, $\text{acc}(\text{Fork}^{\mathcal{A}}) \geq \text{acc}(\mathcal{A})^2/|S|$.

Proof (of Theorem 2). Let \mathcal{A} be a TS-UF-0 adversary described in the theorem. W.l.o.g. we assume that \mathcal{A} is deterministic and corrupts exactly $t - 1$ signers. Also, we assume if \mathcal{A} returns $(\mu^*, (\mathbf{R}^*, \mathbf{z}^*))$, the RO query $H_2(\text{pk}, \mu^*, \mathbf{R}^*)$ was made by \mathcal{A} , which adds at most one RO query. Also, since the game makes at most one RO query to H_1 and H_2 respectively for each signing query, the total number of RO queries to each of H_1 and H_2 is bounded $q = q_h + q_s + 1$. We first construct an algorithm \mathcal{C} compatible with the syntax in Lemma 14 and construct \mathcal{B} from $\text{Fork}^{\mathcal{C}}$. The input of \mathcal{C} consists of $\text{par} = A$, public key pk , secret key shares

```

ForkA(x) :
Pick the random coin ρ of A at random
(h1, . . . , hq), (h̄1, . . . , h̄q) ←s HG
(I, Out) ← A(x, h1, . . . , hq; ρ)
If I = ⊥ then return ⊥
(Ī, Out̄) ← A(x, h1, . . . , hI-1, h̄I, . . . , h̄q; ρ)
If I ≠ Ī then return ⊥
Return (I, Out, Out̄)
    
```

Fig. 6. The forking algorithm build from \mathcal{A} .

$\{\text{sk}_i\}_{i \in [n]}$, the randomness $\{\mathbf{r}_j^{(i)}\}_{i \in [q_s], j \in [0..\ell]}$ for generating the nonces, and the random RO outputs h_1, \dots, h_{2q} , where $h_{2i-1} \in \mathcal{S}_b^\ell$ and $h_{2i} \in \mathcal{S}_c$ for $i \in [q]$. To start with, \mathcal{C} does initialization exactly as in the game TS-UF-0 and then runs \mathcal{A} with access to oracles INIT, PPO, PSIGNO simulated in the same manner as in the game TS-UF-0 (the randomness $\{\mathbf{r}_j^{(i)}\}_{j \in [0..\ell]}$ are used for the i -th signing query to PPO) and the RO oracle $\widetilde{\text{RO}}$, which is simulated as follows.

$\widetilde{\text{RO}}$ query $H_1(x)$: If $H_1(x) \neq \perp$, \mathcal{C} returns $H_1(x)$. Otherwise, parse x as $(\widetilde{\text{pk}}, lr)$.

If the parsing fails or $\widetilde{\text{pk}} \neq \text{pk}$, \mathcal{C} sets $H_1(x) \leftarrow \mathcal{S}_b^\ell$ and returns $H_1(x)$. Otherwise, \mathcal{C} increases ctr_h by 1, sets $H_1(x) \leftarrow h_{2\text{ctr}_h-1}$. Also, \mathcal{C} computes $\mathbf{R} \leftarrow \sum_{i \in lr.\text{SS}} (\mathbf{R}_{i,0} + \sum_{j \in [\ell]} b_j \cdot \mathbf{R}_{i,j})$, where $(\mathbf{R}_{i,j})_{j \in [0..\ell]} \leftarrow lr.\text{PP}(i)$ and $\{b_j\}_{j \in [\ell]} \leftarrow h_{2\text{ctr}_h-1}$. If $H_2(\text{pk}, lr.\text{msg}, \mathbf{R}) = \perp$, \mathcal{C} sets $H_2(\text{pk}, lr.\text{msg}, \mathbf{R}) \leftarrow h_{2\text{ctr}_h}$. Finally, \mathcal{C} returns $H_1(x)$.

$\widetilde{\text{RO}}$ query $H_2(x)$: If $H_2(x) \neq \perp$, \mathcal{C} returns $H_2(x)$. Otherwise, parse x as $(\widetilde{\text{pk}}, \mu, \mathbf{R})$. If the parsing fails or $\widetilde{\text{pk}} \neq \text{pk}$, \mathcal{C} sets $H_2(x) \leftarrow \mathcal{S}_c$. Otherwise, \mathcal{C} increases ctr_h by 1 and sets $H_2(x) \leftarrow h_{2\text{ctr}_h}$. Finally, \mathcal{C} returns $H_2(x)$.

After receiving the output $(\mu^*, (\mathbf{R}^*, \mathbf{z}^*))$ from \mathcal{A} , \mathcal{C} aborts if \mathcal{A} does not win the TS-UF-0 game. Otherwise \mathcal{C} finds the index I such that $H_2(\text{pk}, \mu^*, \mathbf{R}^*)$ is set to h_I during the simulation. By our assumption of \mathcal{A} , we know such I must exist. Then, \mathcal{C} returns $(I, \text{Out} = (\mu^*, \mathbf{R}^*, \mathbf{z}^*))$.

ANALYSIS OF \mathcal{C} . To use Lemma 14, we define $S := \{2j\}_{j \in [q]}$ and IG as the algorithm that runs $A \leftarrow \text{Setup}(1^\kappa)$, $(\text{pk}, \{\text{sk}_i\}_{i \in [n]}) \leftarrow \text{KeyGen}()$, samples $\mathbf{r}_j^{(i)} \leftarrow \mathcal{D}_{\sigma_r}^m$ for each $i \in [q_s]$ and $j \in [0..\ell]$, and returns $(A, \text{pk}, \{\text{sk}_i\}_{i \in [n]}, \{\mathbf{r}_j^{(i)}\}_{i \in [q_s], j \in [0..\ell]})$. We define HG as the algorithm that samples $h_1, h_3, \dots, h_{2q-1}$ uniformly from \mathcal{S}_b^ℓ and h_2, h_4, \dots, h_{2q} uniformly from \mathcal{S}_c . From the simulation, we know that the output index I of \mathcal{C} is always in S . Also, we can see that \mathcal{C} simulates the game TS-SUF-0 perfectly, which implies $\text{acc}(\mathcal{C}) \geq \text{Adv}_{\text{TSL}}^{\text{ts-uf-0}}(\mathcal{A}, \kappa)$. By Lemma 14, we have that $\text{acc}(\text{Fork}^{\mathcal{C}}) \geq \text{Adv}_{\text{TSL}}^{\text{ts-uf-0}}(\mathcal{A}, \kappa)^2/q$.

CONSTRUCT \mathcal{B} FROM $\text{Fork}^{\mathcal{C}}$. We now construct the MSIS adversary \mathcal{B} using $\text{Fork}^{\mathcal{C}}$. To start with, \mathcal{B} receives $A \in R_q^{k \times m}$ from the MSIS game, follows the algorithm $\text{KeyGen}()$ to generate $(\text{pk}, \text{sk}, \{\text{sk}_i\}_{i \in [n]})$, and samples $\{\mathbf{r}_j^{(i)}\}_{i \in [q_s], j \in [0..\ell]}$ exactly as in IG. Then, \mathcal{B} runs $\text{Fork}^{\mathcal{C}}$. If $\text{Fork}^{\mathcal{C}}$ outputs $(I, \text{Out} = (\mu^*, \mathbf{R}^*, \mathbf{z}^*), \overline{\text{Out}} = (\bar{\mu}^*, \bar{\mathbf{R}}^*, \bar{\mathbf{z}}^*))$, \mathcal{B} returns $\mathbf{z}^* - \bar{\mathbf{z}}^* - 2(h_I - \bar{h}_I)\text{sk}$. Otherwise, \mathcal{B} aborts.

By the execution of Fork^C , we know $(\mu^*, \mathbf{R}^*) = (\bar{\mu}^*, \bar{\mathbf{R}}^*)$, $Az^* = \mathbf{R}^* + 2h_I \cdot \text{pk}$ and $A\bar{z}^* = \bar{\mathbf{R}}^* + 2\bar{h}_I \cdot \text{pk}$. Therefore, $A(z^* - \bar{z}^* - 2(h_I - \bar{h}_I)\text{sk}) = 0$. Also, it is clear that $\|z^* - \bar{z}^* - 2(h_I - \bar{h}_I)\text{sk}\| \leq 2\beta_z + 2\sqrt{mN} \|(h_I - \bar{h}_I)\text{sk}\|_\infty \leq 2\beta_z + 4\sqrt{mN}\beta_c\beta_{\text{sk}} = \beta$, where the last inequality is due to $\|(h_I - \bar{h}_I)\text{sk}\|_\infty \leq 2\beta_c\beta_{\text{sk}}$.

It is left to show that $z^* - \bar{z}^* - 2(h_I - \bar{h}_I)\text{sk} \neq 0$ with high probability. Denote Win as the event that \mathcal{B} returns and $z^* - \bar{z}^* - 2(h_I - \bar{h}_I)\text{sk} \neq 0$, which means that \mathcal{B} wins the MSIS game, and Zero as the event that \mathcal{B} returns and $z^* - \bar{z}^* - 2(h_I - \bar{h}_I)\text{sk} = 0$. Since \mathcal{B} returns if Fork^C returns,

$$\Pr[\text{Win} \vee \text{Zero}] = \text{acc}(\text{Fork}^C) \geq \text{Adv}_{\text{TSL}}^{\text{ts-uf-0}}(\mathcal{A}, \kappa)^2/q. \quad (1)$$

Denote BadHash as the event that $h_1, \bar{h}_1, \dots, h_{2q}, \bar{h}_{2q}$ are *not* all distinct, \mathcal{S}_{gA} as the set of MSIS challenge $A \in R_q^{k \times m}$ that $\eta_\varepsilon(A_q^1(A)) \leq \sigma_r/(2N\sqrt{3m})$, and $\mathcal{S}_{\text{gk},A}$ as the set of secret key $\text{sk} \in \mathcal{B}_{\beta_{\text{sk}}}^m$ such that there exists another key $\text{sk}' \neq \text{sk}$ and $\text{Ask}' = \text{Ask}$. Then, denote the event Good as $(\neg \text{BadHash} \wedge A \in \mathcal{S}_{\text{gA}} \wedge \text{sk} \in \mathcal{S}_{\text{gk},A})$. We bound $\Pr[\text{Win}]$ using the following main lemma proved in Sect. 4.3.

Lemma 15. *For any $\alpha \geq 2$, $\Pr[\text{Win} \wedge \text{Good}] \geq \Pr[\text{Zero} \wedge \text{Good}]^{\alpha/(\alpha-1)}/\delta_\alpha$, where $\delta_\alpha = (1 + 160\ell q \cdot 2^{-2\kappa}) \cdot e^\alpha$.*

We now show that Good occurs with overwhelming probability. By Lemma 4, $\Pr[A \notin \mathcal{S}_{\text{gA}}] \leq 2^{-N} \leq 2^{-2\kappa}$. Since each of $h_1, h_3, \dots, h_{2q-1}$ and h_2, h_4, \dots, h_{2q} are sampled uniformly from \mathcal{S}_b^ℓ and \mathcal{S}_c respectively, $\Pr[\text{BadHash}] \leq (2q)^2/|\mathcal{S}_b|^\ell + (2q)^2/|\mathcal{S}_c| \leq 8q^2 2^{-2\kappa}$. Also, we can see that $\Pr[\text{sk} \notin \mathcal{S}_{\text{gk},A}] \leq 2^{-2\kappa}$, since the size of $\mathcal{B}_{\beta_{\text{sk}}}^m$ is much larger than R_q^k . In particular, there is at most q^{kN} possible values of Ask , so $\text{sk} \notin \mathcal{S}_{\text{gk},A}$ with probability at most $q^{kN}/(2\beta_{\text{sk}})^{mN}$. The bound then follows from the value of m in Fig. 5. Thus, $\Pr[\neg \text{Good}] \leq (2 + 8q^2)2^{-2\kappa}$.

Finally, by Lemma 15 and Equation (1), we conclude our theorem, since

$$\begin{aligned} \Pr[\text{Win}] &\geq \frac{1}{2} \left(\Pr[\text{Win} \wedge \text{Good}] + \Pr[\text{Zero} \wedge \text{Good}]^{\alpha/(\alpha-1)}/\delta_\alpha \right) \\ &\geq \frac{\alpha-1}{2\alpha\delta_\alpha} (\Pr[\text{Win} \wedge \text{Good}] + \Pr[\text{Zero} \wedge \text{Good}]^{\alpha/(\alpha-1)}) \\ &\geq \frac{1}{2\alpha\delta_\alpha} (\Pr[(\text{Win} \vee \text{Zero}) \wedge \text{Good}]^{\alpha/(\alpha-1)}) \\ &\geq \frac{1}{2\alpha\delta_\alpha} \left(\text{Adv}_{\text{TSL}}^{\text{ts-uf-0}}(\mathcal{A}, \kappa)^2/q - (2 + 8q^2)2^{-2\kappa} \right)^{\alpha/(\alpha-1)}, \end{aligned}$$

where the third inequality is due to Lemma 11 and the fact that $\delta_\alpha > 1$. \square

4.3 Proof of Lemma 15

By the definition of $\mathcal{S}_{\text{gk},A}$, there exists a bijection $f_A : \mathcal{S}_{\text{gk},A} \rightarrow \mathcal{S}_{\text{gk},A}$ such that $f_A(\text{sk}) \neq \text{sk}$ and $A \cdot f(\text{sk}) = A \cdot \text{sk}$. Denote a random variable $T_{A,\text{sk},h}$ as the view of \mathcal{A} during its interaction with \mathcal{B} given the MSIS challenge being A , the secret key being sk and the hash values being $\mathbf{h} = (h_1, \dots, h_{2q_h}, \bar{h}_1, \dots, \bar{h}_{2q_h})$ for answering

RO queries. More concretely, $T_{A,\text{sk},\mathbf{h}}$ contains the public key pk , the secret key shares of corrupted signers $\{\text{sk}_j\}_{j \in CS}$, the transcripts of all queries to the oracles PPO, PSIGNO, RO, and the outputs of \mathcal{A} in both executions. Denote $W_{A,\text{sk},\mathbf{h}}$ as the distribution of $T_{A,\text{sk},\mathbf{h}}$. Denote \mathcal{S}_{gh} as the set of hash values \mathbf{h} such that BadHash does not occur.

We first show that the lemma holds if $R_\alpha(W_{A,\text{sk},\mathbf{h}} \| W_{A,f_A(\text{sk}),\mathbf{h}}) \leq \delta_\alpha$ for any $A \in \mathcal{S}_{\text{gA}}$, $\text{sk} \in \mathcal{S}_{\text{gk},A}$ and $\mathbf{h} \in \mathcal{S}_{\text{gh}}$. Given a view T , we denote $(\mu^*, \mathbf{R}^*, \mathbf{z}^*)$ and $(\bar{\mu}^*, \bar{\mathbf{R}}^*, \bar{\mathbf{z}}^*)$ as the outputs of \mathcal{A} in T , and we follow the execution of \mathcal{C} to find an index I such that $\text{H}_2(\text{pk}, \mu^*, \mathbf{R}^*)$ is set to h_I if \mathcal{A} wins during the first execution. Denote \bar{I} as such an index for the second execution of \mathcal{A} . We define the event E_x as \mathcal{A} wins in both executions and $I = \bar{I} \wedge \mathbf{z}^* - \bar{\mathbf{z}}^* - 2(h_I - \bar{h}_I)x = 0$.

For any fixed $A \in \mathcal{S}_{\text{gA}}$, $\text{sk} \in \mathcal{S}_{\text{gk},A}$, $\mathbf{h} \in \mathcal{S}_{\text{gh}}$ and $T \leftarrow_{\$} W_{A,\text{sk},\mathbf{h}}$, if E_{sk} occurs, since $\text{sk} \neq f_A(\text{sk})$ and $\mathbf{h} \in \mathcal{S}_{\text{gh}}$ which implies $h_I - \bar{h}_I \neq 0$, we know $\mathbf{z}^* - \bar{\mathbf{z}}^* - 2(h_I - \bar{h}_I)f_A(\text{sk}) \neq \mathbf{z}^* - \bar{\mathbf{z}}^* - 2(h_I - \bar{h}_I)\text{sk} = 0$, which means that \mathcal{B} wins the MSIS game given $(A, f_A(\text{sk}), \mathbf{h}, T)$. Therefore, $\Pr[\text{Win}|A, f_A(\text{sk}), \mathbf{h}] \geq \Pr_{T \leftarrow_{\$} W_{A,\text{sk},\mathbf{h}}}[E_{\text{sk}}]$, where $\Pr[\text{Win}|A, f_A(\text{sk}), \mathbf{h}]$ denotes the probability that Win occurs given the MSIS challenge being A , the secret key being $f_A(\text{sk})$, and the hash values being \mathbf{h} . For $T \leftarrow_{\$} W_{A,\text{sk},\mathbf{h}}$, if E_{sk} occurs, we know the event Zero occurs given the secret key being sk and the view of \mathcal{A} being T , which means $\Pr[\text{Zero}|A, \text{sk}, \mathbf{h}] = \Pr_{T \leftarrow_{\$} W_{A,\text{sk},\mathbf{h}}}[E_{\text{sk}}]$. Therefore, by Lemma 5,

$$\begin{aligned} \Pr[\text{Win}|A, f_A(\text{sk}), \mathbf{h}] &\geq \Pr_{T \leftarrow_{\$} W_{A,f_A(\text{sk}),\mathbf{h}}}[E_{\text{sk}}] \\ &\geq \Pr_{T \leftarrow_{\$} W_{A,\text{sk},\mathbf{h}}}[E_{\text{sk}}]^{\alpha/(\alpha-1)} / R_\alpha(W_{A,\text{sk},\mathbf{h}} \| W_{A,f_A(\text{sk}),\mathbf{h}}) \\ &\geq \Pr[\text{Zero}|A, \text{sk}, \mathbf{h}]^{\alpha/(\alpha-1)} / \delta_\alpha, \end{aligned}$$

which implies

$$\begin{aligned} \Pr[\text{Win}|\text{Good}] &= \mathbb{E}_{A,\text{sk},\mathbf{h}}[\Pr[\text{Win}|A, f_A(\text{sk}), \mathbf{h}]] \geq \mathbb{E}_{A,\text{sk},\mathbf{h}}\left[\frac{1}{\delta_\alpha} \Pr[\text{Zero}|A, \text{sk}, \mathbf{h}]^{\frac{\alpha}{\alpha-1}}\right] \\ &\geq \frac{1}{\delta_\alpha} \mathbb{E}_{A,\text{sk},\mathbf{h}}[\Pr[\text{Zero}|A, \text{sk}, \mathbf{h}]^{\frac{\alpha}{\alpha-1}}] = \frac{1}{\delta_\alpha} \Pr[\text{Zero}|\text{Good}]^{\frac{\alpha}{\alpha-1}}, \end{aligned}$$

where the expectation is taken over $(A, \text{sk}, \mathbf{h})$ uniformly sampled from $\mathcal{S}_{\text{gA}} \times \mathcal{S}_{\text{gk},A} \times \mathcal{S}_{\text{gh}}$, the first equation is due to the fact that f_A is a bijection on $\mathcal{S}_{\text{gk},A}$, and the last inequality is due to Jensen's inequality. Therefore,

$$\begin{aligned} \Pr[\text{Win} \wedge \text{Good}] &= \Pr[\text{Win}|\text{Good}]\Pr[\text{Good}] \geq \frac{1}{\delta_\alpha} \Pr[\text{Good}] \cdot \Pr[\text{Zero}|\text{Good}]^{\alpha/(\alpha-1)} \\ &\geq \frac{1}{\delta_\alpha} (\Pr[\text{Good}] \cdot \Pr[\text{Zero}|\text{Good}])^{\alpha/(\alpha-1)} = \frac{1}{\delta_\alpha} (\Pr[\text{Zero} \wedge \text{Good}])^{\alpha/(\alpha-1)}, \end{aligned}$$

where the second inequality is due to $\Pr[\text{Good}] \leq 1$ and $\frac{\alpha}{\alpha-1} > 1$.

ANALYSIS OF $R_\alpha(W_{A,\text{sk},\mathbf{h}} \| W_{A,f_A(\text{sk}),\mathbf{h}})$. We first define a more fine-grained view $T_{A,\text{sk},\rho,\mathbf{h}}$ by further fixing the randomness ρ used for generating the secret key

shares. We can view $W_{A,\text{sk},h}$ as the distribution of $T_{A,\text{sk},\rho,h}$ for ρ uniformly sampled from $(R_q^m)^K$.

We also extend the bijection f_A to a bijection f'_A that additionally takes the randomness ρ as input such that f'_A maps (sk, ρ) to $(f_A(\text{sk}), \rho')$ such that the shares of corrupted signers CS given (sk, ρ) are the same as that given $(f_A(\text{sk}), \rho')$.⁵ By Lemma 12, we construct the bijection as $f'_A(\text{sk}, \rho) := (f_A(\text{sk}), \Phi_{\text{sk}, f_A(\text{sk}), CS}(\rho))$. As a result, $W_{A, f_A(\text{sk}), h}$ can be viewed as the distribution of $T_{A, f'_A(\text{sk}, \rho), h}$ for uniformly sampled ρ .

Denote $W_{A,\text{sk},\rho,h}$ as the distribution of $T_{A,\text{sk},\rho,h}$. Denote P as the distribution of $(\rho, T_{A,\text{sk},\rho,h})$ and Q as the distribution of $(\rho, T_{A, f'_A(\text{sk}, \rho), h})$ for uniformly sampled ρ . By the data processing inequality from Lemma 5, we have that $R_\alpha(W_{A,\text{sk},h} \| W_{A, f_A(\text{sk}), h}) \leq R_\alpha(P \| Q)$. By Lemma 6, denoting P_1 as the uniform distribution of ρ and $P_{2|\rho}$ as the distribution of $T_{A,\text{sk},\rho,h}$ conditioned on the value of ρ (Q_1 and $Q_{2|\rho}$ are defined analogously), then $R_\alpha(P \| Q) \leq R_\alpha(P_1 \| Q_1) \cdot \max_\rho R_\alpha(P_{2|\rho} \| Q_{2|\rho}) = \max_\rho R_\alpha(W_{A,\text{sk},\rho,h} \| W_{A, f'_A(\text{sk}, \rho), h})$.

Therefore, $R_\alpha(W_{A,\text{sk},h} \| W_{A, f_A(\text{sk}), h}) \leq \max_\rho R_\alpha(W_{A,\text{sk},\rho,h} \| W_{A, f'_A(\text{sk}, \rho), h})$, and we can conclude the lemma by the following claim.

Claim. For any $A \in \mathcal{S}_{\text{gA}}$, $\text{sk} \in \mathcal{S}_{\text{gk}, A}$, $\rho \in (R_q^m)^K$, and $h \in \mathcal{S}_{\text{gh}}$, we have $R_\alpha(W_{A,\text{sk},\rho,h} \| W_{A, f'_A(\text{sk}, \rho), h}) \leq (1 + 160\ell q \cdot 2^{-2\kappa}) \cdot e^\alpha$.

Proof. Denote $(\text{sk}', \rho') = f'_A(\text{sk}, \rho)$ and denote $\{\text{ss}_i\}_{i \in [L]}$ and $\{\text{ss}'_i\}_{i \in [L]}$ as the secret shares generated by $\text{SecSha.Share}(\text{sk}; \rho)$ and $\text{SecSha.Share}(\text{sk}'; \rho')$, respectively. Since \mathcal{A} is deterministic, $T_{A,\text{sk},\rho,h}$ is determined by the nonces $\{\mathbf{R}_0^{(j)}, \dots, \mathbf{R}_\ell^{(j)}\}_{j \in [q_s]}$ and the outputs (\mathbf{R}, \mathbf{z}) of queries to oracle PSIGNO. Therefore, we only need to consider the marginal distribution of those variables when comparing the two distributions. We further ignore \mathbf{R} from the outputs of PSIGNO queries since it is determined given $\{\mathbf{R}_0^{(j)}, \dots, \mathbf{R}_\ell^{(j)}\}_{j \in [q_s]}$ and h .

We now use Lemma 6 to bound $R_\alpha(W_{A,\text{sk},\rho,h} \| W_{A, f'_A(\text{sk}, \rho), h})$ by defining random variables X_0, \dots, X_{2q_s} as follows. Let $X_0 := \{\mathbf{R}_0^{(j)}, \dots, \mathbf{R}_\ell^{(j)}\}_{j \in [q_s]}$. For $j \in [q_s]$, let X_j be the output \mathbf{z} of the j -th query to PSIGNO made by \mathcal{A} during the first execution, and let X_j be \perp if \mathcal{A} makes less than j queries to PSIGNO during the first execution. Similarly, let X_{q_s+j} be the output \mathbf{z} of the j -th query to PSIGNO made by \mathcal{A} during the second execution, and let X_{q_s+j} be \perp if \mathcal{A} does not win during the first execution or makes less than j queries to PSIGNO during the second execution. We denote D and D' as the distributions of X_0, \dots, X_{2q_s} sampled from $W_{A,\text{sk},\rho,h}$ and $W_{A, f'_A(\text{sk}, \rho), h}$, respectively.

By Lemma 6, denoting $D_j|_{x_{[0..j-1]}}$ as the distribution of X_j conditioned on $x_{[0..j-1]}$ ($D'_j|_{x_{[0..j-1]}}$ is defined analogously), we only need to upper-bound $R_\alpha(D_j|_{x_{[0..j-1]}} \| D'_j|_{x_{[0..j-1]}})$ for any $j \in [0..2q_s]$ and $x_{[0..j-1]}$. For simplicity of our explanation, we denote $\delta_{\alpha,j} := \max_{x_{[0..j-1]}} R_\alpha(D_j|_{x_{[0..j-1]}} \| D'_j|_{x_{[0..j-1]}})$.

⁵ The corrupted set CS is fixed here since we assume that \mathcal{A} is deterministic.

For $j = 0$, since $\{\mathbf{R}_0^{(j)}, \dots, \mathbf{R}_\ell^{(j)}\}_{j \in [q_s]}$ are sampled independently of sk, ρ, D_0 and D'_0 are the same distributions, which implies $\delta_{\alpha, j} = 1$.

For $1 \leq j \leq q_s$, given $X_{[0..j-1]} = x_{[0..j-1]}$ for any $x_{[0..j-1]}$, we know $T_{A, \text{sk}, \rho, \mathbf{h}}$ and $T_{A, f'_A(\text{sk}, \rho), \mathbf{h}}$ are identical prior to the j -th PSIGNO query in the first execution. Denote the j -th query to PSIGNO as (i, lr) . We say that the query corresponds to the j' -th token if $lr.\text{PP}(i) = (\mathbf{R}_0^{(j')}, \dots, \mathbf{R}_\ell^{(j')})$. Suppose that the query is valid, i.e., the query corresponds to the j' -th token for some $j' \in [q_s]$ and there is no prior PSIGNO query corresponding to the same token. Let (c, b_1, \dots, b_ℓ) be the parameters computed from $\text{CompPar}(\text{pk}, lr)$. Let $\mathbf{s}_i \in R^m$ be an arbitrary vector such that $A\mathbf{s}_i = \mathbf{R}_i^{(j')}$ for $i \in [0..\ell]$. Then, the distribution of $\mathbf{r}_i^{(j')}$ given $\mathbf{R}_i^{(j')}$ is $\mathcal{D}_{\Lambda_q^\perp(A) + \mathbf{s}_i, \sigma_r}^m$ for $i \in [0..\ell]$. Let $\mathbf{v} := \sum_{j \in T_i} \lambda_j^{lr.SS} \mathbf{ss}_j$. Since $X_j = \mathbf{r}_0^{(j')} + \sum_{i \in [\ell]} b_i \mathbf{r}_i^{(j')} + 2c\mathbf{v}$ and $\eta_\varepsilon(\Lambda_q^\perp(A)) \leq \sigma_r / (2\sqrt{3mN})$, by Lemma 9, we have $D_j \approx \mathcal{D}_{\Lambda_q^\perp(A) + \mathbf{S} + 2c\mathbf{v}, \sigma', 2c\mathbf{v}}^{m, \text{mod } q}$,⁶ where $\varepsilon' = \frac{2((1+\varepsilon)^\ell - 1)}{2 - (1+\varepsilon)^\ell}$, $\mathbf{S} = \mathbf{s}_0 + \sum_{i \in [\ell]} b_i \mathbf{s}_i$ and $\sigma' = \sigma_r \sqrt{1 + \ell}$. Similarly, $D'_j \approx \mathcal{D}_{\Lambda_q^\perp(A) + \mathbf{S} + 2c\mathbf{v}', \sigma', 2c\mathbf{v}'}$, where $\mathbf{v}' = \sum_{j \in T_i} \lambda_j^{lr.SS} \mathbf{ss}'_j$. By weak triangle inequality from Lemma 5,

$$\delta_{\alpha, j} \leq (1 + \varepsilon')^{1 + \frac{\alpha}{\alpha-1}} R_\alpha \left(\mathcal{D}_{\Lambda_q^\perp(A) + \mathbf{S} + 2c\mathbf{v}, \sigma', 2c\mathbf{v}}^{m, \text{mod } q} \parallel \mathcal{D}_{\Lambda_q^\perp(A) + \mathbf{S} + 2c\mathbf{v}', \sigma', 2c\mathbf{v}'}^{m, \text{mod } q} \right).$$

By Lemma 12, we have $A\mathbf{v} = A\mathbf{v}'$, which implies $2c(\mathbf{v} - \mathbf{v}') \in \Lambda_q^\perp(A)$, and thus the two lattice cosets $\Lambda_q^\perp(A) + \mathbf{S} + 2c\mathbf{v}$ and $\Lambda_q^\perp(A) + \mathbf{S} + 2c\mathbf{v}'$ are the same. Then, by Lemma 12, we have $\|\mathbf{v} - \mathbf{v}'\| \leq \beta_{\text{ss}} \beta_{\text{sk}} \sqrt{Nm}$, so $\|2c(\mathbf{v} - \mathbf{v}')\|^2 \leq 4\beta_c^2 \beta_{\text{ss}}^2 \beta_{\text{sk}}^2 Nm$. Thus, by Lemma 8,

$$\delta_{\alpha, j} \leq (1 + \varepsilon')^{1 + \frac{\alpha}{\alpha-1}} \exp \left(\alpha \pi \frac{\|2c(\mathbf{v} - \mathbf{v}')\|^2}{\sigma'^2} \right) \leq (1 + \varepsilon')^3 e^{\alpha/(2q)}, \quad (2)$$

where the last inequality is due to the fact that σ_r is set as shown in Lemma 5. If the j -query is not valid or \mathcal{A} makes less than j queries to PSIGNO in the first execution, we have $X_j = \perp$ in both distributions, which means $\delta_{\alpha, j} = 1$.

For $q_s + 1 \leq j \leq 2q_s$, given $X_{[0..j-1]} = x_{[0..j-1]}$ for any $x_{[0..j-1]}$, we know $T_{A, \text{sk}, \rho, \mathbf{h}}$ and $T_{A, f'_A(\text{sk}, \rho), \mathbf{h}}$ are identical prior to the $(j - q_s)$ -th PSIGNO query in the second execution. W.l.o.g. assume \mathcal{A} wins the TS-UF-0 game during the first execution since otherwise $X_j = \perp$ in both D_j and D'_j and $\delta_{\alpha, j} = 1$. Also, w.l.o.g. assume \mathcal{A} makes at least $(j - q_s)$ queries to PSIGNO and the $(j - q_s)$ -th query is valid during the second execution since otherwise $X_j = \perp$. We denote the $(j - q_s)$ -th query as (i, \bar{lr}) and let $(\bar{c}, \bar{b}_1, \dots, \bar{b}_\ell)$ be the parameters computed from $\text{CompPar}(\text{pk}, \bar{lr})$. Suppose the query corresponds to the j' -th token. There are three cases:

- The adversary does not make a PSIGNO query that corresponds to the j' -th token during the first execution. Since X_j is the distribution of \bar{z} conditioning

⁶ This follows from Lemma 9 showing that $X_j - 2c\mathbf{v}$ is distributed closely to $\mathcal{D}_{\Lambda_q^\perp(A) + \mathbf{S}, \sigma'}^{m, \text{mod } q}$.

- on $\{\mathbf{R}_0^{(j')}, \dots, \mathbf{R}_\ell^{(j')}\}$, we can use the same analysis as in the case of the first execution and get the same bound on $\delta_{\alpha,j}$ as Equation (2).
- Otherwise, the adversary makes a valid PSIGNO query that also corresponds to the j' -th token during the first execution. Denote the query as (i, lr) , and suppose it is the \tilde{j} -th PSIGNO query. (Since the query corresponds to the j' -th token, it must be for signer i too.) Let (c, b_1, \dots, b_ℓ) be the parameters computed from $\text{CompPar}(\text{pk}, lr)$ during the first execution. Denote J as the index such that $(b_1, \dots, b_\ell) = h_J$. If $lr = \bar{lr}$ and $J < I$, where we recall that I denotes the index such that $H_2(\text{pk}, \mu^*, \mathbf{R}^*) = h_I$ and $(\mu^*, (\mathbf{R}^*, \mathbf{z}^*))$ denotes the output of A during the first execution, we have $(\bar{b}_1, \dots, \bar{b}_\ell) = h_J$. Denote J' as the index such that $c = h_{J'}$. By the simulation of the random oracles, J' is either $J + 1$ or less than J . Since \mathcal{A} wins the TS-UF-0 game during the first execution, $\mu^* \neq lr.\text{msg}$, which implies $J' \neq I$. Since $J' \leq J + 1 \leq I$, we know $J' < I$. Therefore, from the algorithm CompPar , we know $c = h_{J'} = \bar{c}$, which implies that the answer to the $(j - q)$ -th PSIGNO query during the second execution is the same as the \tilde{j} -th PSIGNO query during the first execution. Thus, $X_j = X_{\tilde{j}} = x_{\tilde{j}}$ for both $D_{j|x_{[0..j-1]}}$ and $D'_{j|x_{[0..j-1]}}$ and $\delta_{\alpha,j} = 1$.
 - Otherwise, either $lr \neq \bar{lr}$ or $J > I$. Since $\mathbf{h} \in \mathcal{S}_{\text{gh}}$, in either of the cases, $(b_1, \dots, b_\ell) \neq (\bar{b}_1, \dots, \bar{b}_\ell)$. We denote the output of the \tilde{j} -th PSIGNO query during the first execution as \mathbf{z} and define $\{\mathbf{s}_i\}_{i \in [0..\ell]}$, and $(\mathbf{v}, \mathbf{v}')$ for the query following the analysis of the first execution. Then, we have that $X_j = \mathbf{r}_0 + \sum_{i \in [\ell]} \bar{b}_i \mathbf{r}_i + 2\bar{c}\bar{\mathbf{v}}$, where $\bar{\mathbf{v}} = \sum_{\hat{j} \in T_i} \lambda_{\hat{j}}^{\bar{lr}.SS} \mathbf{ss}_{\hat{j}}$ and each \mathbf{r}_i , for $i \in [0..\ell]$, is independently sampled from $\mathcal{D}_{\Lambda_q^+(A) + \mathbf{s}_i, \sigma_r}^{m, \text{mod } q}$ conditioning on $\mathbf{r}_0 + \sum_{i \in [\ell]} b_i \mathbf{r}_i = \mathbf{z} - 2c\mathbf{v}$. Denote $\mathbf{y}_0 = \mathbf{z} - 2c\mathbf{v}$. By Lemma 9,

$$D_j \stackrel{\varepsilon'}{\approx} \mathcal{D}_{\mathcal{I} \otimes \Lambda_q^+(A) + \mathbf{y}_0 + \mathbf{S}' + 2\bar{c}\bar{\mathbf{v}}, \frac{\Delta(\Sigma)}{\Sigma_{11}} \cdot \mathbb{I}_m, \frac{\Sigma_{12}}{\Sigma_{11}} \mathbf{y}_0 + 2\bar{c}\bar{\mathbf{v}}}$$

where $\mathbf{S}' = \sum_{i \in [\ell]} (\bar{b}_i - b_i) \mathbf{s}_i$, \mathcal{I} denotes the ideal generated by $b_1 - \bar{b}_1, \dots, b_n - \bar{b}_n$, and $\Sigma = \sigma_r^2 \begin{pmatrix} 1 + \sum_{i \in [\ell]} b_i^* b_i & 1 + \sum_{i \in [\ell]} \bar{b}_i b_i^* \\ 1 + \sum_{i \in [\ell]} \bar{b}_i^* b_i & 1 + \sum_{i \in [\ell]} b_i^* b_i \end{pmatrix}$.

Similarly, $D'_j \stackrel{\varepsilon'}{\approx} \mathcal{D}_{\mathcal{I} \otimes \Lambda_q^+(A) + \mathbf{y}'_0 + \mathbf{S}' + 2\bar{c}\bar{\mathbf{v}}', \frac{\Delta(\Sigma)}{\Sigma_{11}} \cdot \mathbb{I}_m, \frac{\Sigma_{12}}{\Sigma_{11}} \mathbf{y}'_0 + 2\bar{c}\bar{\mathbf{v}}'}$, where $\mathbf{y}'_0 = \mathbf{z} - 2c\mathbf{v}'$ and $\bar{\mathbf{v}}' = \sum_{\hat{j} \in T_i} \lambda_{\hat{j}}^{\bar{lr}.SS} \mathbf{ss}'_{\hat{j}}$. Since $(b_1, \dots, b_\ell) \neq (\bar{b}_1, \dots, \bar{b}_\ell)$, we know $2 \in \mathcal{I}$ by Lemma 1. Since $c(\mathbf{v} - \mathbf{v}') + \bar{c}(\bar{\mathbf{v}} - \bar{\mathbf{v}}') \in \Lambda_q^+(A)$ by Lemma 12, we know $2c(\mathbf{v} - \mathbf{v}') + 2\bar{c}(\bar{\mathbf{v}} - \bar{\mathbf{v}}') \in 2\Lambda_q^+(A) \subset \mathcal{I} \otimes \Lambda_q^+(A)$, which implies $\mathcal{I} \otimes \Lambda_q^+(A) + \mathbf{z} - 2c\mathbf{v} + \mathbf{S}' + 2\bar{c}\bar{\mathbf{v}}$ and $\mathcal{I} \otimes \Lambda_q^+(A) + \mathbf{z} - 2c\mathbf{v}' + \mathbf{S}' + 2\bar{c}\bar{\mathbf{v}}'$ are the same lattice cosets. Also, since $b^\dagger b = 1$ for any $b \in \mathcal{S}_b$, we have $\Sigma_{11} = \Sigma_{22} = (1 + \ell)\sigma_r^2$. Let $w = \sum_{i \in [\ell]} \bar{b}_i b_i^*$. Since $\|w\|_1 \leq \ell$ and $w \neq \ell$, by Lemma 10, we have $\sigma_{\max}(\Sigma_{12}\Sigma_{21}) = \sigma_{\max}(\sigma_r^4(1+w)^*(1+w)) = \sigma_r^4 \sigma_{\max}((1+w)^*(1+w)) \leq \sigma_r^4((\ell+1)^2 - 2\ell/N^2)$. Therefore, $\sigma_{\min}(\Delta(\Sigma)) = \sigma_{\min}(\Sigma_{11}\Sigma_{22} - \Sigma_{21}\Sigma_{12}) = \sigma_{\min}((\ell+1)^2\sigma_r^4 - \Sigma_{21}\Sigma_{12}) \geq (\ell+1)^2\sigma_r^4 - \sigma_{\max}(\Sigma_{21}\Sigma_{12}) \geq 2\ell\sigma_r^4/N^2$, which implies $\sigma_{\min}(\Delta(\Sigma)/\Sigma_{11}) \geq \frac{2\ell}{N^2(\ell+1)}\sigma_r^2 \geq \sigma_r^2/N^2$. Since $2\Lambda_q^+(A) \subset \mathcal{I} \otimes \Lambda_q^+(A)$,

n	q	k	m	β_{sk}	σ_r	β_z	$ \text{pk} $	$ \text{sig} $	Comm.	$ \{\text{sk}_i\} /n$
5	2^{106}	7	31	2^{23}	$2^{93.12}$	$2^{103.62}$	47.55KB	258.10KB	1.49MB	610.68KB
32	2^{119}	8	50	2^{18}	$2^{105.06}$	$2^{117.25}$	60.73KB	440.32KB	2.02MB	16.60GB

Fig. 7. The concrete parameters and estimated efficiency for $\kappa = 128$ and $n = 5, 32$. In both cases, we use $(N, \ell, \beta_c) = (512, 26, 64)$. The last second column denotes the communication complexity per signer and the last column denotes the average secret key size.

$\eta_\varepsilon(\mathcal{I} \otimes A_q^\perp(A)) \leq 2\eta_\varepsilon(A_q^\perp(A)) \leq \sigma_r/N \leq \sqrt{\sigma_{\min}(\Delta(\Sigma)/\Sigma_{11})}$, by Lemma 8 and using weak-triangle inequality as in the case of the first execution, we have $\delta_{\alpha,j} \leq (1 + \varepsilon')^{1 + \frac{\alpha}{\alpha-1}} \left(\frac{1+\varepsilon}{1-\varepsilon}\right)^{\frac{\alpha}{\alpha-1}} \exp\left(\alpha\pi \frac{N^2 \left\| \frac{\Sigma_{12}}{\Sigma_{11}} 2c(v-v') + 2\bar{c}(\bar{v}-\bar{v}') \right\|^2}{\sigma_r^2}\right)$.

Also, by Lemma 12, $\|v - v'\| \leq \beta_{\text{ss}}\beta_{\text{sk}}\sqrt{mN}$ and $\|\bar{v} - \bar{v}'\| \leq \beta_{\text{ss}}\beta_{\text{sk}}\sqrt{mN}$, and since $\left\| 1 + \sum_{i \in [\ell]} \bar{b}_i^\dagger b_i \right\|_1 \leq 1 + \ell$, we know that $\left\| \frac{\Sigma_{21}}{\Sigma_{11}} \right\|_1 \leq \frac{\sigma_r^2(\ell+1)}{\sigma_r^2(\ell+1)} \leq 1$. Finally, by how σ_r is set in Fig. 5 and $\frac{\alpha}{\alpha-1} \leq 2$,

$$\delta_{\alpha,j} \leq (1 + \varepsilon')^3 \left(\frac{1 + \varepsilon}{1 - \varepsilon}\right)^2 \cdot e^{\alpha/(2q)}. \tag{3}$$

Since $\varepsilon = 2^{-2\kappa}$ and $\ell \leq 2\kappa$, $\varepsilon' \leq 8\ell \cdot 2^{-2\kappa}$ and $\frac{1+\varepsilon}{1-\varepsilon} \leq 1 + 4 \cdot 2^{-2\kappa}$. From the above analysis, $R_\alpha(D_0 \| D'_0) = 1$ and by Eqs. 2 and 3, for any $j \in [2q_s]$ and $x_{[0..j-1]}$, $R_\alpha\left(D_{j|x_{[0..j-1]}} \| D'_{j|x_{[0..j-1]}}\right) \leq (1 + 8\ell \cdot 2^{-2\kappa})^5 e^{\alpha/(2q)}$. Thus, by Lemma 6, $R_\alpha(W_{A,\text{sk},\rho,h} \| W_{A,f'_A(\text{sk},\rho),h}) \leq (1 + 8\ell \cdot 2^{-2\kappa})^{10q} e^\alpha \leq (1 + 160\ell q \cdot 2^{-2\kappa}) \cdot e^\alpha$. \square

4.4 Concrete Instantiation and Efficiency Analysis

We analyze the concrete efficiency of our protocol in the setting considered by [45], where the security parameter is $\kappa = 128$, the maximum number of signing sessions is $q_s = 2^{64}$ (following NIST recommendations and used in other related works [60]), and $n = 5$. We consider arbitrary threshold $1 \leq t \leq n$ here. We set $N = 512$ and $k = 7$. We set q such that the logarithm of β , the ℓ_2 -norm of the short solution, satisfies $\log \beta \leq 2\sqrt{kN} \log q \log \delta$, according to [57]. We use $\delta = 1.005$ as in [45] so that we get roughly 128-bit security of the MSIS problem. Note that we are not choosing the MSIS parameters according to the concrete bounds of Theorem 2, but rather we are choosing parameters so that MSIS gives 128 bits of security. This follows common practice, and it is justified by the fact that our bound is likely not tight due to the use of the Forking Lemma. We will see that the estimated $\beta \leq 2^{104.63}$, so we set $q \geq 2^{106}$. We set $\beta_{\text{sk}} = 2^{23}$ and then, according to Fig. 5, we set $\beta_c = 64$, $m = 33$, $\ell = 26$. We set $\sigma_r = 2^{93.12}$ due to the first term of the maximum function⁷ with $\beta_{\text{ss}} \approx 7200$ by Lemma

⁷ The second term is much smaller given the parameters we set.

13. Then, we set $\beta_z = 2^{103.63}$. By Theorem 2, we have that β is bounded by $2^{104.63}$. Then, our public key size is $|\text{pk}| = kN \log q = 47.55$ KB, the signature size is $|\text{sig}| = (m + k)N \log q = 258.10$ KB, the communication complexity per signer is $((\ell + 1)k + m)N \log q = 1.49$ MB, and the average secret key size is $|\{\text{sk}_i\}|/n = \lfloor n/2 \rfloor^{4.3} \log(n) \log(q) mN = 610.68$ KB. We summarize the parameters in Fig. 7, where we also show the concrete parameters and efficiency for $n = 32$ estimated in the same manner as above.

Acknowledgments. This research was partially supported by NSF grants CNS-2026774, CNS-2154174, a JP Morgan Faculty Award, a CISCO Faculty Award, and a gift from Microsoft.

References

1. Agrawal, S., Gentry, C., Halevi, S., Sahai, A.: Discrete Gaussian leftover hash lemma over infinite domains. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT 2013, Part I. LNCS, vol. 8269, pp. 97–116. Springer, Berlin, Heidelberg (Dec 2013). https://doi.org/10.1007/978-3-642-42033-7_6
2. Agrawal, S., Stehlé, D., Yadav, A.: Round-optimal lattice-based threshold signatures, revisited. In: Bojanczyk, M., Merelli, E., Woodruff, D.P. (eds.) ICALP 2022. LIPIcs, vol. 229, pp. 8:1–8:20. Schloss Dagstuhl (Jul 2022). <https://doi.org/10.4230/LIPIcs.ICALP.2022.8>
3. Applebaum, B., Nir, O., Pinkas, B.: How to recover a secret with $o(n)$ additions. In: Handschuh, H., Lysyanskaya, A. (eds.) CRYPTO 2023, Part I. LNCS, vol. 14081, pp. 236–262. Springer, Cham (Aug 2023). https://doi.org/10.1007/978-3-031-38557-5_8
4. Bacho, R., Loss, J.: On the adaptive security of the threshold BLS signature scheme. In: Yin, H., Stavrou, A., Cremers, C., Shi, E. (eds.) ACM CCS 2022. pp. 193–207. ACM Press (Nov 2022). <https://doi.org/10.1145/3548606.3560656>
5. Bacho, R., Loss, J., Tessaro, S., Wagner, B., Zhu, C.: Twinkle: Threshold signatures from DDH with full adaptive security. In: Joye, M., Leander, G. (eds.) EUROCRYPT 2024, Part I. LNCS, vol. 14651, pp. 429–459. Springer, Cham (May 2024). https://doi.org/10.1007/978-3-031-58716-0_15
6. Beimel, A.: Secure schemes for secret sharing and key distribution. PhD thesis, Israel Institute of Technology, Technion (1996)
7. Bellare, M., Crites, E.C., Komlo, C., Maller, M., Tessaro, S., Zhu, C.: Better than advertised security for non-interactive threshold signatures. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022, Part IV. LNCS, vol. 13510, pp. 517–550. Springer, Cham (Aug 2022). https://doi.org/10.1007/978-3-031-15985-5_18
8. Bellare, M., Tessaro, S., Zhu, C.: Stronger security for non-interactive threshold signatures: BLS and FROST. Cryptology ePrint Archive, Report 2022/833 (2022), <https://eprint.iacr.org/2022/833>
9. Benaloh, J.C., Leichter, J.: Generalized secret sharing and monotone functions. In: Goldwasser, S. (ed.) CRYPTO’88. LNCS, vol. 403, pp. 27–35. Springer, New York (Aug 1990). https://doi.org/10.1007/0-387-34799-2_3
10. Bendlin, R., Krehbiel, S., Peikert, C.: How to share a lattice trapdoor: Threshold protocols for signatures and (H)IBE. In: Jacobson Jr., M.J., Locasto, M.E., Mohassel, P., Safavi-Naini, R. (eds.) ACNS 13International Conference on Applied Cryptography and Network Security. LNCS, vol. 7954, pp. 218–236. Springer, Berlin, Heidelberg (Jun 2013). https://doi.org/10.1007/978-3-642-38980-1_14

11. Benhamouda, F., Camenisch, J., Krenn, S., Lyubashevsky, V., Neven, G.: Better zero-knowledge proofs for lattice encryption and their application to group signatures. In: Sarkar, P., Iwata, T. (eds.) ASIACRYPT 2014, Part I. LNCS, vol. 8873, pp. 551–572. Springer, Berlin, Heidelberg (Dec 2014). https://doi.org/10.1007/978-3-662-45611-8_29
12. Boldyreva, A.: Threshold signatures, multisignatures and blind signatures based on the gap-Diffie-Hellman-group signature scheme. In: Desmedt, Y. (ed.) PKC 2003. LNCS, vol. 2567, pp. 31–46. Springer, Berlin, Heidelberg (Jan 2003). https://doi.org/10.1007/3-540-36288-6_3
13. Boneh, D., Gennaro, R., Goldfeder, S.: Using level-1 homomorphic encryption to improve threshold DSA signatures for bitcoin wallet security. In: Lange, T., Dunkelman, O. (eds.) LATINCRYPT 2017. LNCS, vol. 11368, pp. 352–377. Springer, Cham (Sep 2019). https://doi.org/10.1007/978-3-030-25283-0_19
14. Boneh, D., Gennaro, R., Goldfeder, S., Jain, A., Kim, S., Rasmussen, P.M.R., Sahai, A.: Threshold cryptosystems from threshold fully homomorphic encryption. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, Part I. LNCS, vol. 10991, pp. 565–596. Springer, Cham (Aug 2018). https://doi.org/10.1007/978-3-319-96884-1_19
15. Boneh, D., Gennaro, R., Goldfeder, S., Kim, S.: A lattice-based universal thresholdizer for cryptographic systems. Cryptology ePrint Archive, Report 2017/251 (2017), <https://eprint.iacr.org/2017/251>
16. Boppana, R.B.: Amplification of probabilistic boolean formulas. In: 26th Annual Symposium on Foundations of Computer Science (sfcs 1985). pp. 20–29 (1985). <https://doi.org/10.1109/SFCS.1985.5>
17. Boschini, C., Takahashi, A., Tibouchi, M.: MuSig-L: Lattice-based multi-signature with single-round online phase. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022, Part II. LNCS, vol. 13508, pp. 276–305. Springer, Cham (Aug 2022). https://doi.org/10.1007/978-3-031-15979-4_10
18. Boudgoust, K., Scholl, P.: Simple threshold (fully homomorphic) encryption from LWE with polynomial modulus. In: Guo, J., Steinfeld, R. (eds.) ASIACRYPT 2023, Part I. LNCS, vol. 14438, pp. 371–404. Springer, Singapore (Dec 2023). https://doi.org/10.1007/978-981-99-8721-4_12
19. Canetti, R., Gennaro, R., Goldfeder, S., Makriyannis, N., Peled, U.: UC non-interactive, proactive, threshold ECDSA with identifiable aborts. In: Ligatti, J., Ou, X., Katz, J., Vigna, G. (eds.) ACM CCS 2020. pp. 1769–1787. ACM Press (Nov 2020) <https://doi.org/10.1145/3372297.3423367>
20. Chen, Y.: DualMS: Efficient lattice-based two-round multi-signature with trapdoor-free simulation. In: Handschuh, H., Lysyanskaya, A. (eds.) CRYPTO 2023, Part V. LNCS, vol. 14085, pp. 716–747. Springer, Cham (Aug 2023) https://doi.org/10.1007/978-3-031-38554-4_23
21. Cheon, J.H., Cho, W., Kim, J.: Improved universal thresholdizer from iterative shamir secret sharing. Cryptology ePrint Archive, Paper 2023/545 (2023), <https://eprint.iacr.org/2023/545>
22. Chu, H., Gerhart, P., Ruffing, T., Schröder, D.: Practical Schnorr threshold signatures without the algebraic group model. In: Handschuh, H., Lysyanskaya, A. (eds.) CRYPTO 2023, Part I. LNCS, vol. 14081, pp. 743–773. Springer, Cham (Aug 2023). https://doi.org/10.1007/978-3-031-38557-5_24
23. Connolly, D., Komlo, C., Goldberg, I., Wood, C.A.: Two-Round Threshold Schnorr Signatures with FROST. Internet-Draft draft-irtf-cfrg-frost-10, Internet Engineering Task Force (Sep 2022), <https://datatracker.ietf.org/doc/draft-irtf-cfrg-frost/10/>, work in Progress

24. Cozzo, D., Smart, N.P.: Sharing the LUOV: Threshold post-quantum signatures. In: Albrecht, M. (ed.) 17th IMA International Conference on Cryptography and Coding. LNCS, vol. 11929, pp. 128–153. Springer, Cham (Dec 2019). https://doi.org/10.1007/978-3-030-35199-1_7
25. Cramer, R., Fehr, S.: Optimal black-box secret sharing over arbitrary Abelian groups. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 272–287. Springer, Berlin, Heidelberg (Aug 2002). https://doi.org/10.1007/3-540-45708-9_18
26. Crites, E.C., Komlo, C., Maller, M.: Fully adaptive Schnorr threshold signatures. In: Handschuh, H., Lysyanskaya, A. (eds.) CRYPTO 2023, Part I. LNCS, vol. 14081, pp. 678–709. Springer, Cham (Aug 2023). https://doi.org/10.1007/978-3-031-38557-5_22
27. Crites, E.C., Komlo, C., Maller, M., Tessaro, S., Zhu, C.: Snowblind: A threshold blind signature in pairing-free groups. In: Handschuh, H., Lysyanskaya, A. (eds.) CRYPTO 2023, Part I. LNCS, vol. 14081, pp. 710–742. Springer, Cham (Aug 2023)https://doi.org/10.1007/978-3-031-38557-5_23
28. Damgård, I., Koprowski, M.: Practical threshold RSA signatures without a trusted dealer. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 152–165. Springer, Berlin, Heidelberg (May 2001). https://doi.org/10.1007/3-540-44987-6_10
29. Damgård, I., Orlandi, C., Takahashi, A., Tibouchi, M.: Two-round n-out-of-n and multi-signatures and trapdoor commitment from lattices. In: Garay, J. (ed.) PKC 2021, Part I. LNCS, vol. 12710, pp. 99–130. Springer, Cham (May 2021). https://doi.org/10.1007/978-3-030-75245-3_5
30. Damgård, I., Thorbek, R.: Linear integer secret sharing and distributed exponentiation. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T. (eds.) PKC 2006. LNCS, vol. 3958, pp. 75–90. Springer, Berlin, Heidelberg (Apr 2006). https://doi.org/10.1007/11745853_6
31. De Santis, A., Desmedt, Y., Frankel, Y., Yung, M.: How to share a function securely. In: 26th ACM STOC. pp. 522–533. ACM Press (May 1994). <https://doi.org/10.1145/195058.195405>
32. Desmedt, Y.: Society and group oriented cryptography: A new concept. In: Pomerance, C. (ed.) CRYPTO’87. LNCS, vol. 293, pp. 120–127. Springer, Berlin, Heidelberg (Aug 1988). https://doi.org/10.1007/3-540-48184-2_8
33. Desmedt, Y., Frankel, Y.: Threshold cryptosystems. In: Brassard, G. (ed.) CRYPTO’89. LNCS, vol. 435, pp. 307–315. Springer, New York (Aug 1990). https://doi.org/10.1007/0-387-34805-0_28
34. Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schwabe, P., Seiler, G., Stehlé, D.: CRYSTALS-Dilithium: A lattice-based digital signature scheme. IACR TCHES **2018**(1), 238–268 (2018). <https://doi.org/10.13154/tches.v2018.i1.238-268>, <https://tches.iacr.org/index.php/TCHES/article/view/839>
35. Espitau, T., Katsumata, S., Takemure, K.: Two-round threshold signature from algebraic one-more learning with errors. In: Reyzin, L., Stebila, D. (eds.) CRYPTO 2024, Part VII. LNCS, vol. 14926, pp. 387–424. Springer, Cham (Aug 2024). https://doi.org/10.1007/978-3-031-68394-7_13
36. Espitau, T., Niot, G., Prest, T.: Flood and submerge: Distributed key generation and robust threshold signature from lattices. In: Reyzin, L., Stebila, D. (eds.) CRYPTO 2024, Part VII. LNCS, vol. 14926, pp. 425–458. Springer, Cham (Aug 2024). https://doi.org/10.1007/978-3-031-68394-7_14

37. Genise, N., Micciancio, D., Peikert, C., Walter, M.: Improved discrete gaussian and subgaussian analysis for lattice cryptography. In: Kiayias, A., Kohlweiss, M., Wallden, P., Zikas, V. (eds.) PKC 2020, Part I. LNCS, vol. 12110, pp. 623–651. Springer, Cham (May 2020). https://doi.org/10.1007/978-3-030-45374-9_21
38. Gennaro, R., Goldfeder, S.: Fast multiparty threshold ECDSA with fast trustless setup. In: Lie, D., Mannan, M., Backes, M., Wang, X. (eds.) ACM CCS 2018. pp. 1179–1194. ACM Press (Oct 2018). <https://doi.org/10.1145/3243734.3243859>
39. Gennaro, R., Goldfeder, S., Narayanan, A.: Threshold-optimal DSA/ECDSA signatures and an application to bitcoin wallet security. In: Manulis, M., Sadeghi, A.R., Schneider, S. (eds.) ACNS 16International Conference on Applied Cryptography and Network Security. LNCS, vol. 9696, pp. 156–174. Springer, Cham (Jun 2016). https://doi.org/10.1007/978-3-319-39555-5_9
40. Gennaro, R., Jarecki, S., Krawczyk, H., Rabin, T.: Robust threshold DSS signatures. In: Maurer, U.M. (ed.) EUROCRYPT’96. LNCS, vol. 1070, pp. 354–371. Springer, Berlin, Heidelberg (May 1996). https://doi.org/10.1007/3-540-68339-9_31
41. Gennaro, R., Jarecki, S., Krawczyk, H., Rabin, T.: Secure applications of Pedersen’s distributed key generation protocol. In: Joye, M. (ed.) CT-RSA 2003. LNCS, vol. 2612, pp. 373–390. Springer, Berlin, Heidelberg (Apr 2003). https://doi.org/10.1007/3-540-36563-X_26
42. Gennaro, R., Jarecki, S., Krawczyk, H., Rabin, T.: Secure distributed key generation for discrete-log based cryptosystems. *Journal of Cryptology* **20**(1), 51–83 (Jan 2007). <https://doi.org/10.1007/s00145-006-0347-3>
43. Gennaro, R., Rabin, T., Jarecki, S., Krawczyk, H.: Robust and efficient sharing of RSA functions. *Journal of Cryptology* **13**(2), 273–300 (2000). <https://doi.org/10.1007/s001459910011>
44. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Ladner, R.E., Dwork, C. (eds.) 40th ACM STOC. pp. 197–206. ACM Press (May 2008). <https://doi.org/10.1145/1374376.1374407>
45. Gür, K.D., Katz, J., Silde, T.: Two-round threshold lattice-based signatures from threshold homomorphic encryption. In: Saarinen, M.J., Smith-Tone, D. (eds.) Post-Quantum Cryptography - 15th International Workshop, PQCrypto 2024, Part II. pp. 266–300. Springer, Cham (Jun 2024). https://doi.org/10.1007/978-3-031-62746-0_12
46. Hauck, E., Kiltz, E., Loss, J., Nguyen, N.K.: Lattice-based blind signatures, revisited. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020, Part II. LNCS, vol. 12171, pp. 500–529. Springer, Cham (Aug 2020). https://doi.org/10.1007/978-3-030-56880-1_18
47. Hoory, S., Magen, A., Pitassi, T.: Monotone circuits for the majority function. In: International Workshop on Approximation Algorithms for Combinatorial Optimization. pp. 410–425. Springer (2006). https://doi.org/10.1007/11830924_38
48. Karchmer, M., Wigderson, A.: On span programs. In: [1993] Proceedings of the Eighth Annual Structure in Complexity Theory Conference. pp. 102–111 (1993). <https://doi.org/10.1109/SCT.1993.336536>
49. Katsumata, S., Reichle, M., Takemure, K.: Adaptively secure 5 round threshold signatures from MLWE/MSIS and DL with rewinding. In: Reyzin, L., Stebila, D. (eds.) CRYPTO 2024, Part VII. LNCS, vol. 14926, pp. 459–491. Springer, Cham (Aug 2024). https://doi.org/10.1007/978-3-031-68394-7_15

50. Kiltz, E., Lyubashevsky, V., Schaffner, C.: A concrete treatment of Fiat-Shamir signatures in the quantum random-oracle model. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018, Part III. LNCS, vol. 10822, pp. 552–586. Springer, Cham (Apr / May 2018). https://doi.org/10.1007/978-3-319-78372-7_18
51. Komlo, C., Goldberg, I.: FROST: Flexible round-optimized Schnorr threshold signatures. In: Dunkelman, O., Jr., M.J.J., O’Flynn, C. (eds.) SAC 2020. LNCS, vol. 12804, pp. 34–65. Springer, Cham (Oct 2020). https://doi.org/10.1007/978-3-030-81652-0_2
52. Lindell, Y.: Simple three-round multiparty Schnorr signing with full simulatability. *CiC* 1(1), 25 (2024). <https://doi.org/10.62056/a36c015vt>
53. Lindell, Y., Nof, A., Ranellucci, S.: Fast secure multiparty ECDSA with practical distributed key generation and applications to cryptocurrency custody. *Cryptology ePrint Archive, Report 2018/987* (2018), <https://eprint.iacr.org/2018/987>
54. Lyubashevsky, V.: Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 598–616. Springer, Berlin, Heidelberg (Dec 2009). https://doi.org/10.1007/978-3-642-10366-7_35
55. Lyubashevsky, V.: Lattice signatures without trapdoors. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 738–755. Springer, Berlin, Heidelberg (Apr 2012). https://doi.org/10.1007/978-3-642-29011-4_43
56. Lyubashevsky, V., Ducas, L., Kiltz, E., Lepoint, T., Schwabe, P., Seiler, G., Stehlé, D., Bai, S.: CRYSTALS-DILITHIUM. Tech. rep., National Institute of Standards and Technology (2022), available at <https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022>
57. Micciancio, D., Regev, O.: Lattice-based cryptography. In: *Post-quantum cryptography*, pp. 147–191. Springer (2009)
58. National Institute of Standards and Technology: Multi-Party Threshold Cryptography (2018–Present), <https://csrc.nist.gov/Projects/threshold-cryptography>
59. Nick, J., Ruffing, T., Seurin, Y.: MuSig2: Simple two-round Schnorr multi-signatures. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021, Part I. LNCS, vol. 12825, pp. 189–221. Springer, Cham, Virtual Event (Aug 2021). https://doi.org/10.1007/978-3-030-84242-0_8
60. Pino, R.D., Katsumata, S., Maller, M., Mouhartem, F., Prest, T., Saarinen, M.J.O.: Threshold raccoon: Practical threshold signatures from standard lattice assumptions. In: Joye, M., Leander, G. (eds.) EUROCRYPT 2024, Part II. LNCS, vol. 14652, pp. 219–248. Springer, Cham (May 2024). https://doi.org/10.1007/978-3-031-58723-8_8
61. Prest, T., Fouque, P.A., Hoffstein, J., Kirchner, P., Lyubashevsky, V., Pornin, T., Ricosset, T., Seiler, G., Whyte, W., Zhang, Z.: FALCON. Tech. rep., National Institute of Standards and Technology (2022), available at <https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022>
62. Rényi, A.: On measures of entropy and information. In: *Proceedings of the fourth Berkeley symposium on mathematical statistics and probability, volume 1: contributions to the theory of statistics. vol. 4*, pp. 547–562. University of California Press (1961)
63. Rosser, J.B., Schoenfeld, L.: Approximate formulas for some functions of prime numbers. *Illinois Journal of Mathematics* 6(1), 64–94 (1962). <https://doi.org/10.1215/ijm/1255631807>
64. Rossi, M.: Extended security of lattice-based cryptography. Ph.D. thesis, Université Paris sciences et lettres (2020)

65. Shoup, V.: Practical threshold signatures. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 207–220. Springer, Berlin, Heidelberg (May 2000). https://doi.org/10.1007/3-540-45539-6_15
66. Stinson, D.R., Strobl, R.: Provably secure distributed Schnorr signatures and a (t, n) threshold scheme for implicit certificates. In: Varadharajan, V., Mu, Y. (eds.) ACISP 01. LNCS, vol. 2119, pp. 417–434. Springer, Berlin, Heidelberg (Jul 2001). https://doi.org/10.1007/3-540-47719-5_33
67. Takashima, K., Takayasu, A.: Tighter security for efficient lattice cryptography via the Rényi divergence of optimized orders. In: Au, M.H., Miyaji, A. (eds.) ProvSec 2015. LNCS, vol. 9451, pp. 412–431. Springer, Cham (Nov 2015). https://doi.org/10.1007/978-3-319-26059-4_23
68. Tessaro, S., Zhu, C.: Threshold and multi-signature schemes from linear hash functions. In: Hazay, C., Stam, M. (eds.) EUROCRYPT 2023, Part V. LNCS, vol. 14008, pp. 628–658. Springer, Cham (Apr 2023). https://doi.org/10.1007/978-3-031-30589-4_22
69. Valiant, L.: Short monotone formulae for the majority function. *Journal of Algorithms* **5**(3), 363–366 (1984). [https://doi.org/10.1016/0196-6774\(84\)90016-6](https://doi.org/10.1016/0196-6774(84)90016-6)



Lova: Lattice-Based Folding Scheme from Unstructured Lattices

Giacomo Fenzi¹(✉), Christian Knabenhans¹, Ngoc Khanh Nguyen²,
and Duc Tu Pham³

¹ EPFL, Lausanne, Switzerland
{giacomo.fenzi, christian.knabehans}@epfl.ch,
ngoc.khanh.nguyen@kcl.ac.uk

² King's College London, London, UK

³ ENS Paris, Paris, France

Abstract. Folding schemes (Kothapalli et al., CRYPTO 2022) are a conceptually simple, yet powerful cryptographic primitive that can be used as a building block to realise incrementally verifiable computation (IVC) with low recursive overhead without general-purpose non-interactive succinct arguments of knowledge (SNARK). Most folding schemes known rely on the hardness of the discrete logarithm problem, and thus are both not quantum-resistant and operate over large prime fields. Existing post-quantum folding schemes (Boneh, Chen, ePrint 2024/257) based on lattice assumptions instead are secure under structured lattice assumptions, such as the Module Short Integer Solution Assumption (MSIS), which also binds them to relatively complex arithmetic. In contrast, we construct Lova, the first folding scheme whose security relies on the (unstructured) SIS assumption. We provide a Rust implementation of Lova, which makes only use of arithmetic in hardware-friendly power-of-two moduli. Crucially, this avoids the need of implementing and performing any finite field arithmetic. At the core of our results lies a new *exact* Euclidean norm proof which might be of independent interest.

1 Introduction

Incrementally verifiable computation [Val08] (IVC) is a cryptographic primitive that allows a long (possibly infinite) computation to be run, such that correctness of the state of the computation can be efficiently verified at any point. IVC and its generalisation, proof-carrying data [CT10] (PCD), have found numerous applications in succinct blockchains [BMRS20, BGH19, Mina], verifiable delay functions [BBBF18, KMT22], SNARKs for machine computations [BCTV14], and more.

Originally, IVC and PCD were built on recursive SNARKs [BCCT13, BCTV14, Val08] which prove that: (i) the current computation step was executed correctly, and (ii) there exists a proof that the computation was performed correctly for all previous steps up to that point. This approach, however, suffers from several restrictions on the choice of the underlying SNARKs, making the approach rather impractical. More recent constructions of IVC and PCD were

proposed from so-called folding and (split-)accumulation schemes [BCMS20, BCL+21, BC23, KST22, KS22, KS23]. Informally, a folding scheme “folds” several instances of a certain relation into a single instance, so that correctness of the folded instance implies correctness of all original instances. Until recently, folding (aka accumulation) schemes are instantiated using Pedersen commitments, and their security holds in the random oracle model under the discrete logarithm assumption. Consequently, all the constructions are currently exposed to efficient quantum attacks [Sho94].

Given the recent announcement of the US National Institute of Standardisation and Technology (NIST) on the post-quantum standardisation effort [NIST], it is becoming more and more likely that lattices will form the future foundation of public-key cryptography. Hence, a natural question arises as to whether folding schemes can be efficiently realised from lattice-based assumptions.

1.1 Our Results

In this paper, we present Lova¹, the first folding scheme based on unstructured lattice assumptions, i.e. the Short Integer Solution (SIS) assumption. Our construction brings the following benefits over relying on more structured assumptions, such as Module-SIS [LS15]. It allows for much simpler (yet efficient) instantiations of the folding scheme, without implementing polynomial ring arithmetic and requiring NTT-friendly prime moduli while relying on a more established computational assumption.

Our starting point is a generic construction of a folding scheme from Nova [KST22], which requires an additively homomorphic compressing commitment scheme. The rough intuition can be described as follows; the folding scheme focuses on “commit-and-prove”-type relations:

$$R := \{((x, t), (w, r)) : (x, w) \in R \wedge t = (w; r)\},$$

where R is a binary NP relation. Further, given two valid instances $(\mathbf{x}_0, \mathbf{w}_0)$ and $(\mathbf{x}_1, \mathbf{w}_1) \in R$, the folded instance $(\mathbf{x}^* := (x^*, t^*), \mathbf{w}^* := (w^*, r^*)) \in R$ is constructed by taking a linear combination of $(\mathbf{x}_0, \mathbf{w}_0)$ and $(\mathbf{x}_1, \mathbf{w}_1)$ with challenges generated by the verifier, or in the non-interactive case, output values of the random oracle. Thus, one could naively obtain a lattice-based folding scheme by instantiating R , with the folklore Ajtai commitment scheme [Ajt96].

The resulting construction, unfortunately, comes with a major efficiency drawback. Indeed, Ajtai commitments are binding only with respect to short message and randomness vectors. This limitation becomes particularly problematic because the norm of the folded witness \mathbf{w}^* increases after each folding step. The consequences are twofold. First, a maximal number of folding steps must be known ahead of setting the lattice parameters. This is contradictory to the concept of IVC, where we do consider long, and possibly infinite, computations. Second, the extracted message w^* may not be a valid witness for x^* with

¹ The name comes from the fact that our construction is a direct lattice adaptation of the Nova folding scheme [KST22].

respect to the relation R , due to slack and other related norm growth problems [BLNS20, ACK21, AL21]. In this work, we incorporate two main techniques to circumvent these limitations.

Decompose-and-Fold. First, we apply the (folklore by now) “decompose-and-fold” paradigm [PSTY13, BS23, BC24] which allows us to control the norm growth during an honest execution. Intuitively, given a witness w_i of norm at most β , where $i \in \{0, 1\}$, the prover starts by decomposing it (usually w.r.t. some decomposition base b) into many intermediate witnesses $w_{i,1}, \dots, w_{i,k}$, where each $w_{i,j}$ has much smaller norm than w_i . Afterwards, the prover folds all the $2k$ intermediate witnesses $(w_{i,j})_{i \in \{0,1\}, j \in [k]}$ into the final witness w^* . By picking appropriate parameters b and β , one can ensure that norm of the folded witness w^* is also bounded by β ; thus no norm growth occurs when following the protocol honestly.

Exact Euclidean Norm Proof. The second component is a new *exact* Euclidean norm proof. This ingredient ensures that no slack and stretch occurs in the knowledge soundness/extractability argument. Combined with the decompose-and-fold approach, this enables us to build a lattice-based folding scheme, where the number of folding steps is independent of the instantiated lattice parameters. We highlight that our Euclidean norm proof could be of independent interest, and may be applied in the context of lattice-based succinct arguments with fast verification, e.g., in the recent polynomial commitment scheme by Cini et al. [CMNW24].

To showcase the simplicity and practicality of our folding scheme, we provide a concrete instantiation and a proof-of-concept implementation. The Lova protocol is relatively simple and relies on unstructured assumptions, which makes it particularly easy to implement and straightforward to parallelize. Both our prover and verifier mostly perform linear algebra operations (especially matrix-matrix multiplication with bounded-norm entries), and we do not require more complex operations that appear in other lattice-based constructions (e.g., number-theoretic transforms for polynomial arithmetic, or sumcheck-style computations). In addition, we are able to choose the lattice modulus to be a hardware-friendly power-of-two ($q = 2^{64}$ in our evaluation), which eschews modular arithmetic altogether and reduces to standard integer arithmetic.

1.2 Technical Overview

We provide a brief overview of our techniques.

1.2.1 Background

Ajtai Commitment. In the Ajtai commitment scheme [Ajt96], one commits to a short vector $\mathbf{s} \in \mathbb{Z}^m$ by computing

$$\mathbf{A}\mathbf{s} \equiv \mathbf{t} \pmod{q} .$$

In the above q and $\mathbf{A} \in \mathbb{Z}^{n \times m}$ are public parameters of the scheme, and $\mathbf{t} \in \mathbb{Z}^n$ is the commitment to \mathbf{s} . That Ajtai commitments are binding follows directly

from the SIS assumptions, as two distinct short vectors \mathbf{s}, \mathbf{s}^* that satisfy the above equation imply that $\mathbf{s} - \mathbf{s}^* \neq \mathbf{0}$ is also short and $\mathbf{A}(\mathbf{s} - \mathbf{s}^*) = \mathbf{0} \pmod{q}$.

Reductions of Knowledge. Reductions of Knowledge (RoK) [KST22] are interactive protocols between a prover and a verifier that reduce checking membership of an instance in a relation to checking membership a related instance in a (usually simpler) relation. In a reduction of knowledge from R to R' , the prover and the verifier have access to an index i and an instance \mathbf{x} . The honest prover additionally has access to a witness \mathbf{w} for the instance. They interact and at the end of the interaction:

- If $(i, \mathbf{x}, \mathbf{w}) \in R$, the verifier accepts and outputs an instance \mathbf{x}' and the prover outputs a witness \mathbf{w}' such that $(i, \mathbf{x}', \mathbf{w}') \in R'$.
- If at the end of the interaction the verifier accepts and outputs an instance \mathbf{x}' , there is an efficient extractor that given $(i, \mathbf{x}, \mathbf{x}')$ and \mathbf{w}' such that $(i, \mathbf{x}', \mathbf{w}') \in R'$ outputs \mathbf{w} such that $(i, \mathbf{x}, \mathbf{w}) \in R$

A folding scheme is then simply reduction of knowledge from a relation R^2 to itself. Note that both completeness and (knowledge) soundness require then that the updated witness belongs to the same relation and that the extracted witness belong to the original relation. For the lattice setting, where norm growth and slack tend to accrue, this is the major technical hurdle to solve. The relation that we consider is the following, which is a slight generalization of the natural opening relation for Ajtai commitments²:

$$R_{q,\beta,t}^{\text{SIS}} := \left\{ (\mathbf{A}, \mathbf{T}, \mathbf{S}) \in \mathbb{Z}^{n \times m} \times \mathbb{Z}^{n \times t} \times \mathbb{Z}^{m \times t} \mid \begin{array}{l} \mathbf{AS} \equiv \mathbf{T} \pmod{q} \\ \forall i \in [t], \|\mathbf{S}_{*,i}\| \leq \beta \end{array} \right\} .$$

Since an instance of $(R_{q,\beta,t}^{\text{SIS}})^2$ can be reduced to one of $R_{q,\beta,2t}^{\text{SIS}}$, we consider designing a RoK for $R_{q,\beta,2t}^{\text{SIS}} \rightarrow R_{q,\beta,t}^{\text{SIS}}$.

1.2.2 A Naive Attempt to Folding Schemes

As in previous folding approaches, we will aim to do so via a random linear combinations, which will inevitably incur into problems. Let $(\mathbf{A}, \mathbf{T}, \mathbf{S}) \in R_{q,\beta,2t}^{\text{SIS}}$. The (naive) protocol that we design is the following:

1. The verifier samples a challenge $\mathbf{C} \leftarrow \mathcal{C}^{2t \times t} \subseteq \mathbb{Z}^{2t \times t}$ (from a yet unspecified sampling set) and send it to the prover.
2. The prover computes and outputs the updated witness $\mathbf{Z} := \mathbf{SC}$.
3. The verifier computes the updated instance $\mathbf{T}' := \mathbf{TC}$, accepts and outputs it.

This protocol suffers from two main issues:

Completeness Norm Growth. Folding must reduce checking two instances of a relation to checking a single instance of the *same* relation. In this case, the new opening \mathbf{Z} will *not* in fact satisfy $\|\mathbf{Z}_{*,i}\| \leq \beta$ for any non-trivial sampling set \mathcal{C} .

² Which we can recover by setting $t = 1$. We use this formulation as it will notationally more convenient later on.

Extraction Norm Growth. The protocol is knowledge sound, as we can construct an extractor that produces a (relaxed) witness via coordinate-wise special soundness [BBC+18, FMN23]. Interpreting the challenge set $\mathcal{C}^{2t \times t} \cong (\mathcal{C}^t)^{2t}$ (i.e. so that each coordinate correspond to a row of the matrix) the extractor is given access to a tree of $2t + 1$ accepting transcripts,

$$\left(\left(\begin{array}{c} \mathbf{C}^{(0)} \\ \mathbf{Z}^{(0)} \end{array} \right), \dots, \left(\begin{array}{c} \mathbf{C}^{(2t)} \\ \mathbf{Z}^{(2t)} \end{array} \right) \right),$$

such that, for $j \in [2t]$, $\mathbf{C}^{(0)}, \mathbf{C}^{(j)}$ differ in exactly row j . Letting i^* denote the column in which the two differ we have that $C_{j,i^*}^{(0)} \neq C_{j,i^*}^{(j)}$ and $C_{j',i}^{(0)} = C_{j',i}^{(j')}$ for $i \in [t]$ and $j' \neq j$. For $j \in [2t]$, the extractor computes

$$\mathbf{S}_{*,j} \equiv \frac{\mathbf{Z}_{*,j}^{(0)} - \mathbf{Z}_{*,j}^{(j)}}{C_{j,i^*}^{(0)} - C_{j,i^*}^{(j)}} \pmod{q},$$

and sets $\mathbf{S} := [\mathbf{S}_{*,1}, \dots, \mathbf{S}_{*,2t}]$. It is easy to see then that, for every $j \in [2t]$,

$$\mathbf{A}\mathbf{S}_{*,j} \equiv \frac{\mathbf{A}\mathbf{Z}_{*,j}^{(0)} - \mathbf{A}\mathbf{Z}_{*,j}^{(j)}}{C_{j,i^*}^{(0)} - C_{j,i^*}^{(j)}} \equiv \mathbf{T}_{*,j} \pmod{q}.$$

What is left is to bound the norm of the extracted witness \mathbf{S} . Letting $\beta_{\mathcal{C}} := \max_{c \neq c' \in \mathcal{C}} \|(c - c')^{-1} \pmod{q}\|$, and β' denote the completeness norm bound on \mathbf{Z} , we can only conclude that norm of $\mathbf{S}_{*,j}$ is at most $2 \cdot \beta_{\mathcal{C}} \cdot \beta' > \beta$. So, even if there were no completeness norm growth and $\beta' = \beta$ (which as argued before, is not currently the case), the extraction incurs in a norm blowup. The particularly hard term to control is $\beta_{\mathcal{C}}$. Selecting \mathcal{C} that simultaneously is (i) large enough for soundness; (ii) with elements of small norm (to keep the completeness norm under control); (iii) and with $\beta_{\mathcal{C}}$ small is challenging. In polynomial rings, setting \mathcal{C} to be the monomials can partially help, but there are limitations even in the cyclotomic ring setting [AL21].

To construct an efficient folding scheme for Ajtai commitments, we have to solve both of the above problems.

- To solve the completeness norm growth, we will ask the prover to decompose its opening and send us an updated commitment, which we can check for consistency against the old commitment.
- To solve the extraction norm growth, we will proceed in steps. First, we will present an approach to extract (a decomposed witness) with almost no extraction blowup, and then we will augment this protocol with a proof of exact norm that allows it to eliminate it completely.

1.2.3 Extracting Witness with Small Norm

We now aim to choose a challenge set \mathcal{C} suitable for both keeping completeness and extraction norm growth under control. A natural choice is the set of binary challenges $\mathcal{C} = \{0, 1\}$ as used in [BBC+18, CMNW24]. Then, as demonstrated in

the aforementioned works, we have $\beta_C = 1$ and norm of the extracted matrix is at most $2\beta'$. Below, we consider a slight extension of this approach, which later will be crucial to prove *exact* norm bounds.

Namely, consider ternary challenges $\mathcal{C} = \{-1, 0, 1\}$ instead. As in the binary case, those challenges are small, so they will contribute little to the completeness growth. For extraction, recall that norm growth was in large part contributed by the term β_C . For our choice of \mathcal{C} , the differences of challenges consists of $\delta := \alpha - \alpha'$ with $\alpha, \alpha' \in \{-1, 0, 1\}$ and $\alpha \neq \alpha'$. We notice that $\delta \in \{\pm 1, \pm 2\}$ and further equals 2 only if $(\alpha, \alpha') = (\pm 1, \mp 1)$. When $\delta = \pm 1$, dividing by δ does not create *any norm blowup*, similarly as in the binary case. On the other hand, for $\delta = \pm 2$, it is unclear whether the extracted witness is short, or even if it is well-defined, e.g. for even moduli q .

To leverage this observation, we revisit the coordinate-wise special soundness (CWSS) property and the heavy-row analysis in [BBC+18, Lemma 3]. For each coordinate i , we construct an extractor that recovers two accepting transcripts $(\mathbf{C}, \mathbf{Z}), (\mathbf{C}', \mathbf{Z}')$ such that: (i) \mathbf{C} and \mathbf{C}' differ exactly, and only, in the i -th row, and (ii) their corresponding i -th row vectors $\mathbf{c}_i, \mathbf{c}'_i$ satisfy $\mathbf{c}_i \not\equiv \mathbf{c}'_i \pmod{2}$. The latter condition makes sure that there exists an entry of $\mathbf{c}_i - \mathbf{c}'_i$ which is ± 1 and allows for extracting a witness with norm at most $2\beta'$ as in the binary setting.

Roughly, the analysis relies on the heavy-row argument [Dam10]. Suppose a cheating prover succeeds to produce a valid response \mathbf{Z} for a random challenge matrix \mathbf{C} with a noticeable probability. Then, for any coordinate i , with sufficiently large probability (i.e. the probability of “landing in a heavy row”), the set of matrix challenges \mathbf{C}' , which satisfy conditions (i) and (ii) described above, that are simultaneously “good” (in the sense that the prover outputs an accepting transcript) must be big enough.

Replicating the CWSS analysis with the improved extraction procedure to the strawman protocol, we reduce the extraction norm blowup of the strawman protocol to $2 \cdot \beta'$. We highlight that the new approach suffers from a larger soundness than in the binary challenge setting, which is now roughly $(\frac{2}{3})^t$.

1.2.4 Almost a Folding Scheme

Following the above strategy, we design a folding scheme with no completeness blowup. Further, we use the extraction strategy previously described to extract a very short (decomposed) witness, which we later show how to upgrade to extract a witness with no extraction norm blowup.

b -Decomposition. In the sequel \mathbf{G} is the b -decomposition gadget matrix, and \mathbf{G}^{-1} denote its inverse, i.e. $\mathbf{G}^{-1}(\mathbf{S})\mathbf{G} = \mathbf{S}$ for every \mathbf{S} . \mathbf{G}^{-1} decomposes \mathbf{S} into a matrix $\tilde{\mathbf{S}}$ where each entry is in $[-\lfloor b/2 \rfloor, \lfloor b/2 \rfloor]$ (in this work, we use balanced base- b decomposition).

Folding Scheme. Let $(\mathbf{A}, \mathbf{T}, \mathbf{S}) \in \mathcal{R}_{q, \beta, 2t}^{\text{SIS}}$. The new protocol that we design is the following:

1. The prover computes $\tilde{\mathbf{S}} := \mathbf{G}^{-1}(\mathbf{S}), \tilde{\mathbf{T}} := \mathbf{A}\tilde{\mathbf{S}} \pmod{q}$ and sends $\tilde{\mathbf{T}}$ to the verifier.
2. The verifier samples a challenge $\mathbf{C} \leftarrow \{-1, 0, 1\}^{2kt \times t}$ and send it to the prover.

3. The prover computes and outputs the updated witness $\mathbf{Z} := \tilde{\mathbf{S}}\mathbf{C}$.
4. The verifier computes $\mathbf{T}' := \tilde{\mathbf{T}}\mathbf{C}$, accepts if

$$\tilde{\mathbf{T}}\mathbf{G} \equiv \mathbf{T} \pmod{q} ,$$

and outputs the updated instance \mathbf{T}' .

We analyse completeness and knowledge soundness of the above RoK. **Completeness** First, it is easy to see that the verifier’s algebraic checks succeed.

$$\begin{aligned} \tilde{\mathbf{T}}\mathbf{G} &\equiv \mathbf{A}\tilde{\mathbf{S}}\mathbf{G} \equiv \mathbf{A}\mathbf{S} \equiv \mathbf{T} \pmod{q} , \\ \mathbf{A}\mathbf{Z} &\equiv \mathbf{A}\tilde{\mathbf{S}}\mathbf{C} \equiv \tilde{\mathbf{T}}\mathbf{C} \pmod{q} . \end{aligned}$$

We are left to check the norms of \mathbf{Z} . Let $i \in [t]$, and consider $\|\mathbf{Z}_{*,i}\|$. Since $\|\tilde{\mathbf{S}}_{*,j}\| \leq \lfloor \frac{b}{2} \rfloor \sqrt{m}$, we have that

$$\|\mathbf{Z}_{*,j}\| \leq \left\| \sum_{i=1}^{2kt} \mathbf{C}_{i,j} \tilde{\mathbf{S}}_{*,i} \right\| \leq 2kt \left\lfloor \frac{b}{2} \right\rfloor \sqrt{m} .$$

As long as $t \leq \frac{\beta}{2k \lfloor \frac{b}{2} \rfloor \sqrt{m}}$, the above norm is then bounded above by β .

Relaxed Knowledge Soundness. We apply a similar analysis to that in the strawman protocol, except now that the extraction procedure is applied on $2kt+1$ coordinates instead of $2t+1$. This recovers a decomposed witness $\tilde{\mathbf{S}} \in \mathbb{Z}^{n \times 2kt}$ which has $\|\tilde{\mathbf{S}}_{*,j}\| \leq 2\beta$ and for which $\mathbf{A}\tilde{\mathbf{S}} \equiv \tilde{\mathbf{T}} \pmod{q}$. Later on, we will make use of this intermediate short extracted witness. The final extracted witness is $\mathbf{S} := \tilde{\mathbf{S}}\mathbf{G}$ which satisfies

$$\mathbf{A}\mathbf{S} \equiv \mathbf{A}\tilde{\mathbf{S}}\mathbf{G} \equiv \tilde{\mathbf{T}}\mathbf{G} \equiv \mathbf{T} \pmod{q} .$$

Note that, for $j \in [2t]$, $\|\mathbf{S}_{*,j}\| \leq 2\beta^2$.

1.2.5 Exact Euclidean Norm Proof

To construct the final protocol, we require to augment the above protocol with a proof of exact norm. Our first observation is that, if for every $j \in [2t]$ $\|\mathbf{S}_{*,j}\| \leq \beta$ then the matrix $\mathbf{D} := \mathbf{S}^\top \mathbf{S}$ has a diagonal bounded by β^2 , i.e. for every $i \in [2t]$, has $D_{i,i} \leq \beta^2$. This is because

$$D_{i,i} = \langle \mathbf{S}_{*,i}, \mathbf{S}_{*,i} \rangle = \|\mathbf{S}_{*,i}\|^2 \leq \beta^2 .$$

We then rewrite the relation for opening of Ajtai commitments to:

$$\mathbf{R}_{q,\beta,t} := \left\{ \left(\mathbf{A}, (\mathbf{T}, \mathbf{D}), \mathbf{S} \right) \in \mathbb{Z}^{n \times m} \times (\mathbb{Z}^{n \times t} \times \mathbb{Z}^{t \times t}) \times \mathbb{Z}^{m \times t} \left| \begin{array}{l} \mathbf{A}\mathbf{S} \equiv \mathbf{T} \pmod{q} \\ \wedge \mathbf{D} = \mathbf{S}^\top \mathbf{S} \\ \wedge \forall i \in [t], D_{i,i} \leq \beta^2 \end{array} \right. \right\} . \quad (1)$$

Now, let $(\mathbf{A}, (\mathbf{T}, \mathbf{D}), \mathbf{S}) \in \mathbf{R}_{q,\beta,2t}$. The final protocol that we design is the following:

1. The prover computes $\tilde{\mathbf{S}} := \mathbf{G}^{-1}(\mathbf{S})$, $\tilde{\mathbf{T}} \equiv \mathbf{A}\tilde{\mathbf{S}} \pmod q$ and $\tilde{\mathbf{D}} := \tilde{\mathbf{S}}^\top \tilde{\mathbf{S}}$ and sends $\tilde{\mathbf{T}}, \tilde{\mathbf{D}}$ to the verifier.
2. The verifier samples a challenge $\mathbf{C} \leftarrow \{0, \pm 1\}^{2kt \times t}$ and send it to the prover.
3. The prover computes and outputs the updated witness $\mathbf{Z} := \tilde{\mathbf{S}}\mathbf{C}$.
4. The verifier computes $\mathbf{T}' := \tilde{\mathbf{T}}\mathbf{C}$ and $\mathbf{D}' := \mathbf{C}^\top \tilde{\mathbf{D}}\mathbf{C}$, accepts if

$$\begin{aligned} \mathbf{G}^\top \tilde{\mathbf{D}}\mathbf{G} &= \mathbf{D} \\ \wedge \tilde{\mathbf{T}}\mathbf{G} &\equiv \mathbf{T} \pmod q, \end{aligned}$$

and outputs the updated instance $(\mathbf{T}', \mathbf{D}')$.

The protocol is complete with no norm blowup. We are left to show that the additional information allows us to enforce exact extracted norm. We consider a new extractor that acts a following:

1. Run the malicious prover, answering its query with a uniformly random $\mathbf{C} \leftarrow \mathcal{C}^{2kt \times t}$, to obtain a transcript $(\tilde{\mathbf{T}}, \tilde{\mathbf{D}}, \mathbf{C}, \mathbf{Z})$.
2. If the transcript is not accepting, abort.
3. Rewind the prover to the beginning and run the extractor to obtain a witness $\tilde{\mathbf{S}} \in \mathbb{Z}^{m \times 2kt}$ (note that this is not the final witness that we previously extracted, which can be recovered by right multiplying by \mathbf{G}), aborting if extraction fails.
4. Output $\mathbf{S} := \tilde{\mathbf{S}}\mathbf{G}$.
First note that, as desired:

$$\mathbf{A}\mathbf{S} \equiv \mathbf{A}\tilde{\mathbf{S}}\mathbf{G} \equiv \tilde{\mathbf{T}}\mathbf{G} \equiv \mathbf{T} \pmod q .$$

If $\tilde{\mathbf{S}}^\top \tilde{\mathbf{S}} = \tilde{\mathbf{D}}$, then we have that

$$\mathbf{S}^\top \mathbf{S} = (\tilde{\mathbf{S}}\mathbf{G})^\top \tilde{\mathbf{S}}\mathbf{G} = \mathbf{G}^\top \tilde{\mathbf{D}}\mathbf{G} = \mathbf{D} ,$$

and since, for $i \in [2t]$, $D_{i,i} \leq \beta^2$ we are done. What is left is to bound the probability that $\tilde{\mathbf{S}}^\top \tilde{\mathbf{S}} \neq \tilde{\mathbf{D}}$. Since the first transcript is accepting, it must be that

$$\mathbf{A}\mathbf{Z} \equiv \mathbf{T}' \equiv \tilde{\mathbf{T}}\mathbf{C} \equiv \mathbf{A}\tilde{\mathbf{S}}\mathbf{C} \pmod q .$$

Thus, it must be that $\mathbf{Z} = \tilde{\mathbf{S}}\mathbf{C}$, or else the adversary has found a short SIS solution (since for every $j \in [2t]$, $\|\tilde{\mathbf{S}}_{*,j}\| \leq 2\beta$ and \mathbf{C}, \mathbf{Z} are short). When this holds, it must also be that $(\tilde{\mathbf{S}}\mathbf{C})^\top \tilde{\mathbf{S}}\mathbf{C} = \mathbf{C}^\top \tilde{\mathbf{D}}\mathbf{C}$. Writing $f(\mathbf{X}) = (\tilde{\mathbf{S}}\mathbf{X})^\top \tilde{\mathbf{S}}\mathbf{X}$ and $g(\mathbf{X}) = \mathbf{X}^\top \tilde{\mathbf{D}}\mathbf{X}$, the above conditions can be rewritten as $f(\mathbf{C}) = g(\mathbf{C})$. The functions f and g can be thought as $2kt \times t$ functions (one for each coordinate), and each of these functions is a multivariate polynomial of total degree at most 2. Indexing accordingly, further if $f(\mathbf{C}) = g(\mathbf{C})$ then $f_{i,i}(\mathbf{C}_{*,i}) = g_{i,i}(\mathbf{C}_{*,i})$ for $i \in [t]$. Since $\tilde{\mathbf{S}}^\top \tilde{\mathbf{S}} \neq \tilde{\mathbf{D}}$, these two polynomials are not identically equal, and so the probability that, over a random setting of the variables, the equation holds is at most $\frac{2}{|\mathcal{C}|}$ by the Demillo-Lipton-Schwartz-Zippel lemma (applied over the integral domain \mathbb{Z})³. Since the equation needs to hold jointly over all the choices

³ Choosing \mathcal{C} to be ternary instead of the arguably more natural binary challenges, in hindsight, is what allows us to have soundness in this step.

of i , then the probability is at most $\left(\frac{2}{|C|}\right)^t$. This concludes our argument. We highlight that this probabilistic test was the main reason why chose challenge matrices with ternary entries.

1.3 Related Works

Folding schemes were introduced by Kothapalli et al. [KST22] as a motivation to build incrementally verifiable computation from simple cryptographic building blocks. In a concurrent work, Bünz et al. [BCL+21] generically constructed an IVC from a similar primitive, called a split-accumulation scheme. In both works, the underlying folding/accumulation scheme works for a fixed, but universal, R1CS language. More recently, there has been significant progress in building folding schemes which circumvent the limitation of a single fixed R1CS, by supporting multiple circuits, high-degree relations, and lookup gates [BC23, EG23, KS22, KS23]. The aforementioned constructions still crucially rely on additively homomorphic vector commitments. Thus, we believe that our techniques could be applied to the aforementioned constructions identically as for [BCL+21, KST22].

To the best of our knowledge, the only lattice-based folding scheme is the work by Boneh and Chen [BC24], called `LatticeFold`. The construction also follows the decompose-and-fold paradigm, which circumvents the norm growth issue during an honest execution. On the contrary, the paper introduces a new way to prove shortness in the infinity norm by cleverly combining the CRT packing technique [BLS19, ELL19, YAZ+19], together with the sumcheck argument [LFKN92]. By the nature of the techniques, the folding scheme must rely on structured lattice assumptions. Moreover, proving the ℓ_2 norm, rather than the ℓ_∞ one, is very often what one would like to do when constructing proofs for lattice-based primitives – especially when the witness vector comes from performing trapdoor sampling [ABB10, DLP14, MP12].

2 Preliminaries

Notation. We denote the security parameter by λ , which is implicitly given to all algorithms unless specified otherwise. Further, we write $\text{negl}(\lambda)$ (resp. $\text{poly}(\lambda)$) to denote an unspecified negligible function (resp. polynomial) in λ . In this work, we implicitly assume that the vast majority of the key parameters, e.g. the ring dimension, and the dimensions of matrices and vectors, are $\text{poly}(\lambda)$. However, the modulus used in this work may be super-polynomial in λ .

For $a, b \in \mathbb{N}$ with $a < b$, write $[a, b] := \{a, a + 1, \dots, b\}$, $[a] := [1, a]$. For $q \in \mathbb{N}$ write \mathbb{Z}_q for the integers modulo q . We denote vectors with lowercase boldface (i.e. \mathbf{u}, \mathbf{v}) and matrices with uppercase boldface (i.e. \mathbf{A}, \mathbf{B}). Specifically, for a matrix \mathbf{A} , we write $\mathbf{A}_{i,*}$ and $\mathbf{A}_{*,j}$ for the i -th row and the j -th column of \mathbf{A} respectively, and write with lowercase $A_{i,j}$ for the entry in the i -th row and j -th column. For a vector \mathbf{x} of length n , we write x_i or $\mathbf{x}[i]$ for its i -th entry. Similarly,

we define $\mathbf{x}_i := (x_1, \dots, x_i)$ for $i \in [n]$. Given two vectors \mathbf{u}, \mathbf{v} , we denote by (\mathbf{u}, \mathbf{v}) its concatenation.

Decompose and Gadget Matrix. Let $b > 1$. We set $k := \lfloor \log_b \beta \rfloor + 2^4$ and $\mathbf{g} = [1, b, \dots, b^{k-1}]^\top \in \mathbb{Z}^k$. Given $\mathbf{S} \in \mathbb{Z}^{m \times n}$, we can decompose it by computing $\tilde{\mathbf{S}} \in \mathbb{Z}^{m \times kn}$ such that $\mathbf{S} = \tilde{\mathbf{S}}\mathbf{G}_n$, where \mathbf{G}_n is the gadget matrix and $\mathbf{G}_n := \mathbf{I}_n \otimes \mathbf{g} \in \mathbb{Z}^{kn \times n}$. Note that if $\|\mathbf{S}_{*,i}\| \leq \beta$ for all $i \in [n]$, then $\|\tilde{\mathbf{S}}_{*,j}\| \leq \lfloor \frac{b}{2} \rfloor \sqrt{m}$ for all $j \in [kn]$. We denote $\mathbf{G}_n^{-1} : \mathbb{Z}^{m \times n} \rightarrow \mathbb{Z}^{m \times kn}$ for the function that decomposes \mathbf{S} into $\tilde{\mathbf{S}}$ satisfying $\mathbf{S} = \tilde{\mathbf{S}}\mathbf{G}_n$. When the dimensions are clear from context we simply write \mathbf{G} and \mathbf{G}^{-1} .

Definition 1 (SIS). Let $q = q(\lambda)$, $n = n(\lambda)$, $m = m(\lambda)$ and $\beta = \beta(\lambda)$. We say that the $\text{SIS}_{n,m,q,\beta}$ assumption holds if for any PPT adversary \mathcal{A} , the following holds:

$$\Pr \left[\mathbf{Az} \equiv \mathbf{0} \pmod{q} \wedge 0 < \|\mathbf{z}\| \leq \beta \mid \begin{matrix} \mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m} \\ \mathbf{z} \leftarrow \mathcal{A}(\mathbf{A}) \end{matrix} \right] = \text{negl}(\lambda) .$$

2.1 The Demillo-Lipton-Schwartz-Zippel Lemma

We recall the Demillo-Lipton-Schwartz-Zippel lemma [DL78, Sch80, Zip79], a tool for probabilistic polynomial identity testing commonly used in proof systems.

Lemma 1 (Demillo-Lipton-Schwartz-Zippel

Lemma). *Let $f \in \mathcal{R}[x_1, x_2, \dots, x_n]$ be a non-zero polynomial of total degree d over an integral domain \mathcal{R} . Let S be a finite subset of \mathcal{R} and r_1, \dots, r_n be sampled independently and uniformly random from S . Then*

$$\Pr [f(r_1, \dots, r_n) = 0] \leq \frac{d}{|S|} .$$

2.2 Concentration Inequalities

We will use the following well-known Chernoff-Hoeffding bound.

Lemma 2 (Chernoff-Hoeffding Bound). *Let X_1, \dots, X_n be independent random variables taking value in $\{0, 1\}$. Let $X = \sum_{i=1}^n X_i$ denote their sum and let $\mu = \mathbb{E}[X]$. Then for all $\epsilon \geq 0$:*

$$\Pr [X \leq \mu - \epsilon n] \leq e^{-2\epsilon^2 n} .$$

⁴ We use balanced base- b decomposition throughout, where $x = \sum_{i \in [k]} x_i b^i$ and $|x_i| \leq \lfloor \frac{b}{2} \rfloor$.

2.3 Reduction of Knowledge

We recall the definition of reduction of knowledge from [KP23], which also captures the notion of folding scheme. That is, a prover, who wants to prove that it knows a witness w_1 such that $(x_1, w_1) \in R_1$, can use a reduction of knowledge from R_1 to R_2 and try to prove that it knows a witness w_2 such that $(x_2, w_2) \in R_2$, where x_2 is the reduced instance.

Definition 2 (Reduction of Knowledge). Consider ternary relations R_1 and R_2 . A reduction of knowledge from R_1 to R_2 consists of three PPT algorithms $(\mathcal{G}, \mathcal{P}, \mathcal{V})$ denoting the generator, the prover, and the verifier

- $\mathcal{G}(\lambda) \rightarrow i$: Takes security parameter λ . Outputs public parameters i .
- $\mathcal{P}(i, x_1, w_1) \rightarrow (x_2, w_2)$: Takes as input public parameters i , and statement-witness pair (x_1, w_1) . Interactively reduces the statement $(i, x_1, w_1) \in R_1$ to a new statement $(i, x_2, w_2) \in R_2$.
- $\mathcal{V}(i, x_1) \rightarrow x_2$: Takes as input public parameters i , and statement x_1 associated with R_1 . Interactively reduces the task of checking x_1 to the task of checking a new statement x_2 associated with R_2 .

Let $\langle \mathcal{P}, \mathcal{V} \rangle$ denote the interaction between \mathcal{P} and \mathcal{V} that runs the interaction on prover input (i, x_1, w_1) and verifier input (i, x_1) , then outputs the verifier’s statement x_2 and the prover’s witness w_2 . A reduction of knowledge $\Pi = (\mathcal{G}, \mathcal{P}, \mathcal{V})$ from R_1 to R_2 satisfies the following properties.

Definition 3 (Perfect Completeness). Π has perfect completeness if for all PPT adversaries \mathcal{A} ,

$$\Pr \left[(i, x_2, w_2) \in R_2 \mid \begin{array}{l} i \leftarrow \text{Setup}(1^\lambda) \\ (x_1, w_1) \leftarrow \mathcal{A}(i) \\ (x_2, w_2) \leftarrow \langle \mathcal{P}(i, x_1, w_1), \mathcal{V}(i, x_1) \rangle \end{array} \right] = 1 .$$

Definition 4 (Knowledge Soundness). Π is knowledge sound (with knowledge error $\kappa(\lambda)$) if for all expected polynomial-time adversaries \mathcal{A} and \mathcal{P}^* , there is an expected polynomial-time extractor \mathcal{E} such that

$$\Pr \left[\begin{array}{l} (i, x_2, w_2) \in R_2 \\ \wedge (i, x_1, w_1) \notin R_1 \end{array} \mid \begin{array}{l} i \leftarrow \text{Setup}(1^\lambda) \\ (x_1, st) \leftarrow \mathcal{A}(i) \\ (\text{Tr}, x_2, w_2) \leftarrow \langle \mathcal{P}^*(i, x_1, st), \mathcal{V}(i, x_1) \rangle \\ w_1 \leftarrow \mathcal{E}^{\mathcal{P}^*}(i, x_1, st) \end{array} \right] \leq \kappa(\lambda) \quad 5.$$

6

⁶ Our definition of knowledge soundness is different but equivalent to that of [KP23].

Definition 5 (Public Reducibility). Π satisfies public reducibility if there exists a deterministic polynomial-time algorithm f such that for any PPT adversary \mathcal{A} and expected polynomial-time adversary \mathcal{P}^* ,

$$\Pr \left[f(\mathfrak{i}, \mathfrak{x}_1, \text{Tr}) = \mathfrak{x}_2 \left| \begin{array}{l} \mathfrak{i} \leftarrow \text{Setup}(1^\lambda) \\ (\mathfrak{x}_1, \text{st}) \leftarrow \mathcal{A}(\mathfrak{i}) \\ (\text{Tr}, \mathfrak{x}_2, \mathfrak{w}_2) \leftarrow \langle \mathcal{P}^*(\mathfrak{i}, \mathfrak{x}_1, \text{st}), \mathcal{V}(\mathfrak{i}, \mathfrak{x}_1) \rangle \end{array} \right. \right] = 1 .$$

3 A Folding Scheme for Ajtai Commitment Openings

In this section, we construct a folding scheme for the Ajtai commitment openings relation $R_{q,\beta,t}$, defined in Eq. (1); or equivalently, a reduction of knowledge from $(R_{q,\beta,t})^2$ to $R_{q,\beta,t}$.

For simplicity, we describe the folding scheme as the composition of two reductions of knowledge from $(R_{q,\beta,t})^2$ to $R_{q,\beta,2t}$ and from $R_{q,\beta,2t}$ to $R_{q,\beta,t}$. The first reduction of knowledge serves the purpose of merging two instances of $R_{q,\beta,t}$ into one single instance $R_{q,\beta,2t}$ of larger size, while the second reduction of knowledge is where folding takes place to reduce the size of the instance from $R_{q,\beta,2t}$ to $R_{q,\beta,t}$.

3.1 Reduction of Knowledge from $(R_{q,\beta,t})^2$ to $R_{q,\beta,2t}$

Let $(\mathbf{A}, (\mathbf{T}_1, \mathbf{D}_1), \mathbf{S}_1)$, $(\mathbf{A}, (\mathbf{T}_2, \mathbf{D}_2), \mathbf{S}_2)$ be two instances of $R_{q,\beta,t}$. The idea is to concatenate $\mathbf{S} := [\mathbf{S}_1 \ \mathbf{S}_2]$ and $\mathbf{T} := [\mathbf{T}_1 \ \mathbf{T}_2]$. However, the verifier does not have enough information to compute $\mathbf{D} = \mathbf{S}^\top \mathbf{S}$. Hence, we let the prover send $\mathbf{S}_1^\top \mathbf{S}_2$ and $\mathbf{S}_2^\top \mathbf{S}_1$ to the verifier. We illustrate the protocol in Fig. 1.

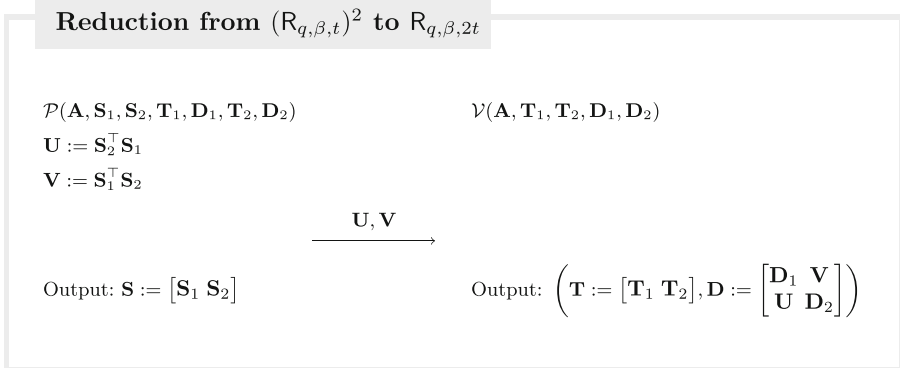


Fig. 1. Reduction of knowledge from $(R_{q,\beta,t})^2$ to $R_{q,\beta,2t}$.

Table 1. Overview of parameters and notation.

Parameter	Explanation
q	SIS modulus
n	Height of the matrix \mathbf{A}
m	Width of the matrix \mathbf{A}
β	Norm bound for SIS instances
t	Number of commitment openings
b	Decomposition base
k	$\lfloor \log_b \beta \rfloor + 2$

Lemma 3. *The protocol shown in Fig. 1 satisfies public reducibility, perfect completeness, and knowledge soundness.*

Proof. We prove each property separately.

Public reducibility: Given the instances $(\mathbf{T}_1, \mathbf{D}_1)$, $(\mathbf{T}_2, \mathbf{D}_2)$ and the transcript $\text{Tr} = (\mathbf{U}, \mathbf{V})$, one can efficiently compute \mathbf{T}, \mathbf{D} .

Completeness: We have that

$$\begin{aligned} \mathbf{AS} &\equiv \mathbf{A} [\mathbf{S}_1 \ \mathbf{S}_2] \equiv [\mathbf{AS}_1 \ \mathbf{AS}_2] \equiv [\mathbf{T}_1 \ \mathbf{T}_2] \equiv \mathbf{T} \pmod{q} , \\ \mathbf{S}^\top \mathbf{S} &= \begin{bmatrix} \mathbf{S}_1^\top \\ \mathbf{S}_2^\top \end{bmatrix} [\mathbf{S}_1 \ \mathbf{S}_2] = \begin{bmatrix} \mathbf{S}_1^\top \mathbf{S}_1 & \mathbf{S}_1^\top \mathbf{S}_2 \\ \mathbf{S}_2^\top \mathbf{S}_1 & \mathbf{S}_2^\top \mathbf{S}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{D}_1 & \mathbf{V} \\ \mathbf{U} & \mathbf{D}_2 \end{bmatrix} = \mathbf{D} . \end{aligned}$$

We can see from the last inequality that the diagonal of \mathbf{D} containing the diagonals of \mathbf{D}_1 and \mathbf{D}_2 , thus $D_{i,i} \leq \beta^2, \forall i \in [2t]$.

Knowledge soundness: Given $(\mathbf{A}, (\mathbf{T}, \mathbf{D}), \mathbf{S}) \in \mathbf{R}_{q,\beta,2t}$, it is not hard to see that if we parse $[\bar{\mathbf{S}}_1 \ \bar{\mathbf{S}}_2] := \mathbf{S}$, then

$$(\mathbf{A}, (\mathbf{T}_1, \mathbf{D}_1), \bar{\mathbf{S}}_1), (\mathbf{A}, (\mathbf{T}_2, \mathbf{D}_2), \bar{\mathbf{S}}_2) \in \mathbf{R}_{q,\beta,t} .$$

□

3.2 Reduction of Knowledge from $\mathbf{R}_{q,\beta,2t}$ to $\mathbf{R}_{q,\beta,t}$

Now, we describe the reduction of knowledge (see Fig. 2) to fold a larger instance to a smaller one while keeping the norm small (Table 1).

The prover starts by decomposing the witness $\tilde{\mathbf{S}} = \mathbf{G}^{-1}(\mathbf{S})$. In this section, the dimension $2t$ is fixed, and we write \mathbf{G} and \mathbf{G}^{-1} as shorthand for \mathbf{G}_{2t} and \mathbf{G}_{2t}^{-1} , respectively.

Next, it computes and sends $\tilde{\mathbf{T}} := \mathbf{A}\tilde{\mathbf{S}}$ and $\tilde{\mathbf{D}} := \tilde{\mathbf{S}}^\top \tilde{\mathbf{S}}$ to the verifier, where $\tilde{\mathbf{D}}$ serves as a proof of exact norm. The verifier then proceeds with uniform sampling and sending the challenge $\mathbf{C} \in \mathcal{C}^{2kt \times t}$.

Finally, the prover outputs the folded witness $\mathbf{S}' := \tilde{\mathbf{S}}\mathbf{C}$. Meanwhile, the verifier performs two checks. Firstly, it checks $\mathbf{G}^\top \tilde{\mathbf{D}}\mathbf{G}^\top = \mathbf{D}$ to verify the norm proof. Secondly, it checks $\tilde{\mathbf{T}}\mathbf{G} \equiv \mathbf{T} \pmod q$ to ensure that the prover decomposes correspondently. Then, it outputs $(\mathbf{T}' := \tilde{\mathbf{T}}\mathbf{C}, \mathbf{D}' := \mathbf{C}^\top \tilde{\mathbf{D}}\mathbf{C})$ as the folded instance. Note that the norm of the new witness does not increase as long as the challenge space only contains small elements. Furthermore, looking ahead to knowledge soundness, we set $\mathcal{C} := \{-1, 0, 1\}$.

Now, we prove that this reduction of knowledge satisfies public reducibility, perfect completeness, and knowledge soundness.

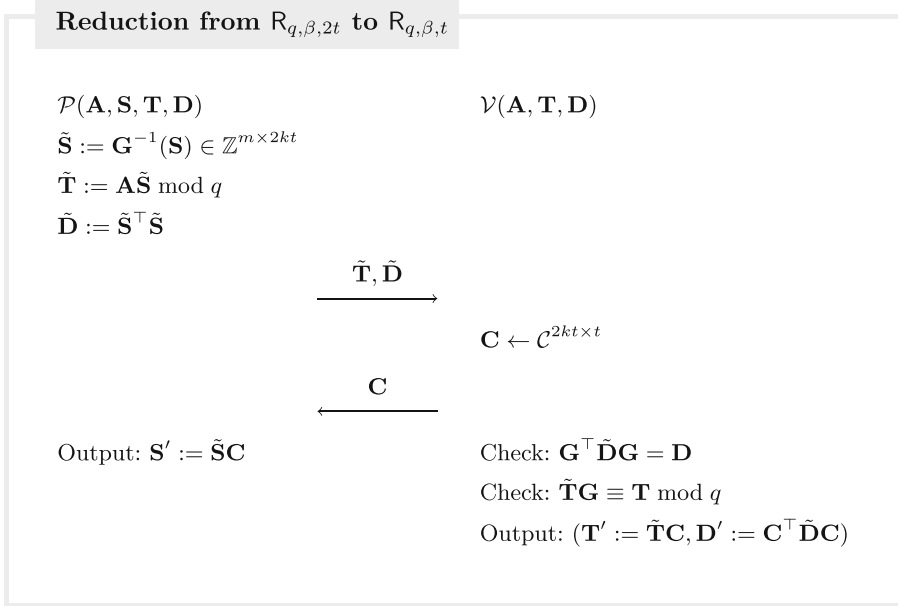


Fig. 2. Reduction of knowledge from $R_{q,\beta,2t}$ to $R_{q,\beta,t}$.

Lemma 4 (Public Reducibility and Perfect Completeness). *The protocol Π shown in Fig. 2 satisfies public reducibility. Furthermore, if $t \leq \beta / (2k \lfloor \frac{b}{2} \rfloor \sqrt{m})$, Π satisfies perfect completeness.*

Proof We prove each property in turn.

Public reducibility Given the instance $(\mathbf{A}, (\mathbf{T}, \mathbf{D}), \mathbf{S})$ and the transcript $\text{Tr} = (\tilde{\mathbf{T}}, \tilde{\mathbf{D}}, \mathbf{C})$, one can efficiently compute $(\mathbf{T}', \mathbf{D}')$.

Perfect completeness We have that

$$\begin{aligned}
\mathbf{G}^\top \tilde{\mathbf{D}} \mathbf{G} &= \mathbf{G}^\top \tilde{\mathbf{S}}^\top \tilde{\mathbf{S}} \mathbf{G} = \mathbf{S}^\top \mathbf{S} = \mathbf{D} \ , \\
\tilde{\mathbf{T}} \mathbf{G} &\equiv \mathbf{A} \tilde{\mathbf{S}} \mathbf{G} \equiv \mathbf{A} \mathbf{S} \equiv \mathbf{T} \pmod{q} \ , \\
\mathbf{A} \mathbf{S}' &\equiv \mathbf{A} \tilde{\mathbf{S}} \mathbf{C} \equiv \tilde{\mathbf{T}} \mathbf{C} \equiv \mathbf{T}' \pmod{q} \ , \\
\mathbf{S}'^\top \mathbf{S}' &= (\tilde{\mathbf{S}} \mathbf{C})^\top \tilde{\mathbf{S}} \mathbf{C} = \mathbf{C}^\top \tilde{\mathbf{S}}^\top \tilde{\mathbf{S}} \mathbf{C} = \mathbf{C}^\top \tilde{\mathbf{D}} \mathbf{C} = \mathbf{D}' \ , \\
\|\mathbf{S}'_{*,j}\| &\leq \left\| \sum_{i=1}^{2kt} \mathbf{C}_{i,j} \tilde{\mathbf{S}}_{*,i} \right\| \leq 2kt \left\lfloor \frac{b}{2} \right\rfloor \sqrt{m} \leq \beta \ ,
\end{aligned}$$

where the last inequality holds when $t \leq \beta / (2k \lfloor \frac{b}{2} \rfloor \sqrt{m})$. \square

To demonstrate that the protocol shown in Fig. 2 is knowledge sound (with exact witnesses), we first construct an extractor that yields a relaxed witness, as detailed in Lemma 5. Then, in Lemma 6, we augment this relaxed extractor in order to achieve (exact) knowledge soundness.

Lemma 5 (Relaxed Knowledge Soundness). *For a malicious prover \mathcal{P} , which convinces the verifier with probability $\epsilon > 4 \cdot 2^{t(\delta+2/3)} / 3^t$ for any $\delta > 0$, there exists an extractor for the protocol in Fig. 2 that yields $\bar{\mathbf{S}}$ satisfying*

$$\mathbf{A} \bar{\mathbf{S}} \equiv \tilde{\mathbf{T}} \pmod{q}, \quad \|\bar{\mathbf{S}}_{*,j}\| \leq 2\beta, \forall j \in [2kt] \ . \quad (2)$$

and runs in time $O(\lambda kt / \epsilon)$.

Proof We closely follow the approach from Baum et al. [BBC+18]. For $j \in [2kt]$, we construct an extractor that produces two accepting transcripts, with challenges $\mathbf{C}^{(0,j)}$, $\mathbf{C}^{(1,j)}$ and corresponding witnesses $\mathbf{Z}^{(0,j)}$, $\mathbf{Z}^{(1,j)}$ such that $\mathbf{C}^{(0,j)}$ and $\mathbf{C}^{(1,j)}$ are identical except for the j -th row, and further such that, there exists $i \in [t]$ such that $C_{j,i}^{(0,j)} - C_{j,i}^{(1,j)} = \pm 1$. This suffices to show the lemma since we have that

$$\mathbf{A}(\mathbf{Z}_{*,i}^{(0,j)} - \mathbf{Z}_{*,i}^{(1,j)}) = (C_{j,i}^{(0,j)} - C_{j,i}^{(1,j)}) \tilde{\mathbf{T}}_{*,j} \pmod{q}.$$

We therefore obtain $\bar{\mathbf{S}}_{*,j} := (\mathbf{Z}_{*,i}^{(0,j)} - \mathbf{Z}_{*,i}^{(1,j)}) / (C_{j,i}^{(0,j)} - C_{j,i}^{(1,j)})$ with norm at most 2β .

Let \mathcal{P} denote a (possibly malicious) prover, which we assume to be deterministic without loss of generality. Let ϵ denote the success probability of the prover \mathcal{P} (over the randomness of the choice of the challenge \mathbf{C}). For $j \in [2kt]$, the extractor \mathcal{E}_j is the following algorithm.

$\mathcal{E}_j^{\mathcal{P}}(\mathbf{x}) :$

1. Run \mathcal{P} until it outputs its first message $\tilde{\mathbf{T}}, \tilde{\mathbf{D}}$.
2. Sample $\mathbf{C}^{(0,j)} \leftarrow \mathcal{C}^{2kt \times t}$.
3. Run \mathcal{P} until it outputs a witness $\mathbf{Z}^{(0,j)}$.

4. If the verifier accepts the transcript $(\tilde{\mathbf{T}}, \tilde{\mathbf{D}}, \mathbf{C}^{(0,j)}, \mathbf{Z}^{(0,j)})$ continue, else go to Item 2
5. Define

$$S_j := \left\{ \mathbf{C} \in \mathcal{C}^{2kt \times t} \left| \begin{array}{l} \exists i \in [t] \text{ s.t. } |C_{j,i}^{(0,j)} - C_{j,i}| = 1 \\ \forall i \in [t], j' \in [2kt] \text{ s.t. } j \neq j' : C_{j,i}^{(0,j)} = C_{j',i} \end{array} \right. \right\}$$

6. Rewind the prover \mathcal{P} . If this label has been reached more than λ/ϵ times, abort.
7. Sample $\mathbf{C}^{(1,j)} \leftarrow S_j$.
8. Run \mathcal{P} until it outputs a witness $\mathbf{Z}^{(1,j)}$.
9. If the verifier accepts the transcript $(\tilde{\mathbf{T}}, \tilde{\mathbf{D}}, \mathbf{C}^{(1,j)}, \mathbf{Z}^{(1,j)})$ continue, else go to 6
10. Output $(\mathbf{C}^{(0,j)}, \mathbf{C}^{(1,j)}, \mathbf{Z}^{(0,j)}, \mathbf{Z}^{(1,j)})$.

The expected running time of the extractor is at most $1/\epsilon + \lambda/\epsilon = \text{poly}(\lambda)/\epsilon$.

We are left to bound the failure probability of the extractor. We denote by $G \subseteq \mathcal{C}^{2kt \times t}$ the set of accepting challenges, i.e., those for which \mathcal{P} outputs an accepting transcript. We also say a challenge \mathbf{C}' is j -special w.r.t. \mathbf{C} if they disagree only in the j -th row, that have the required difference in at least on entry. The goal of the extractor $\mathcal{E}_j^{\mathcal{P}}$ is to output a challenge $\mathbf{C}^{(1,j)}$ that is both accepting and j -special w.r.t. $\mathbf{C}^{(0,j)}$. Consider the binary matrix \mathbf{M}_j , whose entries correspond to challenges. We index the rows of \mathbf{M}_j by $\mathbf{C}_{j,*}$ and its columns by $(\mathbf{C}_{1,*}, \dots, \mathbf{C}_{j-1,*}, \mathbf{C}_{j+1,*}, \dots, \mathbf{C}_{2kt,*})$. An entry \mathbf{C} in \mathbf{M}_j is 1 if $\mathbf{C} \in G$, and 0 otherwise. Note that the fraction of ones in \mathbf{M}_j is at least ϵ .

Following the terminology in [BBC+18], a column in \mathbf{M}_j is heavy if its fractions of ones is at least $\epsilon/2$. By [BBC+18, Lemma 2], given $\mathbf{C}^{(0,j)}$ is accepted, the probability that the column containing $\mathbf{C}^{(0,j)}$ is heavy is at least $1/2$. In this case, the fraction of both accepting and j -special (w.r.t. $\mathbf{C}^{(0,j)}$) challenges associated with the column is at least $\epsilon/2 - g(Z)$, where $g(Z)$ is the fraction of challenges that are not j -special in that column, depending on the number of zeroes Z in the j -th row of $\mathbf{C}^{(0,j)}$. Concretely, a challenge \mathbf{C}' is not j -special w.r.t. to \mathbf{C} if and only if, in the j -th row, \mathbf{C}' has the same zero entries as \mathbf{C} and the remaining entries are ± 1 ; thus

$$g(Z) = \frac{2^{t-Z}}{3^t} .$$

Since $\mathbf{C}^{(0,j)}$ is sampled uniformly from \mathcal{C}^t , Z is concentrated around $t/3$. Using the Chernoff-Hoeffding bound, we obtain an upper-bound on the abort probability. Specifically, let A be the event that the extractor aborts and H be the event that the column containing $\mathbf{C}^{(0,j)}$ is heavy, we have

$$\begin{aligned}
\Pr[A] &= \Pr[A \mid \bar{H}] \cdot \Pr[\bar{H}] + \Pr[A \wedge H] \\
&\leq \Pr[\bar{H}] + \Pr[A \wedge H \mid Z \leq t/3 - \delta t] \cdot \Pr[Z \leq t/3 - \delta t] \\
&\quad + \Pr[A \wedge H \mid Z > t/3 - \delta t] \cdot \Pr[Z > t/3 - \delta t] \\
&\leq \Pr[\bar{H}] + \Pr[Z \leq t/3 - \delta t] + \Pr[A \wedge H \mid Z > t/3 - \delta t] \\
&\leq 1/2 + e^{-2\delta^2 t} + (1 - (\epsilon/2 - g(t/3 - \delta t)))^{\lambda/\epsilon} .
\end{aligned}$$

If $\epsilon > 4 \cdot 2^{t(\delta+2/3)}/3^t$, then $\epsilon/2 - g(t/3 - \delta t) > \epsilon/4$ and

$$\begin{aligned}
(1 - (\epsilon/2 - g(t/3 - \delta t)))^{\lambda/\epsilon} &< (1 - \epsilon/4)^{\lambda/\epsilon} < e^{-4\lambda} < 2^{-\lambda} , \\
\Pr[A] &< 1/2 + e^{-2\delta^2 t} + 2^{-\lambda} .
\end{aligned}$$

Running the extractor $O(\lambda)$ times yields an extractor that runs in expected time $\text{poly}(\lambda)/\epsilon$ and outputs a transcript of the required structure. \square

By using Lemma 5, we obtain a relaxed extractor, which can be used to prove knowledge soundness of the protocol in Fig. 2. We note that this alternative notion of knowledge soundness, where the extractor runs in expected $\text{poly}(\lambda)/\epsilon$ times, is equivalent to the notion adapted for a reduction of knowledge (see [Att23, Remark 2.4] for more discussion).

Lemma 6 (Exact Knowledge Soundness). *Assuming $\text{SIS}_{n,m,q,(2kt+1)\beta}$, the protocol in Fig. 2 satisfies knowledge soundness.*

Proof. Let $(\text{Tr} := (\tilde{\mathbf{T}}, \tilde{\mathbf{D}}, \mathbf{C}), \mathbb{x}_2 := (\mathbf{T}', \mathbf{D}'), \mathbb{w}_2 := \mathbf{S}')$ be the output of the interaction between a malicious prover \mathcal{P}^* and the verifier \mathcal{V} . If $(\mathbf{A}, \mathbb{x}_2, \mathbb{w}_2) \in \mathbb{R}_{q,\beta,t}$, then by Lemma 5, we obtain a relaxed extractor that outputs $\bar{\mathbf{S}}$ satisfying Eq. 2.

Furthermore, in such cases, we have that $\mathbf{S}' = \bar{\mathbf{S}}\mathbf{C}$ with probability at least $1 - \kappa_{\text{SIS}}$. Indeed, otherwise, $\mathbf{S}' - \bar{\mathbf{S}}\mathbf{C}$ is $\text{SIS}_{n,m,q,(2kt+1)\beta}$ solutions since

$$\mathbf{A}\mathbf{S}' \equiv \mathbf{T}' \equiv \tilde{\mathbf{T}}\mathbf{C} \equiv \mathbf{A}\bar{\mathbf{S}}\mathbf{C} \pmod{q}.$$

In addition, when $\mathbf{S}' = \bar{\mathbf{S}}\mathbf{C}$, we have that $\mathbf{C}^\top \bar{\mathbf{S}}^\top \bar{\mathbf{S}}\mathbf{C} = \mathbf{S}'^\top \mathbf{S}' = \mathbf{C}^\top \tilde{\mathbf{D}}\mathbf{C}$, or equivalently, $f(\mathbf{C}) = g(\mathbf{C})$, where $f(\mathbf{X}) := \mathbf{X}^\top \bar{\mathbf{S}}^\top \bar{\mathbf{S}}\mathbf{X}$ and $g(\mathbf{X}) := \mathbf{X}^\top \tilde{\mathbf{D}}\mathbf{X}$ are functions in the variables $\mathbf{X} \in \mathcal{C}^{2kt \times t}$. Looking at the index (i, i) of f and g ,

$$\begin{aligned}
f_{i,i}(\mathbf{X}) &= \sum_{u \in [2kt]} \sum_{v \in [2kt]} (\bar{\mathbf{S}}^\top \bar{\mathbf{S}})_{u,v} X_{u,i} X_{v,i} , \\
g_{i,i}(\mathbf{X}) &= \sum_{u \in [2kt]} \sum_{v \in [2kt]} \tilde{D}_{u,v} X_{u,i} X_{v,i} ,
\end{aligned}$$

which both have total degree 2. Then by the [Demillo-Lipton-Schwartz-Zippel Lemma](#) for the integral domain \mathbb{Z} , we have that the probability of $f_{i,i}(\mathbf{C}_{*,i}) = g_{i,i}(\mathbf{C}_{*,i})$ but $\bar{\mathbf{S}}^\top \bar{\mathbf{S}} \neq \tilde{\mathbf{D}}$ is at most $2/|\mathcal{C}|$ for uniformly random $\mathbf{C}_{*,i}$ in \mathcal{C}^{2kt} .

Note that when $\bar{\mathbf{S}}^\top \bar{\mathbf{S}} = \tilde{\mathbf{D}}$, then

$$\begin{aligned} \mathbf{G}^\top \bar{\mathbf{S}}^\top \bar{\mathbf{S}} \mathbf{G} &= \mathbf{G}^\top \tilde{\mathbf{D}} \mathbf{G} = \mathbf{D} \ , \\ \mathbf{A} \bar{\mathbf{S}} \mathbf{G} &\equiv \tilde{\mathbf{T}} \mathbf{G} \equiv \mathbf{T} \pmod{q} \ , \end{aligned}$$

which implies $\bar{\mathbf{S}} \mathbf{G}$ is a valid witness for \mathbf{T} . Therefore, we can bound the probability of $(\mathbf{A}, \mathbb{x}_2, \mathbb{w}_2) \in R_{q,\beta,t}$ but $\bar{\mathbf{S}} \mathbf{G}$ is not a witness for $(\mathbf{A}, (\mathbf{T}, \mathbf{D}))$ by the probability that $f(\mathbf{C}) = g(\mathbf{C})$ but $\bar{\mathbf{S}}^\top \bar{\mathbf{S}} \neq \tilde{\mathbf{D}}$, which is at most $(2/|\mathcal{C}|)^t$ because for each $i \in [t]$, $f_{i,i}(\mathbf{C}_{*,i}) = g_{i,i}(\mathbf{C}_{*,i})$ and $\mathbf{C}_{*,i}$ is sampled independently and uniformly at random from \mathcal{C}^{2kt} . More precisely, if E is the event that the extractor in [Lemma 5](#) succeeds, then

$$\begin{aligned} &\Pr \left[(\mathbf{A}, (\mathbf{T}', \mathbf{D}'), \mathbf{S}') \in R_{q,\beta,t} \wedge (\mathbf{A}, (\mathbf{T}, \mathbf{D}), \bar{\mathbf{S}} \mathbf{G}) \notin R_{q,\beta,2t} \wedge E \right] \\ &\leq \Pr \left[f(\mathbf{C}) = g(\mathbf{C}) \wedge (\mathbf{A}, (\mathbf{T}, \mathbf{D}), \bar{\mathbf{S}} \mathbf{G}) \notin R_{q,\beta,2t} \wedge E \right] + \kappa_{\text{SIS}} \\ &\leq \Pr \left[f(\mathbf{C}) = g(\mathbf{C}) \wedge \bar{\mathbf{S}}^\top \bar{\mathbf{S}} \neq \tilde{\mathbf{D}} \wedge E \right] + \kappa_{\text{SIS}} \\ &\leq \left(\frac{2}{|\mathcal{C}|} \right)^t + \kappa_{\text{SIS}} \ . \end{aligned}$$

□

Applications to Folding Schemes for NP Relations. Unfortunately, unlike in Nova [\[KST22\]](#), we cannot easily modify our construction to support (relaxed) R1CS relations. The issue is that the norm of the (additional cross-term) folded witness now depends on the magnitude of entries in the R1CS matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$, which we cannot assume is small in general. We leave a construction of a lattice-based folding schemes for R1CS-type relations as future work.

Instead, we provide a sketch on how to build a folding scheme for the subset sum problem which is NP-complete. We recall that the subset sum problem is essentially to find a binary vector \mathbf{s} such that $\mathbf{M}\mathbf{s} = \mathbf{y}$ over \mathbb{Z} for public matrix \mathbf{M} and vector \mathbf{y} .

First, we use the observation from [\[LNP22\]](#) that an integer vector \mathbf{s} has binary values if and only if $\langle \mathbf{s}, \mathbf{s} - \mathbf{1} \rangle = 0$, where $\mathbf{1}$ is the all-one vector. Secondly, we can pick a proof system modulus q large enough so that $\mathbf{M}\mathbf{s} = \mathbf{y} \pmod{q}$ is equivalent to $\mathbf{M}\mathbf{s} = \mathbf{y}$ over integers, i.e. no modulo overflow occurs.

Thus, similarly to [\(1\)](#) we can define a relation:

$$R_{q,\beta,t}^* := \left\{ \begin{array}{l} (\mathbf{A}, (\mathbf{T}, \mathbf{D}), \mathbf{S}) \\ \in \mathbb{Z}^{n \times m} \times (\mathbb{Z}^{n \times t} \times \mathbb{Z}^{t \times t}) \times \mathbb{Z}^{m \times t} \end{array} \middle| \begin{array}{l} \mathbf{A} \mathbf{S} \equiv \mathbf{T} \pmod{q} \\ \wedge \mathbf{D} = \mathbf{S}^\top \mathbf{S} - \mathbf{S}^\top \mathbf{1} \end{array} \right\} \quad (3)$$

where in this equation $\mathbf{1}$ is the all-one matrix. Here, the matrix \mathbf{A} will contain the SIS commitment key (to ensure binding), together with the matrix \mathbf{M} related to the subset sum problem. Then, given a valid tuple $(\mathbf{A}, (\mathbf{T}, \mathbf{D}), \mathbf{S}) \in \mathbb{R}_{q,\beta,t}^*$, one can be convinced that the matrix \mathbf{S} has binary entries by simply checking that diagonal entries $D_{i,i}$ of \mathbf{D} are equal to zero. Finally, building a folding scheme for $\mathbb{R}_{q,\beta,t}^*$ is almost identical to the construction above up to certain straightforward modifications.

4 Implementation and Evaluation

4.1 Parameter Selection

For an input witness length m and a security parameter t , we need to select a SIS modulus $q \in \mathbb{N}$, a commitment output length $n \in \mathbb{N}$, a norm $\beta < q$, a decomposition basis b (which fixes a decomposition length $k = \lfloor \log_b \beta \rfloor + 2$) such that the following conditions are fulfilled:

1. The knowledge error $\kappa_{\text{KS}} = Q(\kappa_{\text{PIT}} + \kappa_{\text{rS}} + \kappa_{\text{SIS}})$ must be at most $2^{-\lambda}$, where κ_{rS} is the knowledge error from Lemma 5;
2. For perfect completeness, $2tk \left(\lfloor \frac{b}{2} \rfloor \sqrt{m}\right) \leq \beta$;
3. For (knowledge) soundness, $\text{SIS}_{n,m,\beta,L_2}$ must be κ_{SIS} -hard.

The knowledge error is $\kappa_{\text{KS}} = Q(\kappa_{\text{PIT}} + \kappa_{\text{rS}} + \kappa_{\text{SIS}}) = Q\left(\left(\frac{2}{3}\right)^t + 4\left(\frac{2^{t(\delta+2/3)}}{3^t}\right) + \kappa_{\text{SIS}}\right)$. Setting $\lambda = 128$ and $Q = 2^{64}$, we choose $t = 330$ and $\kappa_{\text{SIS}} \leq 2^{-(129+64)}$ such that $\kappa_{\text{KS}} \leq 5 \cdot 2^{64} \left(\frac{2}{3}\right)^{334} + 2^{-129} \leq 2^{-\lambda}$. The second condition gives rise to the bounds $\beta \geq e^{-W_{-1}\left(-\frac{\ln b}{bt\sqrt{m}}\right)} + 2 \lfloor \frac{b}{2} \rfloor t\sqrt{m}$ (where W_{-1} is the non-principal branch of the real Lambert W-function). Additionally, $2 \leq b \leq \sqrt{\beta}$. For efficiency, we want b to be as large as possible, i.e., $b \approx \sqrt{\beta}$. Substituting in the condition above, we get $t \left\lfloor \log_{\sqrt{\beta}}(\beta) + 2 \right\rfloor \sqrt{\beta}\sqrt{m} \stackrel{!}{\leq} \beta$, which yields $\beta = (4t)^2 m$, $b = \lfloor \sqrt{\beta} \rfloor$, and $k = 4$.

We choose $q = 2^{64}$ for the lattice modulus, which is both large enough to guarantee SIS hardness and allows for very efficient modular arithmetic (modular reductions reduce to wrapping 64-bit arithmetic, and are implemented directly in hardware for machines with 64-bit instruction sets).

Finally, we perform binary search on n in order to find the smallest n such that the underlying SIS instances are κ_{SIS} -hard. We rely on the `lattice-estimator` tool [est], which uses the methodology outlined by Gama and Nguyen [GN08].

Improving Proof Size. For the parameter sets outlined above, we made use of a worst-case bound on the norm of folded witnesses to ensure perfect completeness. If one is willing to accept a negligibly small completeness error κ_c , we can leverage probabilistic upper bounds on the norm of folded witnesses to reduce proof sizes.

Since $|C_{i,j}|$ is a Bernoulli-distributed random variable with $p = \frac{2}{3}$, we have $\mathbb{E} \left[\sum_{l \in [2kt]} |C_{l,j}| \right] = \frac{4kt}{3}$ and $\Pr \left[\sum_{l \in [2kt]} |C_{l,j}| \geq \frac{4kt}{3} + 2kte \right] \leq e^{-4kte^2}$ by a Chernoff-Hoeffding bound. Solving for $e^{4kte^2} = \kappa_c = 2^{-\mu}$ yields $\epsilon = \frac{\sqrt{\mu \ln 2}}{2kt}$. Putting everything together, we have that

$$\|\mathbf{S}_{*,j}\| \leq \left\lfloor \frac{b}{2} \right\rfloor \left(\frac{4kt}{3} + 2kte \right) \sqrt{m} = \left\lfloor \frac{b}{2} \right\rfloor \left(\frac{4kt}{3} + \sqrt{\mu \ln 2} \right) \sqrt{m}$$

with all but negligible probability. Setting $\mu = 128$, and for $t = 330$ and $k = 4$ as above, this bound is roughly a third of the worst-case bound.

4.2 Implementation

We implement Lova and plan to open-source our implementation. In our implementation, we translate several nice properties of Lova into hardware-friendly optimizations:

- We leverage symmetries to compute and send fewer matrix entries; in particular, our prover only computes one matrix instead of two for the protocol in Fig. 1, and only computes the lower triangular part of symmetric matrices for the protocol in Fig. 2.
- Since our challenges are ternary, random linear combinations can be computed without any multiplications, using only negations, and additions.
- We parallelize both the prover and verifier.
- As mentioned above, we set the SIS modulus to $q = 2^{64}$, which allows us to eschew modular arithmetic in favor of native 64-bit arithmetic.

In order to safely instantiate the Fiat-Shamir transform, we rely on and extend the `nimue` framework [nimue]. We benchmark our implementation on an AWS EC2 m5.8xlarge instance with 128 GB of RAM and 32 Intel Xeon vCPUs @ 3.1 GHz.

4.3 Evaluation

Proof Size. For one run of the Lova folding protocol with two witnesses of size m , the prover sends one $t \times t$ matrix with entries of norm at most β^2 (noting that $\mathbf{U} = \mathbf{V}^\top$), one $n \times 2kt$ matrix with entries in \mathbb{Z}_q , and one $2kt \times 2kt$ symmetric matrix with entries of norm at most $\left\lfloor \frac{b}{2} \right\rfloor^2$, totalling $t^2 [2 + \log \beta^2] + 2hkt [1 + \log q] + \frac{(2kt)(2kt+1)}{2} \left[2 + \log \left\lfloor \frac{b}{2} \right\rfloor^2 \right]$ bits.

In general (what we call PCD-type settings), the prover folds two full witnesses, i.e., matrices with t columns. In IVC-type settings, the prover repeatedly folds a fresh witness (i.e., which consists of the same vector concatenated t times with itself) with a non-fresh witness. In this setting, the prover and the verifier can exploit this extra structure to reduce computation and proof size. We show concrete proof times for varying witness lengths in Table 2.

Table 2. Proof sizes and prover runtime for a single folding step. We consider IVC and PCD-type settings, and perfect completeness (worst-case bound analysis) and negligible completeness error (probabilistic bound analysis).

	Instance length	2^{17}	2^{18}	2^{19}
IVC	Proof size ($\kappa_C = 0$)	17.53 MB	18.36 MB	19.18 MB
	Proof size ($\kappa_C \leq 2^{-128}$)	16.62 MB	17.42MB	18.24MB
	Prover time ($\kappa_C = 0$)	321 s	694 s	1501 s
	Prover time ($\kappa_C \leq 2^{-128}$)	296 s	630 s	1367 s
PCD	Proof size ($\kappa_C = 0$)	43.64 MB	45.51 MB	47.36 MB
	Proof size ($\kappa_C \leq 2^{-128}$)	41.62 MB	43.43MB	45.28MB
	Prover time ($\kappa_C = 0$)	728 s	1567 s	3440 s
	Prover time ($\kappa_C \leq 2^{-128}$)	690 s	1494 s	3236 s

Prover Runtime and Verifier Complexity. Concrete prover runtimes are shown in Table 2. The verifier needs to sample $\lceil 3^{2kt^2} \rceil \approx 3.22 \cdot kt^2$ bits from a hash function in order to generate the ternary challenge matrix. Checking $\mathbf{T}\mathbf{G} \equiv \mathbf{T} \bmod q$ and $\mathbf{G}^\top \mathbf{D}\mathbf{G} = \mathbf{D}$ requires $n \cdot 2t$ and $(2t)^2$ linear constraints, respectively. Finally, in order to check that the new instance is valid, the verifier needs to check $\mathbf{T}' = \mathbf{T}\mathbf{C}$ and $\mathbf{D}' = \mathbf{C}^\top \mathbf{D}\mathbf{C}$, which requires $n \cdot 2t$ and $(2k)^2 + (2kt)^2$ quadratic constraints, respectively. Note that these constraints are very sparse, and the for the latter constraints, the values of some variables are ternary; depending on the chosen constraint and proof systems, these properties may be exploited to significantly reduce the overhead of proving and verifying this circuit.

References

- [ABB10] S. Agrawal, D. Boneh, and X. Boyen. “Efficient Lattice (H)IBE in the Standard Model”. In: *EUROCRYPT*. 2010, pp. 553-572.
- [ACK21] T. Attema, R. Cramer, and L. Kohl. “A Compressed Σ -Protocol Theory for Lattices”. In: *CRYPTO (2)*. Vol. 12826. Lecture Notes in Computer Science. Springer, 2021, pp. 549-579.
- [Ajt96] M. Ajtai. “Generating Hard Instances of Lattice Problems (Extended Abstract)”. In: *STOC*. 1996, pp. 99-108.
- [AL21] M. R. Albrecht and R. W. F. Lai. “Subtractive Sets over Cyclotomic Rings - Limits of Schnorr-Like Arguments over Lattices”. In: *CRYPTO (2)*. Vol. 12826. Lecture Notes in Computer Science. Springer, 2021, pp. 519-548.
- [Att23] T. Attema. “Compressed Sigma-Protocol Theory”. PhD thesis. CWI and TNO, 2023. URL: <https://hdl.handle.net/1887/3619596>
- [BBBF18] D. Boneh, J. Bonneau, B. Bünz, and B. Fisch. “Verifiable Delay Functions”. In: *CRYPTO (1)*. Vol. 10991. Lecture Notes in Computer Science. Springer, 2018, pp. 757-788.
- [BBC+18] C. Baum, J. Bootle, A. Cerulli, R. d. Pino, J. Groth, and V. Lyubashevsky. “Sub-linear Lattice-Based Zero-Knowledge Arguments for Arithmetic Circuits”. In: Proceedings of the 38th Annual International Cryptology Conference. *CRYPTO '18*. 2018, pp. 669-699.

- [BC23] B. Bünz and B. Chen. *ProtoStar: Generic Efficient Accumulation/ Folding for Special Sound Protocols*. Cryptology ePrint Archive, Paper 2023/620. <https://eprint.iacr.org/2023/620> 2023. URL: <https://eprint.iacr.org/2023/620>
- [BC24] D. Boneh and B. Chen. *LatticeFold: A Lattice-based Folding Scheme and its Applications to Succinct Proof Systems*. Cryptology ePrint Archive, Paper 2024/257. <https://eprint.iacr.org/2024/257> Last accessed: 19.05.2024. 2024. URL: <https://eprint.iacr.org/2024/257>
- [BCCT13] N. Bitansky, R. Canetti, A. Chiesa, and E. Tromer. “Recursive Composition and Bootstrapping for SNARKs and Proof-Carrying Data”. In: *Proceedings of the 45th ACM Symposium on the Theory of Computing*. STOC ’13. 2013, pp. 111-120.
- [BCL+21] B. Bünz, A. Chiesa, W. Lin, P. Mishra, and N. Spooner. “Proof-Carrying Data Without Succinct Arguments”. In: *CRYPTO (1)*. Vol. 12825. Lecture Notes in Computer Science. Springer, 2021, pp. 681-710.
- [BCMS20] B. Bünz, A. Chiesa, P. Mishra, and N. Spooner. “Recursive Proof Composition from Accumulation Schemes”. In: *TCC (2)*. Vol. 12551. Lecture Notes in Computer Science. Springer, 2020, pp. 1-18.
- [BCTV14] E. Ben-Sasson, A. Chiesa, E. Tromer, and M. Virza. “Scalable Zero Knowledge via Cycles of Elliptic Curves”. In: *Proceedings of the 34th Annual International Cryptology Conference*. CRYPTO ’14. Extended version at <http://eprint.iacr.org/2014/595.2014,pp.276-294>
- [BGH19] S. Bowe, J. Grigg, and D. Hopwood. *Halo2*. 2019. URL: <https://github.com/zcash/halo2>
- [BLNS20] J. Bootle, V. Lyubashevsky, N. K. Nguyen, and G. Seiler. “A Non-PCP Approach to Succinct Quantum-Safe Zero-Knowledge”. In: *CRYPTO (2)*. Vol. 12171. Lecture Notes in Computer Science. Springer, 2020, pp. 441-469.
- [BLS19] J. Bootle, V. Lyubashevsky, and G. Seiler. “Algebraic Techniques for Short(er) Exact Lattice-Based Zero-Knowledge Proofs”. In: *CRYPTO (1)*. Vol. 11692. Lecture Notes in Computer Science. Springer, 2019, pp. 176-202.
- [BMRS20] J. Bonneau, I. Meckler, V. Rao, and E. Shapiro. *Coda: Decentralized Cryptocurrency at Scale*. Cryptology ePrint Archive, Paper 2020/352. <https://eprint.iacr.org/2020/352.2020> URL: <https://eprint.iacr.org/2020/352>
- [BS23] W. Beullens and G. Seiler. “LaBRADOR: Compact Proofs for RICS from Module-SIS”. In: *Lecture Notes in Computer Science 14085 (2023)*, pp. 518-548.
- [CMNW24] V. Cini, G. Malavolta, N. K. Nguyen, and H. Wee. *Polynomial Commitments from Lattices: Post-Quantum Security, Fast Verification and Transparent Setup*. Cryptology ePrint Archive, Paper 2024/281. <https://eprint.iacr.org/2024/281.2024> URL: <https://eprint.iacr.org/2024/281>
- [CT10] A. Chiesa and E. Tromer. “Proof-Carrying Data and Hearsay Arguments from Signature Cards”. In: *Proceedings of the 1st Symposium on Innovations in Computer Science*. ICS ’10. 2010, pp. 310-331.
- [Dam10] I. Damgård. On σ Protocols. <http://www.cs.au.dk/ivan/Sigma.pdf.2010>
- [DL78] R. A. DeMillo and R. J. Lipton. “A Probabilistic Remark on Algebraic Program Testing”. In: *Information Processing Letters 7.4 (1978)*, pp. 193-195.
- [DLP14] L. Ducas, V. Lyubashevsky, and T. Prest. “Efficient Identity-Based Encryption over NTRU Lattices”. In: *ASIACRYPT*. 2014, pp. 22- 41.

- [EG23] L. Eagen and A. Gabizon. *ProtoGalaxy: Efficient ProtoStar-style folding of multiple instances*. Cryptology ePrint Archive, Paper 2023/1106. <https://eprint.iacr.org/2023/1106> URL: <https://eprint.iacr.org/2023/1106>
- [ESLL19] M. F. Esgin, R. Steinfeld, J. K. Liu, and D. Liu. “Lattice-Based Zero-Knowledge Proofs: New Techniques for Shorter and Faster Constructions and Applications”. In: *CRYPTO (1)*. Springer, 2019, pp. 115-146.
- [est] M. R. Albrecht, B. Curtis, C. Yun, C. Lefebvre, F. Virdia, F. Göpfert, H. Hunt, H. Kippen, J. Owen, L. Ducas, L. Pulles, M. Schmidt, M. Walter, R. Player, and S. Scott. *lattice-estimator*. URL: <https://github.com/malb/lattice-estimator>.
- [FMN23] G. Fenzi, H. Moghaddas, and N. K. Nguyen. *Lattice-Based Polynomial Commitments: Towards Asymptotic and Concrete Efficiency*. Cryptology ePrint Archive, Paper 2023/846. <https://eprint.iacr.org/2023/846> URL: <https://eprint.iacr.org/2023/846>
- [GN08] N. Gama and P. Q. Nguyen. “Predicting Lattice Reduction”. In: *Advances in Cryptology - EUROCRYPT 2008*. Springer Berlin Heidelberg, 2008, pp. 31-51. URL: http://dx.doi.org/10.1007/978-3-540-78967-3_3
- [KMT22] D. Khovratovich, M. Maller, and P. R. Tiwari. *MinRoot: Candidate Sequential Function for Ethereum VDF*. Cryptology ePrint Archive, Paper 2022/1626. <https://eprint.iacr.org/2022/1626> URL: <https://eprint.iacr.org/2022/1626>
- [KP23] A. Kothapalli and B. Parno. “Algebraic Reductions of Knowledge”. In: *Advances in Cryptology - CRYPTO 2023*. Ed. by H. Handschuh and A. Lysyanskaya. Cham: Springer Nature Switzerland, 2023, pp. 669-701. isbn: 978-3-031-38551-3.
- [KS22] A. Kothapalli and S. Setty. *SuperNova: Proving universal machine executions without universal circuits*. Cryptology ePrint Archive, Paper 2022/1758. <https://eprint.iacr.org/2022/1758> URL: <https://eprint.iacr.org/2022/1758>
- [KS23] A. Kothapalli and S. Setty. HyperNova: Recursive arguments for customizable constraint systems. Cryptology ePrint Archive, Paper 2023/573. <https://eprint.iacr.org/2023/573> URL: <https://eprint.iacr.org/2023/573>
- [KST22] A. Kothapalli, S. T. V. Setty, and I. Tzialla. “Nova: Recursive Zero-Knowledge Arguments from Folding Schemes”. In: *CRYPTO (4)*. Vol. 13510. Lecture Notes in Computer Science. Springer, 2022, pp. 359-388.
- [LFKN92] C. Lund, L. Fortnow, H. J. Karloff, and N. Nisan. “Algebraic Methods for Interactive Proof Systems”. In: *Journal of the ACM* 39.4 (1992), pp. 859-868.
- [LNP22] V. Lyubashevsky, N. K. Nguyen, and M. Plançon. “Lattice-Based Zero-Knowledge Proofs and Applications: Shorter, Simpler, and More General”. In: *CRYPTO (2)*. Vol. 13508. Lecture Notes in Computer Science. Springer, 2022, pp. 71-101.
- [LS15] A. Langlois and D. Stehlé. “Worst-case to average-case reductions for module lattices”. In: *Des. Codes Cryptogr.* 75.3 (2015), pp. 565- 599.
- [Mina] O(1) Labs. *Mina cryptocurrency. 2017*. URL:<https://minaprotocol.com/>
- [MP12] D. Micciancio and C. Peikert. “Trapdoors for Lattices: Simpler, Tighter, Faster, Smaller”. In: *EUROCRYPT. 2012*, pp. 700-718.
- [nimue] M. Orr’u. *nimue*. URL: <https://github.com/arkworks-rs/nimue>

- [NIST] NIST. *Status Report on the Third Round of the NIST Post-Quantum Cryptography Standardization Process*. 2022. URL: <https://csrc.nist.gov/pubs/ir/8413/final>
- [PSTY13] C. Papamanthou, E. Shi, R. Tamassia, and K. Yi. “Streaming Authenticated Data Structures”. In: *EUROCRYPT*. Vol. 7881. Lecture Notes in Computer Science. Springer, 2013, pp. 353-370.
- [Sch80] J. T. Schwartz. “Fast Probabilistic Algorithms for Verification of Polynomial Identities”. In: *Journal of the ACM* 27.4 (1980), pp. 701-717.
- [Sho94] P. W. Shor. “Algorithms for Quantum Computation: Discrete Logarithms and Factoring”. In: *FOCS*. IEEE Computer Society, 1994, pp. 124-134.
- [Val08] P. Valiant. “Incrementally Verifiable Computation or Proofs of Knowledge Imply Time/Space Efficiency”. In: *Proceedings of the 5th Theory of Cryptography Conference*. TCC '08. 2008, pp. 1-18.
- [YAZ+19] R. Yang, M. H. Au, Z. Zhang, Q. Xu, Z. Yu, and W. Whyte. “Efficient Lattice-Based Zero-Knowledge Arguments with Standard Soundness: Construction and Applications”. In: *CRYPTO (1)*. Springer, 2019, pp. 147-175.
- [Zip79] R. Zippel. “Probabilistic algorithms for sparse polynomials”. In: *Proceedings of the 1979 International Symposium on Symbolic and Algebraic Computation*. EUROSAM '79. 1979, pp. 216-226.

Lattice Assumptions



On the Spinor Genus and the Distinguishing Lattice Isomorphism Problem

Cong Ling¹, Jingbo Liu^{2(✉)}, and Andrew Mendelsohn¹

¹ Department of EEE, Imperial College London, London, UK
{c.ling, andrew.mendelsohn18}@imperial.ac.uk

² Department of Computational, Engineering, and Mathematical Sciences,
Texas A&M University-San Antonio, San Antonio, USA
jliu@tamusa.edu

Abstract. This paper addresses the spinor genus, a previously unrecognized classification of quadratic forms in the context of cryptography, related to the lattice isomorphism problem (LIP). The spinor genus lies between the genus and equivalence class, thus refining the concept of genus. We present algorithms to determine whether two quadratic forms belong to the same spinor genus. If they do not, it provides a negative answer to the distinguishing variant of LIP. However, these algorithms have very high complexity, and we show that the proportion of genera splitting into multiple spinor genera is vanishing (assuming rank $n \geq 3$). For the special case of anisotropic integral binary forms ($n = 2$) over number fields with class number 1, we offer an efficient quantum algorithm to test if two forms lie in the same spinor genus. Our algorithm does not apply to the HAWK protocol, which uses integral binary Hermitian forms over number fields with class number greater than 1.

Keywords: quadratic forms · lattice isomorphism problem · spinor genus · class group

1 Introduction

Lattices have been studied for almost 30 years by the cryptographic community, since works by Ajtai [1, 2] gave worst-case to average-case reductions for lattice problems and an encryption scheme whose hardness was based on such worst-case lattice problems, as well as the introduction of NTRU [21]. Since then other foundational problems for lattice-based cryptography have been introduced, notably Learning with Errors [28]. A recent addition to these is the Lattice Isomorphism problem.

Informally, the Lattice Isomorphism problem (LIP) is, given two lattices, to decide if they are isomorphic or not. This can be rephrased in the language of quadratic forms: the LIP is, given two quadratic forms, to decide whether they lie in the same equivalence class or not, and if so to find such an isomorphism. This

problem was studied by Haviv and Regev [20], who gave an $n^{O(n)}$ algorithm to solve the problem. This problem was given cryptographic applications by van Woerden and Ducas [15], who gave worst-case to average-case reductions for certain forms of the problem, and constructed a KEM and signature scheme relying on the hardness of LIP. This was followed by the signature scheme HAWK [14], which relied on the security of LIP restricted to module lattices. The growing application and use of LIP-based schemes thus makes cryptanalysis of interest to the cryptological community. We also note [5], which studied a closely related problem to the LIP, named the lattice distortion problem.

The first step in a cryptanalytic direction was made in [7], which analysed the distribution of quadratic forms corresponding to random q -ary lattices in genera. Each quadratic form has an associated equivalence class, and each equivalence class lies in a genus. The disjoint union of equivalence classes ‘fills out’ the genera (i.e. each genus is a disjoint union of equivalence classes, and each class lies in one genus). Thus, *if* two forms were to lie in distinct genera, and this could be efficiently verified, a method for providing a negative answer to the LIP would be provided. The conclusion of that study was, informally, that ‘most’ random q -ary lattices lie inside one of few ‘large’ genera, and thus two forms can be sampled at random from a ‘large’ genus with the property that rejection sampling only negligibly biases the final distribution of forms.

Further work was done investigating the viability of using lattice hulls to solve LIP instances in [13], and the possibility of using characteristic vectors and lattice automorphisms to solve LIP was studied in [23].

In this work we continue the above line of investigation, studying notions of equivalence for positive definite integral quadratic forms. The contribution of this paper begins with a largely expository account of the *spinor genus*, a collection of equivalence classes with respect to an equivalence relation defined by the kernel of a certain homomorphism known as the *spinor norm*. A spinor genus is a disjoint union of equivalence classes, much like the genus, but a genus may contain multiple spinor genera. Thus, given two quadratic forms, one might compute their spinor genera, and if they lie in different spinor genera, the forms are not equivalent, providing a negative answer to distinguish LIP for those two forms. We observe that the spinor genus was omitted from the ‘arithmetic fingerprint’ of [15], and we here fill this lacuna.

1.1 Overview of Methodology

At a high level, spinor genera provide a finer classification of quadratic forms over \mathbb{Z} than genera. It is well known that for quadratic forms over \mathbb{Q} , the equivalence of two forms can be determined by checking their equivalence over the p -adic fields \mathbb{Q}_p for all primes p (including $p = \infty$). According to the famous local-global principle, two forms are equivalent over \mathbb{Q} if and only if they are equivalent over \mathbb{Q}_p for all p . While it may seem tempting to extend this method to classify quadratic forms over \mathbb{Z} , the theory encounters limitations. If two forms are equivalent over the p -adic integers \mathbb{Z}_p for all p (including $p = \infty$), they are not necessarily equivalent over \mathbb{Z} ; rather, they only belong to the same genus. In a sense, the genus highlights the constraints of local methods.

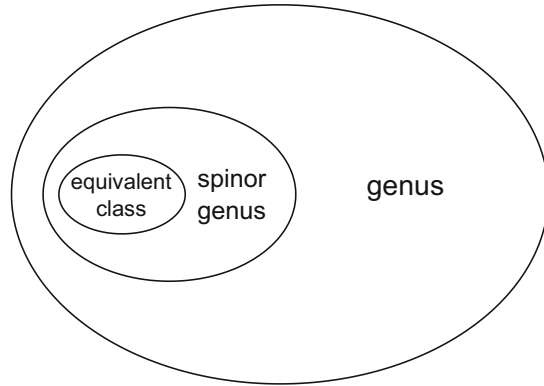


Fig. 1. Relation between the genus and spinor genus of quadratic forms.

The spinor genus is a new classification intermediate between the genus and the integrally equivalent class. It combines local and global methods. To study the spinor genus, we need the theory of Clifford algebra to define the spin group and spinor norm. A Clifford algebra is an algebra generated by a vector space V equipped with a quadratic form, which is a powerful mathematical machinery to study quadratic forms. The spin group $\text{Spin}(n)$ gives a double cover of the special orthogonal group of a vector space. A prototype spin group, $\text{Spin}(3)$, consisting of unit quaternions, is widely used in computer graphics to rotate objects in 3 dimensions. The spin group is closely related to the spinor kernel, which consists of autometries of a vector space determined by certain elements with spinor norm 1 in the Clifford algebra [8]. Therefore, the spinor kernel is well suited to the study of LIP.

The relation between the genus and spinor genus is illustrated in Fig. 1. Using the language of lattices, we give in Table 1 a more precise comparison of the definitions of the equivalent class, spinor genus and genus. Since both orthogonal group $O(V)$ and spinor kernel $\Theta(V_p)$ are subgroups of orthogonal group $O(V_p)$, it is easy to see the inclusions in Fig. 1. For extensive treatments of quadratic forms, see the classic references [8, 27].

Table 1. Comparison of various classifications of lattices. Γ, Λ are lattices, V is a vector space, and subscript p denotes localization. See Sects. 2, 3 for details.

Classification	Definition	Transform	Remark
Class	$\Gamma = \gamma\Lambda$	$\gamma \in O(V)$	O orthogonal group
Spinor genus	$\Gamma_p = \gamma\delta_p\Lambda_p, \forall \gamma \in O(V), \delta_p \in \Theta(V_p)$		Θ spinor kernel
Genus	$\Gamma_p = \beta_p\Lambda_p, \forall p$	$\beta_p \in O(V_p)$	p prime

1.2 Main Results

It is well known that the while genera of quadratic forms exist in any dimension $n \geq 3$ which contain multiple spinor genera, such genera are in some sense rare. To each quadratic form is associated a *Jordan p -symbol*, which classifies the genus of the form. We show that the proportion of Jordan p -symbols which correspond to genera which split into multiple spinor genera is a vanishing fraction of all possible Jordan p -symbols. We summarise this as

Theorem 1. *(Informal) For the set of quadratic forms of prime power determinant and rank such that the genus could split into multiple spinor genera, only a negligible proportion of such forms do in fact lie in such genera.*

A similar result holds for composite determinant. We then proceed to study algorithms to compute the number of spinor genera in a genus, and whether two forms lie in the same spinor genus. We consider such algorithms first over the rational integers, and then over rings of integers in number fields. The latter case is extracted from the work of [4]. We include discussion of the complexity of this algorithm. Summarising the results of Sect. 5, we find:

Theorem 2. *Let F be a number field with ring of integers \mathcal{O}_F is a PID. Assume V is an n -dimensional vector space over F with a non-degenerate quadratic form ϕ and associated symmetric bilinear form b , and $n \geq 3$. Suppose L and \tilde{L} are quadratic lattices on V and they are in the same genus. Then there is an effective algorithm to decide if $\tilde{L} \in \text{spn}^+(L)$, the proper spinor genus of L .*

We also discuss the barriers to this algorithm being efficient. Currently, the complexity of these algorithms to compute the spinor genus appears to be super-exponential, and we welcome further research to reduce their complexity.

Finally, we study the special case of integral binary quadratic forms over the ring of integers of a number field. This is of particular cryptographic interest, since HAWK relies on the hardness of these instances. In this case, when the ring of integers of the number field is a principal ideal domain (PID), it turns out that the spinor genus can be computed via a particularly simple algorithm: deciding if two forms lie in the same spinor genus is equivalent to deciding quartic residuosity in a certain class group, which can be done efficiently using (quantum) algorithms by Hallgren. We note that in the case of forms over \mathbb{Z} , a similar result was proved in [19]; we rely on the subsequent work of [16, 17].

To state our result, let F be a number field with ring of integers \mathcal{O}_F . If (V, ϕ) is a quadratic space over F and we need not reference ϕ , we may simply write V for the space; in the binary case, when (V, ϕ) is anisotropic, we may view V as a quadratic field extension of F with ring of integers \mathcal{O}_V . Let the proper spinor genus of a quadratic form g be written $\text{spn}^+(g)$. Let L_g be the lattice corresponding to the quadratic form g . Finally, denote the left order of a lattice L by $\mathcal{O}_l(L) := \{x \in V : xL \subset L\} \subset V$. We prove:

Theorem 3. *Let F be a number field with ring of integers \mathcal{O}_F is a PID. Let f and g be two anisotropic binary quadratic forms, integral over \mathcal{O}_F , in the same*

genus. Let V be the dimension 2 quadratic space containing L_f and L_g . Then if $L_f \cdot L_g^{-1}$ generates an ideal coprime to the conductor of $\mathcal{O}_l(L_g)$ in \mathcal{O}_V , there is a quantum polynomial time algorithm to decide if $f \in \text{spn}^+(g)$.

We note that this does not affect HAWK, since HAWK uses integral binary Hermitian forms over cyclotomic fields of conductor $n \in \{512, 1024\}$. The cyclotomic field of largest conductor such that it has ring of integers a PID has $n = 90$. However, our work complements that of [26], since they show that module-LIP over the ring of integers of totally real fields has an efficient solution, and we note that the class number of maximal totally real subfields of cyclotomic fields of power-of-two conductor is believed to be 1 for all powers of two, and that this is confirmed up to $n = 256$, and assuming GRH, for $n = 512$ [25]. Our result does not just hold for these (maximal totally real sub-) fields, however, since it applies to all integral binary quadratic forms over number fields with ring of integers a PID.

1.3 Paper Organisation

After providing some background, we then define the notion of spinor genera in Sect. 3. We begin with quadratic forms over \mathbb{Z} : in Sect. 3.3, a step is taken towards understanding ‘how often’ genera split into distinct spinor genera, while in Sect. 4, an algorithm is presented to compute the spinor genus of a positive definite integral quadratic form, adapted from Conway and Sloane [12, Chapter 15]. After this, we move to quadratic forms over number fields: in Sect. 5 Conway and Sloane’s algorithm is extended to lattices over number fields; in Sect. 6 we specialise to binary quadratic forms over maximal orders of number fields, and give a quantum polynomial time algorithm to decide if two forms lie in the same spinor genus. We conclude the paper in Sect. 7, applying our results to LIP and commenting on their applicability to existing schemes.

2 Preliminaries

2.1 Notation

We write $[n]$ for the set of integers $\{1, \dots, n\}$. For any field F , we denote by F^\times its group of units. For two orthogonal subspaces V_1, V_2 we use $V_1 \perp V_2$ to denote their direct sum. The dual space of V will be denoted \widehat{V} .

2.2 Lattices

A lattice is a discrete additive subgroup of \mathbb{R}^n . A lattice L can be generated by a number of linearly independent vectors $\mathbf{b}_1, \dots, \mathbf{b}_m$ that form a basis $B = [\mathbf{b}_1, \dots, \mathbf{b}_m]$, and if $m = n$ then L is full-rank. A lattice L with basis B may be written $L = L(B)$.

One may consider lattices, more abstractly, as discrete additive subgroups of a vector space V over a field F . The case of the previous paragraph is then that of $F = \mathbb{R}$. We state an important theorem for lattices, known as the *Invariant Factors Theorem*:

Theorem 4. [27, §81D] Let L_1 and L_2 be lattices on a vector space V/F . Then there is a basis x_1, \dots, x_n for V such that

$$L_1 = \mathfrak{a}_1 x_1 + \dots + \mathfrak{a}_n x_n \text{ and } L_2 = \mathfrak{a}_1 \mathfrak{t}_1 x_1 + \dots + \mathfrak{a}_n \mathfrak{t}_n x_n$$

where the \mathfrak{a}_i and \mathfrak{t}_i are fractional ideals satisfying $\mathfrak{t}_1 \supset \mathfrak{t}_2 \supset \dots \supset \mathfrak{t}_n$. Moreover, the \mathfrak{t}_i satisfying the above are unique.

2.3 Quadratic Forms

Let F be a number field with characteristic not equal to 2, and \mathcal{O}_F be the ring of integers of F . A quadratic form is a homogeneous polynomial of degree two, written $f(\mathbf{x}) = \sum_{i,j=1}^m a_{ij} x_i x_j$, with coefficients a_{ij} lying in F . Such a form can be associated to an $m \times m$ symmetric matrix $A_f = (a_{ij})_{i,j}$. The determinant of f is the determinant of A_f .

We say that two quadratic forms f, g are *equivalent* over \mathcal{O}_F if and only if there exists $U \in GL_m(\mathcal{O}_F)$ such that $A_g = U^T A_f U$. This is an equivalence relation, and the classes obtained from the quotient by this relation are called *classes* of quadratic forms.

Definition 1. A quadratic form f is called *isotropic* if there exists $x \in V \setminus \{0\}$ such that $f(x) = 0$. If no such x exists, we call f *anisotropic*.

When F is a totally real number field (i.e., all embeddings of F into \mathbb{C} are real), we will be most concerned with certain families of quadratic forms:

Definition 2. We call f *positive definite* if $f(x)$ is totally positive (i.e., all conjugates of $f(x)$ are positive) for any $x \in V \setminus \{0\}$, and f *negative definite* if $f(x)$ is totally negative (i.e., all conjugates of $f(x)$ are negative) for any $x \in V \setminus \{0\}$. Otherwise, we call f *indefinite*.

All forms over totally real number fields below will be assumed positive definite. Note if a form is positive definite, it is anisotropic. One may then obtain the Cholesky decomposition of A_f , $A_f = B_f^T B_f$, so one can always write the symmetric matrix of a quadratic form in such a manner. We denote the lattice with basis B satisfying $A_f = B^T B$ by $L_f = L(B)$.

Given a lattice L with basis B , one can form the symmetric matrix $B^T B$. This can then be considered as the matrix corresponding to a quadratic form f . Thus one can move between the ‘world’ of lattices and the ‘world’ of quadratic forms. In this vein, we call the pair of a vector space V over F and a quadratic form mapping from V to F , say ϕ , a *quadratic space*. We call V *regular* if $\det \phi \neq 0$.

To any quadratic form ϕ on V is also associated a symmetric bilinear form $b : V \times V \rightarrow F$, which can be constructed via the *polarisation identity*

$$b(v, w) = \frac{1}{2}(\phi(v + w) - \phi(v) - \phi(w))$$

2.4 Orthogonal Groups

Let (V, ϕ) be a quadratic space. We may then consider the isomorphisms $\sigma : V \rightarrow V$ such that $\phi(\sigma x) = \phi(x)$, that is the set of automorphisms preserving the quadratic form. This collection forms a group known as the *orthogonal group* $O(V)$ of V . These are linear transformations, so we can define the determinant of σ to be the determinant of the corresponding linear transformation of V , fixing some basis of V/F . An element of the orthogonal group has either determinant equal to 1 or -1 ; it thus contains a subgroup known as the *proper orthogonal group*; we have

$$O(V) = \{\sigma : V \rightarrow V : \phi(\sigma x) = \phi(x) \forall x\} \text{ and } O^+(V) = \{\sigma \in O(V) : \det \sigma = 1\}$$

These notions have analogues for lattices within a quadratic space: for any lattice $L \subset V$ we set

$$O(L) = \{\sigma \in O(V) : \sigma L = L\} \text{ and } O^+(L) = \{\sigma \in O^+(V) : \sigma L = L\}$$

Important subsets of $O(V)$ are the involutions and symmetries. An element $\sigma \in O(V)$ is called an *involution* if $\sigma^2 = \text{Id}$. There is a family of involutions known as symmetries, which we will use below: we say an involution τ is a *symmetry*¹ if there is some fixed anisotropic vector $y \in V$ such that

$$\tau(x) = \tau_y(x) := x - \frac{b(x, y)}{\phi(y)} y$$

for all $x \in V$.

We can use the orthogonal group to give a definition of equivalence of lattices: we say Γ is equivalent to Λ if and only if there exists some $\gamma \in O(V)$ such that $\Gamma = \gamma\Lambda$.

2.5 p -Adic Integers

We give a brief introduction to p -adic arithmetic; for a fuller introduction aimed at cryptographers, see for example [10].

Let p be a prime and $\frac{a}{b} \in \mathbb{Q}^\times$. We may then write $\frac{a}{b} = p^i \cdot \frac{a'}{b'}$ uniquely with both a', b' coprime with p and $i \in \mathbb{Z}$. We may then define the p -adic norm on \mathbb{Q} as $|\frac{a}{b}|_p := p^{-i}$. One may verify that this satisfies the properties of a norm, and satisfies the ultrametric inequality.

Taking the completion of \mathbb{Q} with respect to $|\cdot|_p$ for some fixed prime p yields the p -adic rationals \mathbb{Q}_p . This contains a subring \mathbb{Z}_p , the p -adic integers, defined

$$\mathbb{Z}_p := \{x \in \mathbb{Q}_p : |x|_p \leq 1\}$$

One may view \mathbb{Z}_p as the completion of \mathbb{Z} with respect to the p -adic norm.

¹ Sometimes known as a ‘reflection’.

It will also be useful to consider another subring of \mathbb{Q}_p , the localisation of \mathbb{Z} at a prime p :

$$\mathbb{Z}_{(p)} := \left\{ \frac{a}{b} : a \in \mathbb{Z}, b \in \mathbb{Z} \setminus p\mathbb{Z} \right\}$$

This is in fact both a subring of \mathbb{Z}_p and a subring of \mathbb{Q} .

We now record some useful properties of \mathbb{Z}_p and $\mathbb{Z}_{(p)}$. We begin with units. We have

$$\mathbb{Z}_p^\times = \{x \in \mathbb{Q}_p : |x|_p = 1\} \text{ and } \mathbb{Z}_{(p)}^\times = \left\{ \frac{a}{b} \in \mathbb{Z}_{(p)} : \gcd(a, p) = 1 \right\}$$

Both rings each has only one prime ideal, which is therefore maximal:

$$\text{Spec}(\mathbb{Z}_p) = p\mathbb{Z}_p \text{ and } \text{Spec}(\mathbb{Z}_{(p)}) = p\mathbb{Z}_{(p)}$$

The units therefore are

$$\mathbb{Z}_p^\times = \mathbb{Z}_p \setminus p\mathbb{Z}_p \text{ and } \mathbb{Z}_{(p)}^\times = \mathbb{Z}_{(p)} \setminus p\mathbb{Z}_{(p)}$$

We may define an equivalence relation on \mathbb{Q}_p^\times as follows: we say $a \sim b$ if and only if $\frac{a}{b} \in (\mathbb{Q}_p^\times)^2$. The quotient of \mathbb{Q}_p^\times by this relation yields a number of p -adic *square classes*. We list the possible classes here, for future reference, categorised by the value of p (following the notation of [12]; for more detail see [8]):

1. $p = \infty$ (i.e. the case of \mathbb{R}): we have representatives u and $-u$, where u is any strictly positive number.
2. $p = 2$: we have 8 classes, with representatives $u_1, u_3, u_5, u_7, 2u_1, 2u_3, 2u_5, 2u_7$, where $u_i \in \mathbb{Z}_2^\times$ satisfies $u_i \equiv i \pmod{8}$.
3. $p > 2$: we have 4 classes, with representatives u_+, u_-, pu_+, pu_- , where u_+ (u_- respectively) $\in \mathbb{Z}_p^\times$ is a quadratic residue (nonresidue respectively).

p -adic Diagonalisation. Given any quadratic form f , it is possible to diagonalise the matrix A_f over the p -adic integers, and in fact diagonalise A_f over the subring $\mathbb{Z}_{(p)} \subset \mathbb{Z}_p$ (except that this is a block-diagonalisation if $p = 2$). The algorithm to perform this diagonalisation is given in [12], and runs as follows: find the entry of A_f least divisible by p . If this entry is on the diagonal, begin diagonalising as usual (subtracting multiples of rows and columns from one another). If this entry is off the diagonal in the (i, j) th position, add the j th row to the i th row and the j th column to the i th column, and proceed as before. If $p = 2$, it is possible to obtain a block of the form

$$2^k \begin{pmatrix} a & b \\ b & c \end{pmatrix}$$

for some k , where a, c are even and b is odd, instead of a wholly diagonal matrix.

One then obtains a corresponding decomposition of f over \mathbb{Z}_p . If $p > 2$, this has the form

$$f = f_0 \oplus pf_1 \oplus \dots \oplus p^k f_k \oplus \dots$$

where the f_i are p -adically integral represented by diagonal $n_{p^i} \times n_{p^i}$ matrices with $\gcd(\det f_i, p) = 1$, and if $p = 2$ the f_i may possibly be represented by the 2×2 matrices given above. The f_i are called the *Jordan constituents* of f .

Genera. We say f and g lie in the same *genus* if and only if they are locally equivalent for all primes p , and over the reals; that is to say, we have

$$A_g \sim_{\mathbb{Z}_p} A_f \quad \forall p$$

and $A_g \sim_{\mathbb{R}} A_f$, which is the case if and only if there exist $U_p \in GL_m(\mathbb{Z}_p)$ such that $A_g = U_p^T A_f U_p$ for all p , and $U \in GL_m(\mathbb{R})$ such that $A_g = U^T A_f U$. There are finitely many genera with the given determinant and dimension, and each genus is a finite disjoint union of equivalence classes.

Equivalently, in terms of lattices we say that Γ and Λ which are on the same space V lie in the same genus if there exist $\beta_p \in O(V_p)$ such that $\Gamma_p = \beta_p \Lambda_p$ for all primes p .

p-Adic Norms and Number Fields The above can all be extended to algebraic number fields. We assume some familiarity with the splitting and ramification of primes in rings of integers of number fields; for background, see for example [24]. We begin with a definition: say two norms $|\cdot|_1, |\cdot|_2$ are equivalent if there exists some $\varrho \in \mathbb{R}^+$ such that $|\cdot|_1^\varrho = |\cdot|_2$.

Let F be a number field with ring of integers \mathcal{O}_F . Let \mathfrak{p} be a prime ideal of \mathcal{O}_F . Then we say there is a *norm* of F corresponding to each prime ideal \mathfrak{p} of \mathcal{O}_F , each embedding σ of F into \mathbb{R} , and each pair of embeddings into \mathbb{C} . The latter two kinds of norms are called *real* and *complex* respectively.

We first consider the norms associated to prime ideals. For any $\alpha \in F^\times$, let $(\alpha) = \mathfrak{p}^i \prod_j \mathfrak{p}_j^{e_j}$ be the factorisation of (α) into products of prime ideals. We then define $|\alpha|_{\mathfrak{p}} = N_{F/\mathbb{Q}}(\mathfrak{p})^{-i}$. By a *prime spot* \mathfrak{p} we mean the equivalence class of norms containing $|\cdot|_{\mathfrak{p}}$. We may then consider the completion $F_{\mathfrak{p}}$ of F under $|\cdot|_{\mathfrak{p}}$.

For a real embedding σ , we define a norm $|\cdot|_{\sigma} = |\sigma(\cdot)|$, the absolute value of the embedding into \mathbb{R} . We call the equivalence class of norms containing $|\cdot|_{\sigma}$ a *real spot*. We can define the complex spots in a similar manner.

2.6 Jordan p -Symbols

From a p -adic diagonalisation of f as above, one can read off a number of invariants of f , which classify the genus of f . In fact, with $\left(\frac{a}{p}\right)$ denoting the Legendre symbol, one can say

Theorem 5. [12, Chapter 15, Theorem 9] For $p \neq 2$, f is equivalent to g over \mathbb{Z}_p if and only if the precise powers of p , the dimensions n_{p^i} , and the signs $\epsilon_{p^i} = \left(\frac{\det f_i}{p}\right)$ occurring in their Jordan decompositions are identical.

These invariants are encoded in the Jordan p -symbol (called the ‘ p -adic symbol’ in [12]), which is a formal product of factors q^{ϵ_a, n_a} .

A similar but more complicated result holds for $p = 2$, which we omit for brevity.

3 Spinor Genera and Proportion of Splitting Genera

In this section we define the spinor genus of a quadratic form. In order to make the definition precise, we proceed via Clifford algebra.

3.1 Clifford Algebra

Let (V, ϕ) be a regular quadratic space of dimension n over a field K . There is a unique algebra $C(V)$ over K of dimension 2^n satisfying

1. $C(V)$ is spanned by 1 and formal products $x_1 \dots x_r$, $x_i \in V$,
2. $xx = \phi(x)$ for $x \in V$.

If V has normal basis e_1, \dots, e_n , then in $C(V)$ we have $e_i e_j = -e_j e_i$ and $e_i e_i = \phi(e_i) \in K$. This $C(V)$ is the Clifford algebra associated to V .

For $J \subset [n]$ with $j_1 < \dots < j_r$, define $e(J) := e_{j_1} e_{j_2} \dots e_{j_r}$ and set $e(I)e(J) = \ell(I, J)e(K)$ where $K = \{i : i \in I \text{ or } i \in J, i \notin I \cap J\}$, and

$$\ell(I, J) = \left(\prod_{i \in I, j \in J, i > j} -1 \right) \cdot \left(\prod_{i \in I \cap J} \phi(e_i) \right).$$

Sending $x \mapsto -x \in V$ extends to give an automorphism of $C(V)$. Set

$$C_0(V) = \{u \in C(V) : u \text{ is fixed by the above automorphism}\}.$$

The *involution* on $C(V)$ is defined by extending linearly the map

$$e(J) = e_{j_1} e_{j_2} \dots e_{j_r} \mapsto e_{j_r} e_{j_{r-1}} \dots e_{j_1} =: e(J)'$$

This involution² satisfies

1. $(u')' = u$ for all $u \in C(V)$
2. $u' = u$ if $u \in V$
3. $(uv)' = v'u'$ for any $u, v \in C(V)$.

For $u \in C(V)$, if u is (multiplicatively) invertible define $T_u : x \mapsto u x u^{-1}$. Then

Lemma 1. [8, Chapter 10, Lemma 3.1] *If $u \in C(V)$ is invertible and $T_u(x) = u x u^{-1} \in V$ for all $x \in V$, then $T_u \in O(V)$.*

Inspired by this, define the group

$$M_0(V) = \{u \in C_0(V) : u^{-1} \text{ exists, } T_u : V \rightarrow V\}$$

Then

² The involution (sometimes called the ‘transpose’) is not to be confused with the automorphism used to define $C_0(V)$.

Lemma 2. [8, Chapter 10, Theorem 3.1] $O^+(V) \cong M_0(V)/K^\times$.

Moreover, if $u \in M_0(V)$, then $u = a_1 \dots a_r$ for an even r , $a_j \in V$, and $uu' \in K^\times$. Define the spin group

$$\text{Spin}(V) = \{u \in M_0(V) : uu' = 1\}$$

and

$$\Theta(V) = \{T_u : uu' = 1\}.$$

The latter is called the *spinor kernel*, a name which will be clarified in the following section. We now relate $\text{Spin}(V)$ and $\Theta(V)$:

Theorem 6. [8, Chapter 10, Theorem 3.3] *There is an homomorphism*

$$\text{Spin}(V) \rightarrow \Theta(V), \quad u \mapsto T_u$$

with kernel $\{\pm 1\}$.

Spinor Norm. We reach the application of the theory developed in the previous sections. Let $\sigma \in O^+(V)$. Then we can write σ as a map T_u for some $u \in M_0(V)$, and then map $u \mapsto uu' \pmod{(K^\times)^2}$. The composition of these maps is called the *spinor norm*:

Theorem 7. [8, Chapter 10, Corollary 3] *The map $\theta : \sigma \mapsto uu' \pmod{(K^\times)^2}$ is a multiplicative homomorphism.*

Proof. It is plain that the identity maps to the identity. Consider $\sigma, \tau \in O^+(V)$. We show $\theta(\sigma\tau) = \theta(\sigma)\theta(\tau)$.

First note that by Lemma 2, $\sigma\tau$ corresponds to some product $uv \in M_0(V)/K^\times$. Then $\theta(\sigma\tau) = (uv)(uv)' = uvv'u' = uu'vv'$ since $vv' \in K^\times$. Moreover, we have $\theta(\sigma)\theta(\tau) = uu'vv'$.

3.2 Defining the Spinor Genus

The spinor norm can be used to define an equivalence relation on the space of quadratic forms, via lattices.

Definition 3. Let Γ, Λ be lattices on the quadratic space (V, ϕ) . Say $S(\Gamma, \Lambda)$ holds if there exist $\gamma \in O^+(V)$ and $\delta_p \in \Theta(V_p)$ such that

$$\Gamma_p = \gamma\delta_p\Lambda_p, \text{ for all } p.$$

We call the equivalence classes under this relation the (proper) *spinor genera*. Note that if $f \sim g$, then setting the δ_p to be the identity for all p implies that $S(L_f, L_g)$ holds. So equivalent lattices lie in the same spinor genus. Moreover, if $S(L_f, L_g)$ holds, then since $\Theta(V_p) \subset O(V_p)$, we have $g \in \text{gen}(f)$. The following demonstrates that the spinor genus truly is an ‘intermediate’ relation to the class and the genus:

Theorem 8. [8, Chapter 11, Lemma 1.4] $S(\Gamma, \Lambda)$ is an equivalence relation.

Proof. Symmetry and reflexivity are straightforward; here we demonstrate transitivity. Suppose Γ, Λ, Δ are three lattices satisfying $S(\Gamma, \Lambda)$ and $S(\Lambda, \Delta)$. Then there are $\gamma_1, \gamma_2 \in O^+(V)$ and $\beta_{1p}, \beta_{2p} \in \Theta(V_p)$ such that $\Gamma_p = \gamma_1 \beta_{1p} \Lambda_p$ and $\Lambda_p = \gamma_2 \beta_{2p} \Delta_p$ for each prime p . Combining these, one has

$$\Gamma_p = \gamma_1 \beta_{1p} \gamma_2 \beta_{2p} \Delta_p = (\gamma_1 \gamma_2) (\gamma_2^{-1} \beta_{1p} \gamma_2 \beta_{2p}) \Delta_p$$

for each prime p . It is easy to check $\gamma_1 \gamma_2 \in O^+(V)$ and $\gamma_2^{-1} \beta_{1p} \gamma_2 \beta_{2p} \in \Theta(V_p)$. Hence $S(\Gamma, \Delta)$ holds.

We record some standard facts on the set of spinor genera [8, Chapter 11]:

- Proposition 1.**
1. The number of spinor genera in any genus is a power of 2.
 2. For all $n \geq 3$ there exist lattices whose genus contains multiple spinor genera.
 3. Let (V, ϕ) be a quadratic space of dimension $n \geq 3$, $\Lambda \subset V$ a lattice, and ϕ takes integral values on Λ . If $\text{gen}(\Lambda)$ contains multiple spinor genera, either there exists $p > 2$: $p^{\frac{n(n-1)}{2}} \mid \det(\Lambda)$, or $2^{n(n-3)/2 + \lfloor (n+1)/2 \rfloor} \mid \det(\Lambda)$.

3.3 Proportion of Genera Splitting Into Multiple Spinor Genera

In this section we obtain an upper bound on the number of Jordan p -symbols corresponding to forms which lie in a genus which splits into multiple spinor genera. In the rest of this subsection we will call such genera, forms in a given genera, and their corresponding p -symbols, ‘splitting’, for convenience. The point of this result is to show that a negligible number of such p -symbols in the prime-power case correspond to forms in such genera, when $n \geq 3$.

Odd Prime-Power Determinant. We begin by considering forms of rank $n \geq 3$ and determinant $p^{n(n-1)/2}$ for some prime $p > 2$, as this is the minimal prime-power determinant for which the genus of a form specified by rank and determinant can split into more than one spinor genus (cf. Proposition 1). Observe that a necessary condition for such splitting is that the Jordan decomposition of the form has no component with dimension larger than one, i.e. the prime powers occurring in the p -adic diagonalisation of the form are all distinct (this may be seen from Sect. 4, (i)). That is, up to multiplication by p -adic quadratic residues or non-residues on the diagonal elements and reordering, f has corresponding matrix A p -adically diagonalising to

$$A_{D,p} = \begin{pmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & p & 0 & \dots & 0 \\ 0 & 0 & p^2 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & p^{n-1} \end{pmatrix},$$

which has determinant $p^{n(n-1)/2}$. If a form has determinant a higher power of p , the proportion of forms will be upper bounded by the below result also. We do not address the power-of-two case, for simplicity.

We will proceed by (straightforwardly) finding the number of splitting Jordan p -symbols, and we then lower bound the total number of p -symbols for forms with the above-specified parameters (rank $f = n$, $\det f = p^{n(n-1)/2}$). In the latter instance, f has corresponding matrix A p -adically diagonalising to

$$A_{D,p} = \begin{pmatrix} p^{i_1} & 0 & \dots & 0 \\ 0 & p^{i_2} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & p^{i_n} \end{pmatrix}, \tag{1}$$

up to multiplication by quadratic residues (or non-residues), with determinant $p^{n(n-1)/2}$, so $\sum_j i_j = n(n-1)/2$. In the more general case of determinant p^m for some $m > n(n-1)/2$, one has $\sum_j i_j = m$.

Step 1: splitting symbols. To find the number of splitting Jordan p -symbols, observe from the algorithm reproduced above that we must have distinct prime powers on the diagonal of $A_{D,p}$. Thus the Jordan p -symbol of a splitting form must be $1^{\epsilon_1,1} p^{\epsilon_p,1} p^{2\epsilon_{p^2},1} \dots p^{n-1\epsilon_{p^{n-1}},1}$. Moreover, in the case of prime-power determinant, all the ϵ_i agree: the diagonals must all be distinct powers of p , either multiplied by quadratic residues modulo p , or all multiplied by quadratic non-residues modulo p . So there are at most two splitting p -symbols for such fixed parameters.

Step 2: bounding the number of p -symbols. We lower bound the number of p -symbols for the family of forms mentioned above. The number we are seeking to approximate is the following:

$$A_{n,m} := \left| \left\{ 1^{\epsilon_1, n_1} p^{\epsilon_p, n_p} \dots p^{n-1\epsilon_{p^{n-1}}, n_{p^{n-1}}} : \sum_{i=0}^{n-1} n_{p^i} = n \text{ and } \sum_{i=0}^{n-1} i n_{p^i} = m \right\} \right|,$$

initially in the specific instance $m = n(n-1)/2$. By observing (1), it is sufficient to count the number of n -tuples (i_1, \dots, i_n) satisfying $\sum_j i_j = m$, each multiplied by 2^{k-1} , where k is number of distinct terms in the n -tuple; this multiplicative factor, since each entry of the diagonal of (1) may be multiplied by a quadratic residue or non-residue.

We use some elementary partition theory to conduct this counting exercise; for more background on partitions, see for example [3].

Definition 4. Let l be an integer. A decomposition $l = \lambda_1 + \dots + \lambda_k$ into positive integers λ_i is called a partition of l . The λ_i are called the parts of the partition, and k is called the length of the partition. The number of partitions of an integer l is denoted by $p(l)$.

Now let

$$p_{i,j}(l) := |\{\text{partitions of } l \text{ of length } i \text{ with } j \text{ distinct parts}\}|.$$

Write $r := n(n - 1)/2$. One can then express $A_{n,r}$ as a weighted sum of $p_{i,j}(r)$:

$$A_{n,r} = \sum_{(i,j) : i \geq j \geq 1}^n p_{i,j}(r) \cdot 2^j,$$

because of the quadratic residues or non-residues on each diagonal term of (1).

Pick (i, j) such that $i + j$ is maximal in $[2, 2n]$ with respect to the property that $p_{i,j}(r) \neq 0$. Then $A_{n,r} > p_{i,j}(r) \cdot 2^j$. So we now proceed to find (i, j) maximal with respect to $i + j$ such that $p_{i,j}(r) \neq 0$.

Recall $r = n(n - 1)/2$. Then

$$p_{i,j}(r) = |\{\text{partitions of } n(n - 1)/2 \text{ of length } i \text{ with } j \text{ distinct parts}\}|$$

is $p_{i,j}(\cdot)$ applied to the $(n - 1)$ th triangle number, which has a maximal partition of length $n - 1$ into distinct parts of length $n - 1$ by definition. So $(i, j) = (n - 1, n - 1)$ is a pair satisfying $i + j$ is maximal and $p_{i,j}(r)$ is non-zero. Then

$$\begin{aligned} A_{n,r} &= \sum_{i \geq j \geq 1}^n p_{i,j}(n(n - 1)/2) \cdot 2^j \\ &> p_{n-1,n-1}(n(n - 1)/2) \cdot 2^{n-1} = 2^{n-1}. \end{aligned}$$

Now suppose $\det f \geq p^m$ for odd prime p and integer $m \geq r$. The number of splitting p -symbols in this case is upper bounded by

$$2(p_{n,n}(m) + p_{n-1,n-1}(m)) \leq 4p_{n-1,n-1}(m)$$

To bound the total number of symbols, observe that $p_{n-1,n-1}(m) \geq 1$ for $m \geq r$. From the above reasoning it follows that a lower bound on $A_{n,m}$ is

$$A_{n,m} = \sum_{i \geq j \geq 1}^n p_{i,j}(m) \cdot 2^j \geq p_{n-1,n-1}(m) \cdot 2^{n-1} \geq 2^{n-1}$$

Step 3: Proportion of Splitting Forms for Odd Prime-Power Determinants Since there are at most 2 splitting Jordan p -symbols, for a form of determinant p^r and rank n , the fraction of symbols corresponding to forms whose genus splits into multiple spinor genera is less than $2/2^{n-1} = \frac{1}{2^{n-2}}$, which is a negligible function in n . We have thus arrived at

Theorem 9. *Let $p > 2$ be a prime. The proportion of splitting p -symbols corresponding to positive definite quadratic forms of rank $n \geq 3$ and determinant $p^{n(n-1)/2}$ among all possible p -symbols is strictly less than $2^{-(n-2)}$.*

When $m > r$, an upper bound for the proportion of splitting p -symbols is

$$\frac{4p_{n-1,n-1}(m)}{\sum_{i \geq j \geq 1} p_{i,j}(m) \cdot 2^j}$$

The proportion can then be bounded by considering $i = j = n - 1$:

$$\frac{4p_{n-1,n-1}(m)}{\sum_{i \geq j \geq 1}^n p_{i,j}(m) \cdot 2^j} \leq \frac{4}{2^{n-1}} = \frac{1}{2^{n-3}},$$

which is a negligible function in n .

For the case of odd composite determinants, we note the following. As above, we have a set \mathcal{S} , comprised of prime divisors of $2d$ and ∞ . Suppose $|\mathcal{S}| = t$. We then have to compute the Jordan p -symbols at each element of this set. By Proposition 1, there must be at least one prime divisor of the determinant which divides the determinant many times, and any element of \mathcal{S} may be this divisor. We can thus upper bound the proportion of splitting symbols by $\frac{t}{2^{n-3}}$, which is negligible when t is polynomially large.

We leave the $p = 2$ case to the interested reader, for brevity; the details of this case can be found in [12].

4 An Algorithm to Compute Spinor Genera

Inspired by Conway and Sloane [12, Chapter 15], in this section we provide a (quantum) polynomial time algorithm to calculate the number of spinor genera in the given genus with rank n is at least 3. Let f and g be two positive definite quadratic forms with determinant d in the same genus. In view of [30, Theorem 50], there is a rational matrix M such that $A_f = M^t A_g M$ with $|\det M| = 1$ and denominators of its entries are relatively prime to $2d$. Let L_f and L_g be the corresponding lattices that reflect this property. Then $[L_f : L_f \cap L_g] = [L_g : L_f \cap L_g] = r$ for some integer r which is relatively prime with $2d$.

Let \mathcal{S} be the finite set of prime divisors of $2d$. We denote a spinor operator by a sequence $(\dots, r_p, \dots)_{p \in \mathcal{S}}$, where r_p is a p -adic unit square class. For each $p \in \mathcal{S}$, choose a proper isometry $\sigma_p \in O^+(V_p)$ with $\theta(\sigma_p) = r_p$. Let L_h be the lattice in the genus of L_f with $(L_h)_p = \sigma_p(L_f)_p$ for each $p \in \mathcal{S}$ and $(L_h)_p = (L_f)_p$ for each $p \notin \mathcal{S}$. Then the spinor operator $(\dots, r_p, \dots)_{p \in \mathcal{S}}$ sends $\text{spn}^+(f)$ to $\text{spn}^+(h)$ where h is a quadratic form defined on L_h . More information about the action of a spinor operator can be found in [8, Chapter 11]. In this notation the group operation is componentwise multiplication, and the rational or p -adic integers can be regarded as spinor operators in the following way. For a rational integer r that is relatively prime to $2d$, the corresponding spinor operator is $\Delta(r) = (r, \dots, r)$; for a p -adic integer $A_p = p^k a$ there corresponds the spinor operator $\Delta_p(A_p) = (p^k, \dots, p^k, a, p^k, \dots, p^k)$ whose q -coordinate for $q \neq p$ is the q -adic unit square class of p^k and whose p -coordinate is the p -adic unit square class of a .

By [8, Theorem 4.1 in Chap. 11], $\Delta(r) * \text{spn}^+(f) = \text{spn}^+(g)$ where $r = [L_f : L_f \cap L_g]$. Moreover when the rank is at least 3, $\Delta(r) * \text{spn}^+(f)$ is defined for every positive integer r prime to $2d$. Thus there is a surjective map from the set of spinor operators $\Delta(r)$ with r positive integers prime to $2d$ to the set of spinor genera in $\text{gen}(f)$, and we can determine the number of the spinor genera by determining the kernel of this map, i.e., those $\Delta(r)$ which fix each spinor genus.

By Theorem 16 and Theorem 17 in [12, Chapter 15], the spinor operator kernel consists of the spinor operators $\Delta(r)$ for which the positive integer r is an automorphous number (the spinor norm of a proper integral isometry in $O^+(L_f)$) and relatively prime to $2d$. The spinor operator kernel can be calculated locally and is generated by the spinor operators $\Delta_p(A_p)$ for each $p \in \mathcal{S}$ where A_p is a p -adically automorphous number (the spinor norm of a proper p -adic integral isometry in $O^+((L_f)_p)$).

As the spinor operator kernel is completely determined by the p -adically automorphous numbers, it is necessary to introduce an algorithm for identifying the p -adically automorphous numbers associated with the given quadratic form f . Recall that f is diagonalizable at $p \geq 3$, and f is a direct sum of quadratic forms that are of the shapes $2^k(x)$ or $2^k \begin{pmatrix} a & b \\ b & c \end{pmatrix}$ at $p = 2$, where a, c are even and x, b are odd. Now we want to create a set consisting of numbers in the following two parts:

- (I) when $p \geq 3$, all the diagonal entries; when $p = 2$, the diagonal entries $2^k x$.
- (II) only when $p = 2$, the numbers $2^{k+1}u_1, 2^{k+1}u_3, 2^{k+1}u_5, 2^{k+1}u_7$ for every 2-dimensional component $2^k \begin{pmatrix} a & b \\ b & c \end{pmatrix}$.

Then the group of p -adically automorphous numbers is generated by the p -adic square classes of the products of all pairs of numbers from the above list, and supplemented by:

- (i) all p -adic units if either $p \geq 3$ and $\dim f_k \geq 2$ for any k , or $p = 2$ and $2^k f_k \oplus 2^{k+1} f_{k+1} \oplus 2^{k+2} f_{k+2} \oplus 2^{k+3} f_{k+3}$ has dimension ≥ 3 for any k .
- (ii) the square classes $2u_1, 2u_3, u_5, u_3, u_7$ whenever $p = 2$ and part (I) of the list contains two entries whose product has the form $u_1, u_5, (1 \text{ or } 4 \text{ or } 16)u_{\text{odd}}, (2 \text{ or } 8)u_{1 \text{ or } 5}, (2 \text{ or } 8)u_{3 \text{ or } 7}$ respectively.

Remark:

1. If there is a prime p such that, for each p -adic unit u , the spinor operator $\Delta_p(1, \dots, 1, u, 1, \dots, 1)$ is in the spinor operator kernel, then the prime p can be removed from the set \mathcal{S} , as it conveys no information modulo the spinor operator kernel. We call such a prime p tractable.
2. For any spinor operator (r_1, \dots, r_s) , by the Strong Approximation Theorem [27, 21:2], there is a positive integer r such that $\Delta(r) = (r_1, \dots, r_s)$.

Theorem 10. *There is a (quantum) polynomial time algorithm to determine the number of spinor genera in the genus of the given quadratic form f .*

Algorithm 1: An algorithm to determine the number of spinor genera in $\text{gen}(f)$

Input: Quadratic form f

Output: Answer

- 1: Compute the p -adic diagonalisation of f for each $p \mid 2d$
 - 2: Compute p -adically automorphous numbers for each $p \mid 2d$
 - 3: $\mathcal{S} := \{p \mid 2d : p \text{ is intractable}\}$
 - 4: Compute a basis \mathcal{G} of the spinor operators kernel with respect to \mathcal{S} .
 - 5: **if** $2 \in \mathcal{S}$ **then**
 - 6: Output $2^{|\mathcal{S}|+1-|\mathcal{G}|}$,
 - 7: **else**
 - 8: output $2^{|\mathcal{S}|-|\mathcal{G}|}$,
 - 9: **end if**
-

Proof. Let d be the determinant of the given quadratic form f . There are at most $\log_2(2d)$ different prime divisors of $2d$. We can compute in time $\text{poly}(n, \log_2 d)$ the diagonalisation of f_p using the method introduced in Sect. 2.5 and its p -adically automorphous numbers for all primes p dividing $2d$.

When $p \geq 3$, p is tractable if the non-square unit u_- is in $\theta(O^+((L_f)_p))$ or both pu_+ and pu_- are contained in $\theta(O^+((L_f)_p))$. Therefore, if p is intractable, then p contributes at most one non-trivial spinor operator $\Delta_p(pu_+)$ or $\Delta_p(pu_-)$ to the generators of the spinor operator kernel. The prime 2 is tractable if one of the following is true:

1. two of three non-square units u_3, u_5, u_7 are in $\theta(O^+((L_f)_2))$;
2. three of four prime elements $2u_1, 2u_3, 2u_5, 2u_7$ are contained in $\theta(O^+((L_f)_2))$;
3. one non-square unit u and two prime elements whose product is in the different square class from u in $\theta(O^+((L_f)_2))$.

Therefore if 2 is intractable, then 2 contributes at most three non-trivial spinor operators to the generators of the spinor operator kernel. Let $\mathcal{S} = \{p \mid 2d : p \text{ is intractable}\}$. Then there are $2^{|\mathcal{S}|}$ different spinor operators with respect to \mathcal{S} when $2 \notin \mathcal{S}$ and $2^{|\mathcal{S}|+1}$ different spinor operators with respect to \mathcal{S} when $2 \in \mathcal{S}$. Once we can determine the size of the spinor operator kernel with respect to \mathcal{S} , the number of the spinor genera in $\text{gen}(f)$ will be determined.

Let $\mathcal{G} = \{\Delta_1, \dots, \Delta_t\}$ ($t \leq \log_2(8d)$) be the set of different non-trivial spinor operators obtained from the p -adically automorphous numbers for all $p \in \mathcal{S}$. It generates the spinor operator kernel with respect to \mathcal{S} . We can further obtain a basis by applying Gaussian elimination to the matrix G with $\log \Delta_1, \dots, \log \Delta_t$ as its rows. We denote $\log u_1 = 0$ for $p = 2$ and $\log u_+ = 0$ for $p \geq 3$. Suppose $\log \Delta'_1, \dots, \log \Delta'_t$ are all nonzero rows in the ‘‘Echelon form’’ of G , then the set $\tilde{\mathcal{G}} = \{\Delta'_1, \dots, \Delta'_t\}$ is a basis of the spinor operator kernel with respect to \mathcal{S} and can be obtained in time $\text{poly}(\log_2 d)$.

Algorithm 1 shows the pseudocode of the procedure given above. The spinor operator kernel can also help us to identify if two quadratic forms f and g are

Algorithm 2: An algorithm to distinguish the spinor genera

Input: Quadratic forms f and g in the same genus
Output: Answer

- 1: Compute a rational matrix M with denominators of its entries relatively prime to $2d$ by solving the system of quadratic equations $A_f = M^t A_g M$
- 2: Compute a matrix B_g such that $B_g^t B_g = A_g$ using Cholesky decomposition
- 3: $L_g \leftarrow$ lattice generated by B_g
- 4: $B_f := B_g M$ and $L_f \leftarrow$ lattice generated by B_f
- 5: Compute $r = [L_f : L_f \cap L_g]$ using Hermite Normal Form
- 6: Compute the p -adic diagonalisation of f_p for each $p \mid 2d$
- 7: Compute p -adically automorphous numbers and the corresponding spinor operators for each $p \mid 2d$
- 8: **if** $\Delta(r)$ is a product of some spinor operators obtained in the above step **then**
- 9: output ‘same spinor genus’
- 10: **else**
- 11: output ‘different spinor genera’
- 12: **end if**

in the same spinor genus or not: f and g are in the same spinor genus if and only if $\Delta(r)$ where $r = [L_f : L_f \cap L_g]$ is a product of some generators $\Delta_p(A_p)$ where $A_p \in \theta(O^+(L_f)_p)$. This is shown in Algorithm 2, where all the steps can be completed in (quantum) polynomial time except for Steps 1. For Step 1, [12] suggested an exhaustive search through all rational matrices until a rational equivalence with denominator r relatively prime to $2d$ is found. This would cost complexity at least $e^{O(n^2)}$, which may be reduced significantly using a better method. For Step 8, we can determine whether $\Delta(r)$ is in the spinor operator kernel by applying Gaussian elimination in time $\text{poly}(\log_2 d)$ since the number of generators of spinor operator kernel is bounded by $t \leq \log_2(8d)$, as explained above.

5 Spinor Genus Algorithm for Quadratic Forms over Number Fields

In this section, we will investigate the complexity of an algorithm, that was introduced by Benham and Hsia [4], to determine if two quadratic forms over number fields which are in the same genus are in the same spinor genus or not. For the convenience, we discuss it in the lattice setting. This algorithm was implemented in MAGMA (see [11]) but without discussion of its complexity. Let F be an algebraic number field with \mathcal{O} its ring of integers. Assume V is an n -dimensional vector space over F with a non-degenerate quadratic form ϕ and its associated symmetric bilinear form b satisfying $b(v, w) = \frac{1}{2}(\phi(v+w) - \phi(v) - \phi(w))$, and L is a quadratic lattice on V . We assume $n \geq 3$ in the sequel.

Let Ω be the set consisting of all spots on F . For a prime spot $\mathfrak{p} \in \Omega$, define $dL_{\mathfrak{p}}$, the discriminant of $L_{\mathfrak{p}} = \mathcal{O}_{\mathfrak{p}}v_1 + \dots + \mathcal{O}_{\mathfrak{p}}v_n$ with respect to the basis $\{v_1, \dots, v_n\}$, as the determinant of the Gram matrix $A_{L_{\mathfrak{p}}} = (b(v_i, v_j))_{n \times n}$ when

n is even and as half of the determinant when n is odd. We say L is good at \mathfrak{p} (or simply $L_{\mathfrak{p}}$ is good) if $\phi(L_{\mathfrak{p}}) \subseteq \mathcal{O}_{\mathfrak{p}}$ and $dL_{\mathfrak{p}} \in 2^{-n}\mathfrak{u}_{\mathfrak{p}}$ where $\mathfrak{u}_{\mathfrak{p}}$ is the group of units of $F_{\mathfrak{p}}$. Recall that we say $L_{\mathfrak{p}}$ is $\mathcal{O}_{\mathfrak{p}}$ -maximal if $\phi(L_{\mathfrak{p}}) \subseteq \mathcal{O}_{\mathfrak{p}}$ and if for every lattice $N_{\mathfrak{p}}$ with $L_{\mathfrak{p}} \subseteq N_{\mathfrak{p}}$ and $\phi(N_{\mathfrak{p}}) \subseteq \mathcal{O}_{\mathfrak{p}}$ we have $L_{\mathfrak{p}} = N_{\mathfrak{p}}$. The following lemma shows that $L_{\mathfrak{p}}$ is $\mathcal{O}_{\mathfrak{p}}$ -maximal if $L_{\mathfrak{p}}$ is good.

Lemma 3. *When L is good at \mathfrak{p} , $L_{\mathfrak{p}}$ is $\mathcal{O}_{\mathfrak{p}}$ -maximal.*

Proof. Let $A_{L_{\mathfrak{p}}} = (a_{ij})_{n \times n}$ be the Gram matrix of $L_{\mathfrak{p}}$. Define the scale ideal $\mathfrak{s}L_{\mathfrak{p}}$ to be the fractional ideal generated by the entries a_{ij} with $1 \leq i, j \leq n$, and the volume $\mathfrak{v}L_{\mathfrak{p}}$ to be the fractional ideal generated by $\det A_{L_{\mathfrak{p}}}$. Remember that the discriminant $dL_{\mathfrak{p}}$ is $\det A_{L_{\mathfrak{p}}}$ when n is even and is $\frac{1}{2}\det A_{L_{\mathfrak{p}}}$ when n is odd. When L is good at \mathfrak{p} , $\phi(L_{\mathfrak{p}}) \subseteq \mathcal{O}_{\mathfrak{p}}$ and $dL_{\mathfrak{p}} = 2^{-n}u$ with u a \mathfrak{p} -unit. Therefore $\mathfrak{v}L_{\mathfrak{p}} = 2^{-n}\mathcal{O}_{\mathfrak{p}}$ when n is even and $\mathfrak{v}L_{\mathfrak{p}} = 2^{-n+1}\mathcal{O}_{\mathfrak{p}}$ when n is odd.

When \mathfrak{p} is non-dyadic ($|2|_{\mathfrak{p}} = 1$), 2 is a \mathfrak{p} -unit. We have $L_{\mathfrak{p}}$ is unimodular since $\mathfrak{s}L_{\mathfrak{p}} \subseteq \mathcal{O}_{\mathfrak{p}}$ and $\mathfrak{v}L_{\mathfrak{p}} = \mathcal{O}_{\mathfrak{p}}$ (see [27, 82G]). By [27, 82:19], $L_{\mathfrak{p}}$ is $\mathcal{O}_{\mathfrak{p}}$ -maximal.

Now let \mathfrak{p} be dyadic ($0 < |2|_{\mathfrak{p}} < 1$). Suppose that there is a lattice $N_{\mathfrak{p}}$ on $V_{\mathfrak{p}}$ such that $L_{\mathfrak{p}} \subseteq N_{\mathfrak{p}}$ and $\phi(N_{\mathfrak{p}}) \subseteq \mathcal{O}_{\mathfrak{p}}$. We want to show that $L_{\mathfrak{p}} = N_{\mathfrak{p}}$ by comparing their volumes. When n is even, since $\phi(N_{\mathfrak{p}}) \subseteq \mathcal{O}_{\mathfrak{p}}$, we have $\mathfrak{s}N_{\mathfrak{p}} \subseteq \frac{1}{2}\mathcal{O}_{\mathfrak{p}}$ and $\mathfrak{v}N_{\mathfrak{p}} \subseteq (\mathfrak{s}N_{\mathfrak{p}})^n \subseteq 2^{-n}\mathcal{O}_{\mathfrak{p}} = \mathfrak{v}L_{\mathfrak{p}}$. When n is odd, $N_{\mathfrak{p}} = \langle \alpha \rangle \perp N'_{\mathfrak{p}}$ with $\alpha \in \mathcal{O}_{\mathfrak{p}}$. Then $\mathfrak{v}N_{\mathfrak{p}} = \alpha \mathfrak{v}N'_{\mathfrak{p}} \subseteq (\mathfrak{s}N'_{\mathfrak{p}})^{n-1} \subseteq 2^{-n+1}\mathcal{O}_{\mathfrak{p}} = \mathfrak{v}L_{\mathfrak{p}}$. Therefore, by [27, 82:11], $N_{\mathfrak{p}} = L_{\mathfrak{p}}$ and $L_{\mathfrak{p}}$ is $\mathcal{O}_{\mathfrak{p}}$ -maximal.

Suppose \tilde{L} is a lattice on V that is also good at \mathfrak{p} , so $L_{\mathfrak{p}}$ and $\tilde{L}_{\mathfrak{p}}$ are $\mathcal{O}_{\mathfrak{p}}$ -maximal lattices. By [27, 91:2], there is a local basis $\{e_1, f_1, \dots, e_t, f_t, z_{2t+1}, \dots, z_n\}$ for $L_{\mathfrak{p}}$ satisfying $\phi(e_i) = \phi(f_i) = 0$, $b(e_i, f_j) = \frac{1}{2}\delta_{ij}$, $b(e_i, e_j) = b(f_i, f_j) = 0$ for $i \neq j$, $b(e_i, z_k) = b(f_i, z_k) = 0$, the subspace spanned by $\{z_{2t+1}, \dots, z_n\}$ is anisotropic, and

$$\tilde{L}_{\mathfrak{p}} = \mathfrak{p}^{a_1}e_1 + \mathfrak{p}^{-a_1}f_1 + \dots + \mathfrak{p}^{a_t}e_t + \mathfrak{p}^{-a_t}f_t + \mathcal{O}_{\mathfrak{p}}z_{2t+1} + \dots + \mathcal{O}_{\mathfrak{p}}z_n,$$

where a_1, \dots, a_t are nonnegative exponents. It follows that

$$[L_{\mathfrak{p}} : L_{\mathfrak{p}} \cap \tilde{L}_{\mathfrak{p}}] = [\tilde{L}_{\mathfrak{p}} : L_{\mathfrak{p}} \cap \tilde{L}_{\mathfrak{p}}] = |\mathcal{O}/\mathfrak{p}|^{a_1 + \dots + a_t}.$$

We define $R(L : \mathfrak{p})$ to be the global graph containing lattices $\tilde{L} \in \text{gen}(L)$ such that $\tilde{L}_{\mathfrak{q}} = L_{\mathfrak{q}}$ at all prime spots $\mathfrak{q} \neq \mathfrak{p}$ as vertices. The distance $\text{dist}(L, \tilde{L}, \mathfrak{p})$ between \tilde{L} and L is $r = a_1 + \dots + a_t$. In particular, two vertices \tilde{L} and L are connected by an edge when $r = 1$ and they are called neighbors. It is known that the vertices of $R(L : \mathfrak{p})$ belong to at most two spinor genera, and two vertices belong to the same spinor genus when r is even. Therefore, all the vertices are in the same spinor genus $\text{spn}^+(L)$ if and only if the neighbor of L is in $\text{spn}^+(L)$.

Let $J_V = \{\Sigma \in \prod_{\mathfrak{q} \in \Omega} \mathcal{O}^+(V_{\mathfrak{q}}) : \|\Sigma_{\mathfrak{q}}\|_{\mathfrak{q}} = 1 \text{ for almost all } \mathfrak{q} \in \Omega\}$ be the group of split rotations and $J_V^L = \{\Sigma \in J_V : \Sigma L = L\}$ be the set of stabilizers of L . Let $\pi_{\mathfrak{p}}$ be a fixed prime element of the local field $F_{\mathfrak{p}}$. Define $\Sigma(\mathfrak{p}) \in J_V$ by setting $\Sigma(\mathfrak{p})_{\mathfrak{q}}$ to be the identity map for all primes $\mathfrak{q} \neq \mathfrak{p}$ and $\Sigma(\mathfrak{p})_{\mathfrak{p}} = \tau_{e_1 - f_1} \cdot \tau_{e_1 - \pi_{\mathfrak{p}} f_1}$, where τ_w denotes the symmetry with respect to the anisotropic

line $F_{\mathfrak{p}}w$. It is easy to check that the action of $\Sigma(\mathfrak{p})$ does not depend on the choice of $\pi_{\mathfrak{p}}$. Let $J_F = \{i \in \prod_{\mathfrak{q} \in \Omega} F_{\mathfrak{q}}^{\times} : |i_{\mathfrak{q}}|_{\mathfrak{q}} = 1 \text{ for almost all } \mathfrak{q} \in \Omega\}$. Define $j(\mathfrak{p}) \in J_F$ to have 1 at all primes $\mathfrak{q} \neq \mathfrak{p}$ and $\pi_{\mathfrak{p}}$ at prime \mathfrak{p} . Since $L_{\mathfrak{p}}$ is maximal, $\theta(O^+(L_{\mathfrak{p}}))$ contains all the units in $F_{\mathfrak{p}}$ [27, 91:8] so that $j(\mathfrak{p})$ is well-defined modulo $\theta(J_V^L)$. Moreover, $\theta(\Sigma(\mathfrak{p})) \equiv j(\mathfrak{p}) \pmod{\theta(J_V^L)}$.

Suppose $L' = \Sigma(\mathfrak{p})L$ is a neighbor of L ; then the graph $R(L : \mathfrak{p})$ contains only one spinor genus if and only if $L' \in \text{spn}^+(L)$ if and only if $j(\mathfrak{p}) \in P_D J_F^L$ ([27, 102:7]), where P_D is the subgroup of principal idèles generated by the elements in $D = \theta(O^+(V))$ which equals the set of elements in F^{\times} that are positive at all real spots \mathfrak{q} at which $V_{\mathfrak{q}}$ is anisotropic ([27, 101:8]), and $J_F^L = \{i \in J_F : i_{\mathfrak{q}} \in \theta(O^+(L_{\mathfrak{q}})) \text{ for all prime spots } \mathfrak{q}\}$.

Given a lattice \tilde{L} in the genus of L , based on the above observations, Benham and Hsia designed an algorithm to identify a prime spot \mathfrak{p} such that $L' = \Sigma(\mathfrak{p})L \in \text{spn}^+(\tilde{L})$ is a neighbor of L in the graph $R(L : \mathfrak{p})$. One can determine whether \tilde{L} is in $\text{spn}^+(L)$ by checking if $j(\mathfrak{p}) \in P_D J_F^L$.

Step 1: Compute X and T where X is the set of all real spots on F and T is a finite set of prime spots satisfying

1. $\mathfrak{q} \in T$ for all dyadic prime spots \mathfrak{q} ;
2. $L_{\mathfrak{q}}$ is unimodular at all prime spots $\mathfrak{q} \notin T$;
3. $L_{\mathfrak{q}} = \tilde{L}_{\mathfrak{q}}$ for all prime spots $\mathfrak{q} \notin T$.

Step 2: For each $\mathfrak{q} \in T$, compute an isometry $\Sigma_{\mathfrak{q}} \in O^+(V_{\mathfrak{q}})$ such that $\tilde{L}_{\mathfrak{q}} = \Sigma_{\mathfrak{q}}L_{\mathfrak{q}}$.

Step 3: For each $\mathfrak{q} \in T$, compute an element $x_{\mathfrak{q}} \in \mathcal{O}_{\mathfrak{q}} \cap \theta(\Sigma_{\mathfrak{q}}) \cdot F_{\mathfrak{q}}^{\times 2}$ and set $a_{\mathfrak{q}} = \text{ord}_{\mathfrak{q}}(x_{\mathfrak{q}}) + \text{ord}_{\mathfrak{q}}(4) + 1$.

Step 4: Compute an algebraic integer $c \in \mathcal{O}$ such that c is positive with respect to all $\mathfrak{q} \in X$ and c is congruent to $x_{\mathfrak{q}} \pmod{\mathfrak{q}^{a_{\mathfrak{q}}}}$ for each $\mathfrak{q} \in T$.

Step 5: Write the ideal $(c) = \prod_{\mathfrak{q} \in T} \mathfrak{q}^{k_{\mathfrak{q}}} \cdot \mathfrak{a}$ where \mathfrak{a} is relatively prime to each \mathfrak{q} in T and define a modulus $\mathfrak{m} = \prod_{\mathfrak{q} \in T} \mathfrak{q}^{a_{\mathfrak{q}}} \cdot \prod_{\mathfrak{q} \in X} \mathfrak{q}$. Given a fractional ideal $u = \mathfrak{b}c^{-1}$ where \mathfrak{b} and c are integral ideals, u is said to be relatively prime to \mathfrak{q} if both \mathfrak{b} and c are relatively prime to \mathfrak{q} . Let $I_F^{\mathfrak{m}} := \{\text{fractional ideals of } F \text{ that are relatively prime to each } \mathfrak{q} \in T\}$, $F_{\mathfrak{m},1} := \{a \in F^{\times} : a \equiv 1 \pmod{\mathfrak{m}}\}$ where $a \equiv 1 \pmod{\mathfrak{m}}$ means $\text{ord}_{\mathfrak{q}}(a - 1) \geq a_{\mathfrak{q}}$ for each $\mathfrak{q} \in T$ and $a > 0$ at each $\mathfrak{q} \in X$, and $S_{\mathfrak{m}} = \{a\mathcal{O} : a \in F_{\mathfrak{m},1}\}$. By a density theorem from class field theory, each ray class in the ray class group $I_F^{\mathfrak{m}}/S_{\mathfrak{m}}$ contains infinitely many primes. Compute a prime ideal \mathfrak{p} in the ray class $\mathfrak{a} \cdot S_{\mathfrak{m}}$. This \mathfrak{p} is the prime spot we are looking for.

Step 6: Determine if $j(\mathfrak{p}) \in P_D J_F^L$.

In the remaining of this section, we would like to restrict our attention to the number fields with class number 1, and study the complexity of this algorithm. Every quadratic lattice is free with a basis $\{v_1, \dots, v_n\}$ such that $L = \mathcal{O}v_1 + \dots + \mathcal{O}v_n$, and $dL = \det(b(v_i, v_j))$ is called the discriminant of L with respect

to the basis $\{v_1, \dots, v_n\}$. Moreover, if L is given in the form $\mathfrak{a}_1 w_1 + \dots + \mathfrak{a}_n w_n$ where $\mathfrak{a}_1, \dots, \mathfrak{a}_n$ are fractional ideals, then $\mathfrak{a}_1^2 \dots \mathfrak{a}_n^2 \det(b(w_i, w_j)) = dL\mathcal{O}$. We use the same symbol \mathfrak{q} to denote the prime ideal in both \mathcal{O} and $\mathcal{O}_{\mathfrak{q}}$, and use $\pi_{\mathfrak{q}}$ to denote their common generator.

Lemma 4. *Suppose $L_{\mathfrak{q}}$ and $\tilde{L}_{\mathfrak{q}}$ are maximal lattices, then there is an isometry $\sigma_{\mathfrak{q}}$ in $O^+(V_{\mathfrak{q}})$ sending $L_{\mathfrak{q}}$ to $\tilde{L}_{\mathfrak{q}}$ and $\theta(\sigma_{\mathfrak{q}}) = \pi_{\mathfrak{q}}^{\frac{1}{2}\text{ord}_{\mathfrak{q}}(d(L \cap \tilde{L})/dL)\mathcal{O}}$.*

Proof. Since $L_{\mathfrak{q}}$ and $\tilde{L}_{\mathfrak{q}}$ are maximal lattices, by [27, 91:2] there is a local basis $\{e_1, f_1, \dots, e_t, f_t, z_{2t+1}, \dots, z_n\}$ for $L_{\mathfrak{q}}$ satisfying $\phi(e_i) = \phi(f_i) = 0$, $b(e_i, f_j) = \frac{1}{2}\delta_{ij}$, $b(e_i, e_j) = b(f_i, f_j) = 0$ for $i \neq j$, $b(e_i, z_k) = b(f_i, z_k) = 0$, the subspace spanned by $\{z_{2t+1}, \dots, z_n\}$ is anisotropic, and $\tilde{L}_{\mathfrak{q}} = \mathfrak{q}^{a_1} e_1 + \mathfrak{q}^{-a_1} f_1 + \dots + \mathfrak{q}^{a_t} e_t + \mathfrak{q}^{-a_t} f_t + \mathcal{O}_{\mathfrak{q}} z_{2t+1} + \dots + \mathcal{O}_{\mathfrak{q}} z_n$. Define the isometry

$$\sigma_{\mathfrak{q}} = \tau_{e_1 - f_1} \tau_{e_1 - \pi_{\mathfrak{q}}^{a_1} f_1} \cdots \tau_{e_t - f_t} \tau_{e_t - \pi_{\mathfrak{q}}^{a_t} f_t},$$

then $\sigma_{\mathfrak{q}}(L_{\mathfrak{q}}) = \tilde{L}_{\mathfrak{q}}$ and $\theta(\sigma_{\mathfrak{q}}) = \pi_{\mathfrak{q}}^{a_1 + \dots + a_t}$. Note that

$$L_{\mathfrak{q}} \cap \tilde{L}_{\mathfrak{q}} = \mathfrak{q}^{a_1} e_1 + \mathcal{O}_{\mathfrak{q}} f_1 + \dots + \mathfrak{q}^{a_t} e_t + \mathcal{O}_{\mathfrak{q}} f_t + \mathcal{O}_{\mathfrak{q}} z_{2t+1} + \dots + \mathcal{O}_{\mathfrak{q}} z_n,$$

and $\mathfrak{q}^{2(a_1 + \dots + a_n)} = (d(L_{\mathfrak{q}} \cap \tilde{L}_{\mathfrak{q}})/dL_{\mathfrak{q}})\mathcal{O}_{\mathfrak{q}}$. Therefore

$$\theta(\sigma_{\mathfrak{q}}) = \pi_{\mathfrak{q}}^{\frac{1}{2}\text{ord}_{\mathfrak{q}}(d(L_{\mathfrak{q}} \cap \tilde{L}_{\mathfrak{q}})/dL_{\mathfrak{q}})\mathcal{O}_{\mathfrak{q}}} = \pi_{\mathfrak{q}}^{\frac{1}{2}\text{ord}_{\mathfrak{q}}(d(L \cap \tilde{L})/dL)\mathcal{O}}.$$

Lemma 5. *Suppose \tilde{L} is in the genus of L . Then $L_{\mathfrak{q}} = \tilde{L}_{\mathfrak{q}}$ if and only if $\text{ord}_{\mathfrak{q}}(d(L \cap \tilde{L})/dL)\mathcal{O} = 0$.*

Proof. Since both L and \tilde{L} are lattices on V , their intersection $L \cap \tilde{L}$ is also a lattice on V . Now, by the Invariant Factors Theorem (Theorem 4), there is a basis $\{v_1, \dots, v_n\}$ of V such that

$$L = \mathcal{O}v_1 + \dots + \mathcal{O}v_n \qquad L \cap \tilde{L} = \mathfrak{a}_1 v_1 + \dots + \mathfrak{a}_n v_n$$

with $\mathfrak{a}_1 \supseteq \dots \supseteq \mathfrak{a}_n$ integral ideals. Therefore, $L_{\mathfrak{q}} = \tilde{L}_{\mathfrak{q}}$ if and only if $L_{\mathfrak{q}} = L_{\mathfrak{q}} \cap \tilde{L}_{\mathfrak{q}}$ if and only if $\mathfrak{a}_1, \dots, \mathfrak{a}_n$ are $\mathcal{O}_{\mathfrak{q}}$ at \mathfrak{q} , i.e., $\text{ord}_{\mathfrak{q}}(d(L \cap \tilde{L})/dL)\mathcal{O} = 0$.

Lemma 6. *The finite set T of prime spots in Step 1 can be computed in (quantum) polynomial time.*

Proof. The set T in Step 1 consists of the prime ideals that appear in the factorization of $2dL\mathcal{O}$ or in the factorization of $d((L \cap \tilde{L})/dL)\mathcal{O}$. The basis of $L \cap \tilde{L}$ can be found using Hermite Normal Form in polynomial time, and the factorization of fractional ideals can be obtained in (quantum) polynomial time according to [18, Lemma 4.1].

Lemma 7. *The algebraic integer c in Step 4 can be computed in (quantum) polynomial time.*

Proof. Since \mathcal{O} is dense in the set $\mathcal{O}_{\mathfrak{q}}$, we can choose $x_{\mathfrak{q}}$ to be an element in \mathcal{O} . An algebraic integer c such that $c \equiv x_{\mathfrak{q}} \pmod{\mathfrak{q}^{a_{\mathfrak{q}}}}$ can be obtained in (quantum) polynomial time according to [18, Lemma 3.1, Lemma 3.5]. Also, let q_1, \dots, q_t be the rational prime numbers lying below the prime ideals $\mathfrak{q}_1, \dots, \mathfrak{q}_t$ in T . We can add a multiple of the product $q_1^{a_{\mathfrak{q}_1}} \dots q_t^{a_{\mathfrak{q}_t}}$ to c to satisfy the positive conditions at each real spot in X .

For Steps 2, 3 and 5, there are effective but possibly not efficient methods to compute the solutions. Let B_L and $B_{\tilde{L}}$ be the generating matrix of L and \tilde{L} respectively. Search for a matrix $M \in M_{n \times n}(F)$ with $\det M$ is a unit and whose entries are in $\mathcal{O}_{\mathfrak{q}}$ for each prime spot \mathfrak{q} dividing $2dL\mathcal{O}$ such that $B_L^t B_L = M^t (B_{\tilde{L}}^t B_{\tilde{L}}) M$. The existence of such a matrix M can be found in [27, Example 102:4], and we can replace L by the lattice generated by $B_{\tilde{L}} M$ which is in the proper class of L .

Then in Steps 2 and 3, for all primes spots \mathfrak{q} dividing $2dL\mathcal{O}$, since $L_{\mathfrak{q}} = \tilde{L}_{\mathfrak{q}}$, we can choose $\Sigma_{\mathfrak{q}}$ to be the identity map and $\theta(\Sigma_{\mathfrak{q}}) = 1$. For the remaining prime spots $\mathfrak{q} \in T$, both $L_{\mathfrak{q}}$ and $\tilde{L}_{\mathfrak{q}}$ are maximal and $\Sigma_{\mathfrak{q}}$ and its spinor norm can be found in Lemma 4.

For Step 5, one can find a prime ideal \mathfrak{p} in the ray class of \mathfrak{a} by searching through all (finitely many) prime ideals with norm bounded by the constant given in [31] that can be effectively calculated.

6 Spinor Genera of Binary Forms over Number Fields

Many results given above, which appear to frustrate algebraic approaches to solving LIP via computing spinor genera, have the condition $n \geq 3$. However, HAWK uses rank 2 forms, integral over a cyclotomic field. In this section we ask: how does the spinor norm behave in this setting? In Sect. 7.1 we explain in more detail the connection between our results and the HAWK signature scheme.

Our result relies on the work of Earnest and Estes, who prove in [16] (via [17]) that in the binary setting, two lattices in the same genus lie in the same spinor genus if and only if they are fourth powers (quartic residues) in the class group of some order in the field, when the ring of integers is a PID. Thus if one can compute quartic residues in class groups of non-maximal orders, one could correctly decide the answer to the distinguish LIP problem, when the forms lie in different spinor genera.

There are quantum algorithms to compute the class group of a suborder of any number field efficiently, under GRH [6]. Moreover, deciding quadratic residuosity in the class group can be performed efficiently, given the data from those quantum algorithms; and the same is true for quartic residuosity. The quantum algorithm we will need is:

Theorem 11. [6, Theorem 1.2] (*Class group Computation*) *Under the Generalized Riemann Hypothesis, there is a quantum algorithm for computing the class group of an order \mathcal{O} in a number field K which runs in polynomial time in the parameters $n = \deg(K)$ and $\log(|\Delta|)$, where Δ is the discriminant of \mathcal{O} .*

We note that the above algorithm ‘computes the class group’ by computing a generating set of prime ideals together with the relations between them.

We now explain in more detail the result of Earnest and Estes. In the following, we consider regular binary quadratic spaces (V, ϕ) over a number field F and anisotropic binary quadratic forms over the ring of integers \mathcal{O}_F . These correspond to lattices of rank 2 over that ring of integers, contained in V . We may fix a basis such that this vector space is in fact isomorphic to a field extension of degree 2, when (V, ϕ) is anisotropic. We then have $V \cong F(\sqrt{-d})$ for some d , and we write \mathcal{O}_V for the ring of integers of $F(\sqrt{-d})$. There is then an involution $*$ on V fixing F such that $\phi(x) = xx^*$ for any $x \in V$. For more details see [17].

The following two results combine to imply that two lattices $L_1, L_2 \subset V$ in the same genus are in the same proper spinor genus if and only if $L_1L_2^{-1}$ is a quartic residue in the class group of the left order of L_2 in V . Recall that the left order is defined as $\mathcal{O}_l(L_2) := \{x \in V : xL_2 \subset L_2\} \subset V$, and any lattice is a left ideal in its left order.

Proposition 2. [17, Proposition 2.3] *A necessary and sufficient condition that L_1 be in $\text{cls}^+(L_2)$ (resp. $\text{spn}^+(L_2)$ or $\text{gen}(L_2)$) is that $L_1L_2^{-1}$ be in $\text{cls}^+(\mathcal{O}_l(L_2))$ (resp. $\text{spn}^+(\mathcal{O}_l(L_2))$ or $\text{gen}(\mathcal{O}_l(L_2))$).*

For any (possibly non-maximal) \mathcal{O}_F -order $\mathcal{O} \subset V$, denote the group of invertible fractional ideals of \mathcal{O} by $\mathcal{I}(\mathcal{O})$, and the subgroup of principal invertible fractional ideals by $\mathcal{P}(\mathcal{O})$. Set

$$\mathcal{H}(\mathcal{O}) = \text{gen}(\mathcal{O}) / \text{spn}^+(\mathcal{O}),$$

and

$$\mathcal{C}(\mathcal{O}) = \mathcal{I}(\mathcal{O}) / \mathcal{P}(\mathcal{O}).$$

Then

Corollary 1. [16, §4] *Suppose F is a number field and \mathcal{O}_F is a PID. Let \mathcal{O} be a degree 2 order over \mathcal{O}_F . Then we have $\mathcal{H}(\mathcal{O}) \cong \mathcal{C}(\mathcal{O})^2 / \mathcal{C}(\mathcal{O})^4$.*

A consequence of this is that we find $\text{spn}^+(\mathcal{O}) / \text{cls}^+(\mathcal{O}) \cong \mathcal{C}(\mathcal{O})^4$. Thus we can prove

Theorem 12. *Let F be a number field and suppose \mathcal{O}_F is a PID. Let f and g be two anisotropic binary quadratic forms, integral over \mathcal{O}_F , lying in the same genus. Let $V (= F \cdot L_f = F \cdot L_g)$ be the rank 2 quadratic space containing L_f and L_g . Then if $L_f \cdot L_g^{-1}$ generates an ideal coprime to the conductor of $\mathcal{O}_l(L_g)$ in \mathcal{O}_V , there is a quantum polynomial time algorithm to decide if $f \in \text{spn}^+(g)$.*

Proof. Let the corresponding lattices to f, g be denoted by L_f, L_g . Since f, g are anisotropic, V is anisotropic and hence isomorphic to a quadratic field extension of F ; identify V with this extension. Begin by computing a basis of the left order of L_g in V , $\mathcal{O}_l(L_g)$. Next, use Theorem 11 to compute the class group structure, obtaining a generating set of prime ideals in the class group of $\mathcal{O}_l(L_g)$

in quantum polynomial time, together with their defining relations. This system of relations forms a lattice Λ , and we obtain an isomorphism $\mathcal{C}(\mathcal{O}_l(L_g)) \rightarrow \mathbb{Z}^n/\Lambda$ by writing an element of $\mathcal{C}(\mathcal{O}_l(L_g))$ as a product of powers of prime ideals from our generating set, and mapping to the vector of exponents (modulo the lattice of relations). That is, for $I \in \mathcal{C}(\mathcal{O}_l(L_g))$, we write $I = \prod_{i=1}^n \mathfrak{p}_i^{e_i}$ and then send $I \mapsto (e_1, \dots, e_n) + \Lambda$.

Since \mathbb{Z}^n/Λ is an abelian group, we can then consider $\mathcal{C}(\mathcal{O}_l(L_g)) \cong \oplus_i \mathbb{Z}/d_i\mathbb{Z}$, and the image of $L_f \cdot L_g^{-1}$ in $\oplus_i \mathbb{Z}/d_i\mathbb{Z}$ for some integers d_i , where the factors d_i are obtained by the algorithm of Theorem 11. Moreover, the algorithm outputs a list of vectors \bar{g}_i of order d_i which form a basis of $\mathbb{Z}^n/\Lambda \cong \oplus_i \mathbb{Z}/d_i\mathbb{Z}$ [9, §6.5.4].

If $\mathcal{O}_l(L_g)$ is a maximal order or more generally if $L_f \cdot L_g^{-1}$ generates an ideal coprime to the conductor of $\mathcal{O}_l(L_g)$ in \mathcal{O}_V , this can be done by factorising $L_f \cdot L_g^{-1}$ into a product of prime ideals contained in our generating set, and then reducing modulo the relations between the prime ideals in the class group obtained by the algorithm of Theorem 11. We then map $L_f \cdot L_g^{-1} \mapsto (f_1, \dots, f_n) + \Lambda$ for some exponents f_i .

Testing such an element for quartic residuosity can then be done efficiently as follows: we may take the basis $\bar{g}_1, \dots, \bar{g}_n$ and express $(f_1, \dots, f_n) = \sum_i \lambda_i \bar{g}_i$ for some coefficients $\lambda_i \in \mathbb{Z}/d_i\mathbb{Z}$, $i = 1, \dots, n$; thus $f_j = \sum_i \lambda_i \bar{g}_{ij}$. We then express the above as a matrix-vector equation: $(f_1, \dots, f_n)^T = G \cdot \lambda$ where G is the matrix with i th column \bar{g}_i^T and λ is a vector with i th entry λ_i . We then compute $G^{-1} \cdot (f_1, \dots, f_n)^T = \lambda$; if $\lambda_i = 4\gamma_i \pmod{d_i}$ for some $\gamma_i \in \mathbb{Z}/d_i\mathbb{Z}$ and for all $i = 1, \dots, n$, we conclude that $L_f \cdot L_g^{-1}$ is a quartic residue in the class group.

Finally, if $L_f \cdot L_g^{-1}$ is a quartic residue in $\mathcal{C}(\mathcal{O}_l(L_g))$, then $f \in \text{spn}^+(g)$ by Corollary 1; otherwise, $f \notin \text{spn}^+(g)$.

Algorithm 3: Quantum Algorithm for Spinor Genus of Binary Forms

Input: Quadratic forms f and g in the same genus

Output: Answer

- 1: Compute basis of $\mathcal{O}_l(L_g)$
 - 2: $(\{\mathfrak{p}_1, \dots, \mathfrak{p}_n\}, \Lambda, \{d_1, \dots, d_n\}, \{\bar{g}_1, \dots, \bar{g}_n\}) \leftarrow$ Algorithm of Theorem 11
 - 3: $L_f \cdot L_g^{-1} = \prod \mathfrak{p}_i^{f_i}$
 - 4: $(f'_1, \dots, f'_n) := (f_1, \dots, f_n) \pmod{\Lambda}$
 - 5: Compute $G^{-1} \cdot (f'_1, \dots, f'_n)^T$
 - 6: **if** $G^{-1} \cdot (f'_1, \dots, f'_n)^T = \mathbf{0} \pmod{4}$ **then**
 - 7: Output ‘Yes’
 - 8: **else**
 - 9: Output ‘No’
 - 10: **end if**
-

We make the following remark:

Corollary 2. *Let F be a number field and suppose \mathcal{O}_F is a PID. Let f and g be two anisotropic binary quadratic forms, integral over \mathcal{O}_F , in the same genus. Let*

V be the rank 2 quadratic space containing L_f and L_g . Suppose $L_f \cdot L_g^{-1}$ generate an ideal coprime to the conductor of $\mathcal{O}_l(L_g)$ in \mathcal{O}_V , and $\gcd(|\mathcal{C}(\mathcal{O}_l(L_g))|, 2) = 1$. Then $f \in \text{spn}^+(g)$.

Proof. When $|\mathcal{C}(\mathcal{O}_l(L_g))|$ is odd, then none of the d_i obtained in the course of the algorithm implicit in the proof of Theorem 6 are even. Then in the penultimate paragraph of the proof, when one computes $G^{-1} \cdot (f_1, \dots, f_n)^T = \lambda$, and checks if $\lambda_i = 4\gamma_i \pmod{d_i}$ for some $\gamma_i \in \mathbb{Z}/d_i\mathbb{Z}$ and for all $i = 1, \dots, n$, we must find that there always exist such $\gamma_i \pmod{d_i}$, since $\gcd(4, d_i) = 1$. Thus in this setting the two forms f, g always lie in the same spinor genus.

We note that there are many examples of number fields with odd class numbers which may be considered relevant to LIP in cryptography: for instance, the power-of-two cyclotomic fields $\mathbb{Q}(\zeta_{64})$, $\mathbb{Q}(\zeta_{128})$, and $\mathbb{Q}(\zeta_{256})$ all have odd class number, being 17, 359057, and 10449592865393414737 respectively (see [22, 25, 29]). However, many cyclotomic fields have even class number, such as $\mathbb{Q}(\zeta_{130})$, which has class number 64. We conclude from this that if one is choosing parameters for LIP-based schemes over number fields, one must choose the number field carefully to avoid distinguishing attacks as detailed in the section below.

We also derive two corollaries regarding cyclotomic fields:

Corollary 3. *Let $F = \mathbb{Q}(\zeta_n)$ be a cyclotomic field and*

$$n \in \{1, 3, 4, 5, 7, 8, 9, 11, 12, 13, 15, 16, 17, 19, 20, 21, 24, 25, 27, 28, 32, 33, 35, 36, 40, 44, 45, 48, 60, 84\}$$

Let f and g be two anisotropic binary quadratic forms, integral over \mathcal{O}_F , in the same genus. Let V be the rank 2 quadratic space containing L_f and L_g . Then if $L_f \cdot L_g^{-1}$ generates an ideal coprime to the conductor of $\mathcal{O}_l(L_g)$ in \mathcal{O}_V , there is a quantum polynomial time algorithm to decide if $f \in \text{spn}^+(g)$.

Proof. [29, Theorem 11.1] states that the cyclotomic fields with n as in the corollary statement are all the cyclotomic fields with class number equal to one. We may then apply the theorem for binary integral anisotropic quadratic forms over such rings of integers.

Corollary 4. *Let F be the maximal totally real subfield of $\mathbb{Q}(\zeta_n)$ and $n \in S := \{4, 8, 16, 32, 64, 128, 256\}$ (and assuming GRH, $n \in S \cup \{512\}$). Then there is a quantum polynomial time algorithm to decide if $f \in \text{spn}^+(g)$.*

Proof. [25, Theorem 2.1] states that the cyclotomic fields with n as in the corollary statement are all the cyclotomic fields for which we know unconditionally (and conditionally for $n = 512$) that the maximal real subfield has class number equal to one. We may then apply the theorem for binary integral anisotropic quadratic forms over such rings of integers.

7 Application to Distinguish LIP

We now apply the theory of the previous sections to the lattice isomorphism problem. We begin by defining these problems. Denote the real orthogonal group in n dimensions by $O_n(\mathbb{R})$.

Definition 5. (search LIP, lattices) Given two isometric lattices $L_1, L_2 \subset \mathbb{R}^n$, find an orthogonal transformation $O \in O_n(\mathbb{R})$ such that $L_2 = O \cdot L_1$.

We redefine this in terms of quadratic forms:

Definition 6. (search LIP, quadratic forms) Given two positive definite integral quadratic forms Q_1, Q_2 in the same equivalence class, find a unimodular $U \in GL_n(\mathbb{Z})$ such that $Q_2 = U^t Q_1 U$.

There is a distinguishing variant of this problem:

Definition 7. (distinguish LIP, quadratic forms) Given two positive definite integral quadratic forms Q_0, Q_1 , the distinguish LIP problem Δ -LIP is, given any quadratic form $Q' \in [Q_b]$ for $b \in \{0, 1\}$ a uniform random bit, to find b .

And a decision variant:

Definition 8. Given positive definite integral quadratic form Q , the decision LIP problem $dLIP^Q$ is, given any Q' , to decide if $Q' \in [Q]$ or not.

As discussed in [15], for Δ -LIP to be hard Q_0 and Q_1 must be equivalent over $\mathbb{Q}, \mathbb{R}, \mathbb{Q}_p$, and \mathbb{Z}_p for all p , as well as the forms to agree on any other computable invariant. However, that paper did not discuss the spinor genus of the forms; we fill in that gap in this section.

We outline the immediate consequence of Sect. 6 for LIP over number fields. Suppose in the Δ -LIP experiment, Q_0 and Q_1 are integral binary quadratic forms over the ring of integers of a number field which is a PID, lying in the same genus (and that the implicit quadratic space is anisotropic). Then suppose we are given Q' which lies in either $[Q_0]$ or $[Q_1]$. We run the algorithm implicit in the proof of Theorem 12 on the pairs $(Q', Q_0), (Q', Q_1)$. If the spinor genus has not been accounted for and the forms Q_0, Q_1 lie in different spinor genera within the same genus, we may answer Δ -LIP correctly in polynomial time by ruling out the form lying in the wrong spinor genus, since Q' lies in the same spinor genus as Q_b .

Similarly, in the $dLIP^Q$ experiment, if Q' and Q lie in the same genus but in distinct spinor genera, in the event that the forms not only lie in distinct equivalence classes but also distinct spinor genera, we may detect this and correctly answer ‘No’.

7.1 Implications for the Schemes of [15] and [14]

In [15], the authors gave a KEM and a signature scheme, both having their hardness founded on distinguish LIP. These schemes were designed for integral forms

of rank $n (\gg 5)$ and we conclude from the analysis of Sect. 3.3 that, if a ‘random’ quadratic form of determinant p^m has its Jordan p -symbol uniformly distributed among possible Jordan p -symbols, then with only negligible probability do two such forms lie in a genus which splits into multiple spinor genera.

However, our work does have consequences for structured cases of these LIP instances. Moving from forms over the rational integers of rank $n = 2m$ to binary quadratic forms over the ring of integers of a number field of degree m yields forms of the same overall rank, yet would introduce structure into the LIP instances which makes them vulnerable to our Theorem 12. We thus caution against the use of such structured LIP instances in cryptography.

In HAWK [14], the authors gave a signature scheme, which was later submitted to the first round of NIST’s additional post-quantum standardisation process for digital signatures. This scheme was designed for rank two forms over the ring of integers of cyclotomic fields of power-of-two conductor. These correspond to rank two modules over such rings. At first sight, it might seem that our result affects the security of HAWK. However, firstly, these rings of integers are not PIDs, so our theorem does not apply; secondly, our result applies to distinguish LIP, whereas the security of HAWK is based on a search problem; and thirdly, HAWK is based on Hermitian forms, which we do not consider. To explain this last point, we give a brief overview of HAWK which illuminates the differences to the notions studied in this work, following the notation of [26].

Let K be an algebraic number field of degree n . Then there are n embeddings $\sigma_i : K \hookrightarrow \mathbb{C}$. We define the canonical embedding of K as

$$\sigma_K : K \rightarrow \mathbb{R}^{r_1} \times \mathbb{C}^{2r_2}$$

defined by

$$x \mapsto (\sigma_1(x), \dots, \sigma_n(x))$$

Here $\text{im } \sigma_K \subset H := \{(x_1, \dots, x_n) \in \mathbb{R}^{r_1} \times \mathbb{C}^{2r_2} : x_{r_1+r_2+j} = \overline{x_{r_1+j}}, 1 \leq j \leq r_2\}$. This map is extended to vectors over K componentwise. For every module $\mathcal{M} \subset K^\ell$ of finite rank over a Dedekind domain $R \subset K$, there exist ideals I_k of R and linearly independent vectors b_k of K^ℓ such that $\mathcal{M} = \sum_{k=1}^m I_k \cdot b_k$. Then $[(I_k)_k, (b_k)_k]$ is a *pseudo-basis* of \mathcal{M} . Write $K_{\mathbb{R}} = K \otimes \mathbb{R}$. Then a *module lattice* in $\sigma_K(K_{\mathbb{R}})^\ell$ for some $\ell > 0$ is given by the embedding of \mathcal{M} under σ_K .

Let $U_n(K_{\mathbb{R}})$ denote the $n \times n$ unitary matrices with entries in $K_{\mathbb{R}}$, that is, matrices $A \in M_n(K_{\mathbb{R}})$ such that $A^{-1} = \bar{A}^T$, where $\bar{\cdot}$ is complex conjugation. We then say that two module lattices $\mathcal{M}_1, \mathcal{M}_2 \subset \sigma_K(K_{\mathbb{R}})^\ell$ are isomorphic if there exists $U \in U_\ell(K_{\mathbb{R}})$ such that $\mathcal{M}_2 = U\mathcal{M}_1$. The module LIP problem is, given two such module lattices, to find such a U .

There is a corresponding notion of quadratic forms; these are Hermitian forms, defined as follows. A matrix A in $M_n(K_{\mathbb{R}})$ is a Hermitian form if $\bar{A}^T = A$. Such a form is positive definite when $\phi_A(x) = \bar{x}^T A x > 0$ for all $x \in K_{\mathbb{R}}^n$ with entries which do not embed to 0.

We call K totally real if the image of every embedding lies properly in \mathbb{R} . Then $U_m(K_{\mathbb{R}})$ is the set of matrices A such that $A^{-1} = A^T$, since complex

conjugation acts trivially. Then isomorphism of module lattices corresponds to our above definitions of equivalence of lattices, and our results may apply to such problems. However, this is the very setting in which [26] solved the binary module LIP problem.

In the case when K is not totally real, we use a different notion of equivalence of lattices to HAWK (since we do not define equivalence by the *conjugate* transpose above). Thus our results do not affect HAWK. However, we leave it as an open problem to see if the spinor genus can be efficiently computed for integral binary Hermitian forms over number fields.

Acknowledgements. This work was supported in part by the Engineering and Physical Sciences Research Council (EPSRC) [grant numbers EP/X037010/1 and EP/Y037243/1]. The authors would like to thank Dr. Fuchun Lin for helpful discussions. The second author was also partially supported by the Texas A&M University-San Antonio 2024 Research Council Grant, and extends her thanks to the Department of Electrical and Electronic Engineering at Imperial College London for their hospitality and financial support during her summer visits.

References

1. M. Ajtai. “Generating Hard Instances of Lattice Problems”. In: *Electron. Colloquium Comput. Complex.* TR96 (1996). DOI: <https://doi.org/10.1145/237814.237838>.
2. M. Ajtai and C. Dwork. “A Public-Key Cryptosystem with Worst-Case/Average-Case Equivalence”. In: *STOC 97*. Association for Computing Machinery, 1997, 284–293. DOI: <https://doi.org/10.1145/258533.258604>.
3. G.E. Andrews and K. Eriksson. *Integer Partitions*. Cambridge University Press, 2004. ISBN: 9780521600903.
4. J.W. Benham and J.S. Hsia. “Spinor equivalence of quadratic forms”. In: *Journal of Number Theory* 17.3 (1983), pp. 337–342. DOI: [https://doi.org/10.1016/0022-314X\(83\)90051-3](https://doi.org/10.1016/0022-314X(83)90051-3).
5. H. Bennett, D. Dadush, and N. Stephens-Davidowitz. “On the Lattice Distortion Problem”. In: *ESA 2016*. Ed. by P. Sankowski and C. Zaroliagis. Vol. 57. LIPIcs. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2016. DOI: <https://doi.org/10.4230/LIPIcs.ESA.2016.9>.
6. J.-F. Biasse and F. Song. “Efficient quantum algorithms for computing class groups and solving the principal ideal problem in arbitrary degree number fields”. In: *SODA ’16*. Society for Industrial and Applied Mathematics, 2016, 893–902.
7. P. Bruin, L. Ducas, and S. Gibbons. “Genus distribution of random q -ary lattices”. In: *NuTMiC 2021*. Vol. 126. Banach Center Publications, Jan. 2023, pp. 137–159. DOI: <https://doi.org/10.4064/bc126-9>.
8. J.W.S. Cassels. *Rational Quadratic Forms*. Dover Books on Mathematics. Dover Publications, 2008. ISBN: 9780486466705.
9. H. Cohen. *A Course in Computational Algebraic Number Theory*. Graduate Texts in Mathematics. Springer Berlin Heidelberg, 2013. ISBN: 9783662029459.
10. H. Cohen, G. Frey, R. M. Avanzi, C. Doche, T. Lange, K. Nguyen, and F. Vercauteren. “Handbook of Elliptic and Hyperelliptic Curve Cryptography”. In: ed. by K. Rosen. *Discrete Mathematics and its Applications*. Chapman and Hall/CRC, 2005.

11. School of Mathematics Computational Algebra Group and University of Sydney Statistics. *Genera and Spinor Genera. MAGMA documentation*. Accessed 24/05/2024. URL: <https://magma.maths.usyd.edu.au/magma/handbook/text/339>.
12. J. Conway and N. Sloane. *Sphere Packings, Lattices and Groups*. Vol. 290. Jan. 1988. ISBN: 978-1-4757-2018-1. DOI: <https://doi.org/10.1007/978-1-4757-2016-7>.
13. L. Ducas and S. Gibbons. “Hull Attacks on the Lattice Isomorphism Problem”. In: *PKC 2023*. Ed. by A. Boldyreva and V. Kolesnikov. Vol. 13940. LNCS. Springer Nature Switzerland, 2023, pp. 177–204. DOI: https://doi.org/10.1007/978-3-031-31368-4_7.
14. L. Ducas, E. W. Postlethwaite, L. N. Pulles, and W. van Woerden. “Hawk: Module LIP Makes Lattice Signatures Fast, Compact And Simple”. In: *ASIACRYPT 2022*. Vol. 13794. LNCS. Springer-Verlag, 2023, 65–94. DOI: https://doi.org/10.1007/978-3-031-22972-5_3.
15. L. Ducas and W. van Woerden. “On the Lattice Isomorphism Problem, Quadratic Forms, Remarkable Lattices, and Cryptography”. In: *EUROCRYPT 2022*. Ed. by O. Dunkelman and S. Dziembowski. Vol. 13277. LNCS. Springer International Publishing, pp. 643–673. DOI: https://doi.org/10.1007/978-3-031-07082-2_23.
16. A. G. Earnest and D. R. Estes. “An algebraic approach to the growth of class numbers of binary quadratic lattices”. In: *Mathematika* 28.2 (1981), pp. 160–168. DOI: <https://doi.org/10.1112/S0025579300010214>.
17. A. G. Earnest and D. R. Estes. “Class Groups in the Genus and Spinor Genus of Binary Quadratic Lattices”. In: *Proceedings of the London Mathematical Society* s3-40.1 (1980), pp. 40–52. DOI: <https://doi.org/10.1112/plms/s3-40.1.40>.
18. K. Eisenträger and S. Hallgren. “Algorithms for ray class groups and Hilbert class fields”. In: *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms* (2010), 471–483.
19. D. R. Estes and G. Pall. “Spinor genera of binary quadratic forms”. In: *Journal of Number Theory* 5.6 (1973), pp. 421–432. DOI: [https://doi.org/10.1016/0022-314X\(73\)90012-7](https://doi.org/10.1016/0022-314X(73)90012-7).
20. I. Haviv and O. Regev. “On the Lattice Isomorphism Problem”. In: *SODA 2014*, pp. 391–404. DOI: <https://doi.org/10.1137/1.9781611973402.29>.
21. J. Hoffstein, J. Pipher, and J. Silverman. “NTRU: A Ring-Based Public Key Cryptosystem”. In: *ANTS 1998*. Vol. 1423. LNCS. Springer, 1998, 267–288.
22. OEIS Foundation Inc. *Relative class number h^- of cyclotomic field $\mathbb{Q}(\zeta_n)$* . Entry A061653 in *The On-Line Encyclopedia of Integer Sequences*. Accessed 21/05/2024. URL: <https://oeis.org/A061653>.
23. K. Jiang, A. Wang, H. Luo, G. Liu, Y. Yu, and X. Wang. “Exploiting the Symmetry of \mathbb{Z}^n : Randomization and the Automorphism Problem”. In: *ASIACRYPT 2023*. Ed. by J. Guo and R. Steinfeld. Vol. 14441. LNCS. Springer Nature Singapore, 2023, pp. 167–200. DOI: https://doi.org/10.1007/978-981-99-8730-6_6.
24. D. Marcus. *Number Fields*. Universitext. Springer-Verlag, 1977. ISBN: 9783319902326.
25. J. Miller. “Class numbers of totally real fields and applications to the Weber class number problem”. In: *Acta Arithmetica* 164 (May 2014). DOI: <https://doi.org/10.4064/aa164-4-4>.
26. G. Mureau, A. Pellet-Mary, G. Pliatsok, and A. Wallet. “Cryptanalysis of Rank-2 Module-LIP in Totally Real Number Fields”. In: *EUROCRYPT 2024*. Ed. by M. Joye and G. Leander. Vol. 14657. LNCS. Springer Nature Switzerland, 2024, pp. 226–255.

27. O.T. O'Meara. *Introduction to Quadratic Forms*. Grundlehren der mathematischen Wissenschaften. Springer Berlin Heidelberg, 2013. ISBN: 9783662419229.
28. O. Regev. "On lattices, learning with errors, random linear codes, and cryptography". In: *J. of the ACM* 56 (6 2009). DOI: <https://doi.org/10.1145/1568318>.
29. L. C. Washington. *Introduction to Cyclotomic Fields*. Graduate Texts in Mathematics. Springer New York, 2012. ISBN: 9781461219347.
30. G.L. Watson. *Integral Quadratic Forms*. Cambridge University Press, 1960. ISBN: 9780521091817.
31. A. R. Weiss. "The least prime ideal". In: *J. Reine Angew. Math.* 338 (1983), 56–94.



Cryptanalysis of Rank-2 Module-LIP with Symplectic Automorphisms

Hengyi Luo^{1,2}, Kaijie Jiang³, Yanbin Pan^{1,2(✉)}, and Anyu Wang^{3,4}

¹ Key Laboratory of Mathematics Mechanization, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing, China

luohengyi23@mails.ucas.ac.cn

² School of Mathematical Sciences, University of Chinese Academy of Sciences, Beijing, China

panyanbin@amss.ac.cn

³ Institute for Advanced Study, BNRist, Tsinghua University, Beijing, China

jkj21@mails.tsinghua.edu.cn, anyuwang@tsinghua.edu.cn

⁴ Zhongguancun Laboratory, Beijing, China

Abstract. At Eurocrypt'24, Mureau et al. formally defined the Lattice Isomorphism Problem for module lattices (module-LIP) in a number field \mathbb{K} , and proposed a heuristic randomized algorithm solving module-LIP for modules of rank 2 in \mathbb{K}^2 with a totally real number field \mathbb{K} , which runs in classical polynomial time for a large class of modules and a large class of totally real number field under some reasonable number theoretic assumptions. In this paper, by introducing a (pseudo) symplectic automorphism of the module, we successfully reduce the problem of solving module-LIP over CM number field to the problem of finding certain symplectic automorphism. Furthermore, we show that a weak (pseudo) symplectic automorphism can be computed efficiently, which immediately turns out to be the desired automorphism when the module is in a totally real number field. This directly results in a provable deterministic polynomial-time algorithm solving module-LIP for rank-2 modules in \mathbb{K}^2 where \mathbb{K} is a totally real number field, without any assumptions or restrictions on the modules and the totally real number fields. Moreover, the weak symplectic automorphism can also be utilized to invalidate the omSVP assumption employed in HAWK's forgery security analysis, although it does not yield any actual attacks against HAWK itself.

Keywords: Lattice automorphism · module-LIP · Symplectic matrix

1 Introduction

Lattices are discrete additive subgroups of \mathbb{R}^m , which provide rich geometric structures that can be used to define various computationally hard problems, such as the famous shortest vector problem (SVP) and the closest vector problem (CVP). Based on the hardness of these problems or their variants, lots of lattice-based cryptosystems have been constructed. It is widely believed that lattice-based cryptosystems are quantum-resistant, and some of them are selected as the

standard algorithms in NIST’s Post-Quantum Cryptography Standardization Project.

Lattice Isomorphism Problem (LIP) is another lattice-related computational problem. Two lattices \mathcal{L}_1 and \mathcal{L}_2 are said to be isomorphic if there exists a bijective orthogonal transformation from \mathcal{L}_1 to \mathcal{L}_2 . The search version of LIP refers to the question of finding such orthogonal transformation given the lattice bases of \mathcal{L}_1 and \mathcal{L}_2 , and the decision version asks to determine whether the two given lattices are isomorphic or not. Research on LIP dates back to [22] in the 1990s, in which the LIP for low-dimensional lattices was considered. In [14], Haviv and Regev proposed an $n^{O(n)}$ -time algorithm for solving the general LIP, which remains the fastest known algorithm for LIP. Since then, many more cryptanalytic works have been proposed [3, 6–10, 12, 13, 17, 18, 23].

Most of these works focus on a special case of LIP, namely, \mathbb{Z} LIP, in which \mathcal{L} is the hypercubic lattice \mathbb{Z}^n , such as [4, 12]. Recently, Ducas [6] explored a reduction from n -dimensional \mathbb{Z} LIP to $\frac{n}{2}$ -dimensional SVP, which means that \mathbb{Z} LIP can be solved with $2^{n/2}$ time complexity due to the best provable algorithm [1] for SVP. A similar algorithmic result can be concluded by employing Bennett et al.’s reduction [3] from \mathbb{Z} SVP to $O(1)$ -uSVP with the well-known reduction from \mathbb{Z} LIP to \mathbb{Z} SVP.

To improve the efficiency of LIP-based cryptosystems, an algebraic variant of LIP, called module-LIP problem, was introduced by Ducas et al. [8], where the module can be chosen as free module over a CM number field instead of just the ring of integers. By taking the module M as $\mathcal{O}_{\mathbb{L}}^2$ where the field \mathbb{L} is a cyclotomic field with conductor being a power of 2, Ducas et al. [8] presented a signature scheme called HAWK, whose security relies on the hardness of $\mathcal{O}_{\mathbb{L}}^2$ -LIP problem. HAWK is now a candidate algorithms in the first round of NIST’s Post-Quantum Cryptography Standardization Project for additional digital signature proposals. However, in spite of the additional algebraic structure, we always treat $\mathcal{O}_{\mathbb{L}}^2$ -LIP problem as an LIP on non-structured lattices when analyzing the security of HAWK. Hence, a natural problem is how to solve module-LIP more efficiently than LIP with its special algebraic structure.

For LIP with algebraic structures, the most well-known algorithm originates from the work of Gentry and Szydlo [13], which tries to recover the secret key of NTRUSign [15] by solving some special \mathbb{Z} LIP instance with algebraic structure. Later, Lenstra and Silverberg made [17, 18] lots of in-depth analysis of the Gentry-Szydlo algorithm, and Lenstra and Silverberg [19] generalized it to check isomorphism of lattices over CM-orders. The essence of these algorithms lies in using sufficient lattice automorphisms to solve LIP. Note that the automorphisms provided by the algebraic structure of rank-1 module enable the Gentry-Szydlo algorithm and its variants to solve the corresponding rank-1 module-LIP problems over CM number fields.

At Eurocrypt’24, Mureau et al. [20] formalized the framework for module-LIP using the concept of pseudo-bases, allowing it to be defined on module lattices over general number fields. Furthermore, as their main technical contribution, they presented a heuristic algorithm for solving module-LIP when the module

$M \subset \mathbb{K}^2$ has rank 2 and when the number field \mathbb{K} is a totally real number field. Roughly speaking, the strategy in [20] mainly utilizes the prime decomposition of the principal ideal generated by the sum of squares $x^2 + y^2$ to guess the principal ideal generated by its factor $x + i \cdot y$. To avoid factoring a general integer during the process of prime ideal factorization, which is still hard on classical computers by now, the ideals should be selected carefully such that their norms are easy to be factored under some heuristic assumption. However, guessing the desired principal ideal by enumerating the possible combinations of prime ideals, still makes the time complexity of the final algorithm exponential in the number of distinct prime ideals factors (Theorem 4.6 in [20]). Therefore, the algorithm in [20] runs in polynomial time under some reasonable heuristic assumptions for a class of certain module-LIP, which relates to the arithmetic properties of the module and the field. It should be noted that their algorithm does not impact the security of HAWK, as pointed out in [20].

1.1 Our Contributions

In this paper, we present a provable deterministic polynomial-time algorithm to solve $\mathcal{O}_{\mathbb{L}}^2$ -LIP where \mathbb{L} is a CM number field, with the help of a new module lattice automorphism defined by a symplectic matrix with rank 2. Therefore, we reduce the problem of solving $\mathcal{O}_{\mathbb{L}}^2$ -LIP to the problem of finding out the certain module lattice symplectic automorphism. Although it seems not easy to find the exact symplectic automorphism in general, we can compute another weak module lattice symplectic automorphism for $\mathcal{O}_{\mathbb{L}}^2$ efficiently when \mathbb{L} is a CM number field. Specially, the weak symplectic automorphism will become a module lattice automorphism immediately when a totally real number field \mathbb{K} is considered, which directly yields a provable deterministic polynomial-time algorithm solving $\mathcal{O}_{\mathbb{K}}^2$ -LIP where \mathbb{K} is a totally real number field.

Note that the forgery security of HAWK [8] is based on the hardness of the one more SVP (omSVP), which asks the adversary to find one more short enough non-trivial element in $\mathcal{O}_{\mathbb{L}}^2$ that is out of the trivial set $\{\alpha x\}_{\alpha \in \mu(\mathbb{L})}$ where x is a given short element. However, our weak symplectic automorphism φ , which can be computed efficiently, will yield another non-trivial short element $\varphi(x)$ directly, whose length is as the same as x 's. This invalidates the omSVP assumption used in HAWK's forgery security analysis, although it does not yield any actual attacks against HAWK itself. An easy way to fix this issue is just adjusting the omSVP assumption by adding the new short elements we find into the trivial set.

We also generalize the algorithm to solve module-LIP for the rank-2 module $M \subset \mathbb{K}^2$ where the number field \mathbb{K} is a totally real number field. By introducing a similar pseudo symplectic automorphism and utilizing eigenspaces to acquire isomorphism invariants, we have the following theorem.

Theorem 1.1 (informal). *Let \mathbb{K} be a totally real number field. $M \subseteq \mathbb{K}^2$ is an module lattice of rank 2 with pseudobasis \mathbf{B} and pseudo-Gram matrix \mathbf{G} . \mathbf{G}' is the pseudo-Gram matrix of M' isomorphic to M . If \mathbf{U} is congruence matrices*

between \mathbf{G} and \mathbf{G}' , then there is a deterministic polynomial time algorithm to find \mathbf{U} , given a basis of $\mathcal{O}_{\mathbb{K}}$, \mathbf{B} , and \mathbf{G}' .

The main contributions are summarized as below:

- We introduce a new tool called module lattice symplectic automorphism into designing algorithms solving module-LIP, and reduce the problem solving module-LIP to finding the certain symplectic automorphism. With this framework, we propose a provable deterministic polynomial-time algorithm that solves module-LIP for the rank-2 module $M \subset \mathbb{K}^2$ where \mathbb{K} is a totally real number field.
- Compared with algorithms in [20], our algorithms are provable deterministic polynomial-time algorithm while the algorithms in [20] need some heuristic assumptions. Moreover, our algorithm, that solves module-LIP for the rank-2 module M in totally real number field, always runs in polynomial time regardless of the the arithmetic properties of the module, whereas the time complexity of algorithm in [20] relates to the arithmetic properties.
- We invalidates the omSVP assumption introduced by HAWK to prove its forgery security. Therefore, necessary adjustment about the omSVP assumption should be made to guarantee the validity of the security proof. We stress that our results haven't yielded any actual attack against HAWK.

1.2 Technical Overview

Isomorphism and Automorphism. From a geometric perspective, LIP is to find the unitary matrix O such that $OM = M'$ for isomorphic module lattices M, M' . In this perspective, we call a unitary matrix A such that $AM = M$ an automorphism of M . If O is a unitary matrix such that $OM = M'$, then the automorphisms of M' all have the form OAO^{-1} , where A is an automorphism of M . From the algebraic perspective, LIP is to find the congruence matrix U such that $U^*GU = G'$, where $G = B^*B$ (B is a basis of the lattice). In this perspective, we call U such that $U^*GU = G$ an automorphism of G . The automorphisms of G all have the form $B^{-1}UB$, where U is a unitary matrix. There have been many works showing that automorphisms can play important roles in solving LIP [2, 13, 16, 17].

The key to our technique is that, for the module lattice $\mathcal{O}_{\mathbb{L}}^2$, we find a lattice isomorphism that has not been considered before. In particular, if the base field is also considered to be a totally real number field, this lattice isomorphism will also be a module lattice isomorphism with more algebraic structures.

Specifically, we will exploit the symplectic property of rank 2 matrices (i.e. $B^T J_2 B = J_2, \forall B \in \text{SL}_2(\mathbb{L})$, here $J_2 = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$) and its variants to extract automorphism, such as computing $B^{-1}J_2B$ from $B^T B$ (See Subsect. 3.1).

Analysis of $\mathcal{O}_{\mathbb{L}}^2$ -LIP. We reduce the problem of solving $\mathcal{O}_{\mathbb{L}}^2$ -LIP to the problem of finding out the certain module lattice symplectic automorphism by using

the Lenstra-Silverberg algorithm proposed in [19]. Informally speaking, this algorithm can find an isomorphism between a lattice and its certain canonical form using specific automorphisms of the lattice. The module lattice $\mathcal{O}_{\mathbb{L}}^2$ has the automorphisms $\{aI_2 \mid a \in \mu(\mathbb{L})\}$ ($\mu(\mathbb{L})$ denotes the roots of unity in \mathbb{L}) inherently, but these automorphisms are not enough for Lenstra-Silverberg algorithm. We discovered and carefully demonstrated that adding the certain module lattice symplectic automorphism to the above-mentioned automorphisms meets the requirements for the algorithm in [19].

It is worth mentioning that the main theorem (Theorem 2.1 in this paper) in [19] will be used over and over again as a powerful tool in our technique. However, before us it seems that people only focused on the original version [13].

As the discussion before, this symplectic automorphism can be computed if the considered field is a totally real number field. For HAWK, we can obtain a weak symplectic automorphism, and it will affect the existing omSVP assumptions. This invalidates the assumption used in their security analysis, although it does not yield attacks against the construction itself.

Algorithm for Rank-2 Module-LIP over Totally Real Number Field.

We now explain how our algorithm for module-LIP works when the module $M \subset \mathbb{K}^2$ has rank-2 and when the number field \mathbb{K} is a totally real number field. Firstly, we can still first obtain a pseudo-automorphism of M , which we call pseudo because it does not preserve M . To be specific, for $M' = OM$ where O is a unitary matrix, we can obtain OJ_2O^{-1} . If we look at the pseudo-automorphism from the perspective of matrix conjugation, then the eigenspace before conjugating differs from the eigenspace after conjugating by only one transition matrix for the same eigenvalue.

From a high level view, a fundamental reason why the module isomorphism problem for rank 2 is harder than for rank 1 lies in the fact that it is hard to find rank 1 submodule $N \subseteq M$, $N' \subseteq M'$ such that $N' = ON$. The intersection of the modules and eigenspaces of pseudo-automorphisms provides the submodules N, N' s.t. $N' = ON$, but previously we only knew automorphisms of pure quantities, that is, they only have trivial eigenspaces. This new (pseudo) automorphism fits our requirements nicely. And then rank 1 module-LIP can be solved by using algorithm in [19]. It should be pointed out that the eigenvalues and eigenvectors of this automorphism need to be lifted to be considered in $\mathcal{O}_{\mathbb{L}}$, and thus need to be argued more carefully.

In addition, for better intuition, we here give our technical overview from a geometric point of view. However, the geometric perspective and Gram matrix perspective can be transformed into each other. For computational reasons, the actual algorithm will be performed from the Gram matrix perspective. We will give a sketch of the actual algorithm at the beginning of Sect. 4.

Roadmap. The rest of the paper is organized as follows. Section 2 provides basic definitions and preliminaries. In Sect. 3, we present a provable determinis-

tic polynomial-time algorithm to solve $\mathcal{O}_{\mathbb{L}}^2$ -LIP where \mathbb{L} is a CM number field, with the help of a new module lattice symplectic automorphism, and we also show how to find a weak module lattice symplectic automorphism efficiently, which can invalidate the omSVP assumption introduced by HAWK to prove its forgery security. In Sect. 4, we present the provable deterministic polynomial-time algorithm to solve module-LIP for the rank-2 module $M \subset \mathbb{K}^2$ where the number field \mathbb{K} is a totally real number field. Section 5 concludes the paper shortly.

2 Notations and Preliminaries

2.1 Notations

- The Euclidean norm of $a \in \mathbb{R}^n$ is denoted by $\|a\|$. The transpose of A is denoted by A^T , and $(A^{-1})^T$ is abbreviated as A^{-T} . Let $GL_n(\mathbb{R})$ and $GL_n(\mathbb{Z})$ be the general linear group of rank n over \mathbb{R} and \mathbb{Z} respectively.
- We use J_2 to represent the matrix $\begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$, J_B to represent $B^{-1}J_2B$ for some 2×2 matrix B . We use rI_n to represent the matrix $\text{diag}(r, r, \dots, r)$, and sometimes use r to represent rI_n in matrix multiplications (such as $r \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} rx_1 \\ rx_2 \end{pmatrix}$). We will also emphasize this point from time to time in the proof.
- For a number field \mathbb{K} , the parameter \mathbf{K} denotes degree of \mathbb{K} , $\log \Delta_{\mathbb{K}}$, and a basis of $\mathcal{O}_{\mathbb{K}}$.
- For x in a number field \mathbb{K} , we call x^* is the complex conjugation of x if $\sigma(x^*) = \overline{\sigma(x)}$, $\forall \sigma \in \text{Hom}_{\mathbb{Q}}(\mathbb{K}, \mathbb{C})$. For matrix $H = (h_{ij})$, let H^* denote $(h_{ij}^*)^T$ and \overline{H} denote $(\overline{h_{ij}})$ if all h_{ij}^* exist.
- For a ring A in a number field that are closed under complex conjugating, the unitary matrices over A is $\mathcal{U}_n(A) := \{T \in M_n(A) \mid T^*T = I_n\}$.
- For a number field \mathbb{K} , we use $\mu(\mathbb{K})$ to denote the roots of unity in \mathbb{F} . Note $\mu(\mathbb{K}) \subset \mathcal{O}_{\mathbb{K}}$ and $\mu(\mathbb{K}) = \mathcal{U}_1(\mathcal{O}_{\mathbb{K}})$
- Assume G is an abelian group. For abelian groups A, B equipped a bilinear map $\varphi : A \times B \rightarrow G$, we define the group product $A \cdot B$ as the abelian group generated by $\{\varphi(a, b)\}$. We also usually use $a \cdot b$ to denote $\varphi(a, b)$. Further more, if there are canonical bilinear maps respectively between (A, B) , (B, C) , $(A \cdot B, C)$, $(A, B \cdot C)$ satisfying associative law i.e. $(a \cdot b) \cdot c = a \cdot (b \cdot c)$, $\forall a \in A, b \in B, c \in C$, then the group product also have associative law, i.e. $(A \cdot B) \cdot C = A \cdot (B \cdot C)$. For example, matrix groups $A \subseteq \mathbb{K}^{n \times m}$, $B \subseteq \mathbb{K}^{m \times l}$, $C \subseteq \mathbb{K}^{l \times t}$ or A, B, C are ideals of a ring.

2.2 Lattices

Lattices are discrete additive subgroups of \mathbb{R}^m . A lattice is usually defined by a set of n linearly independent basis vectors $b_1, b_2, \dots, b_n \in \mathbb{R}^m$. Any point in the lattice can be expressed as an integer linear combination of the basis vectors.

A lattice \mathcal{L} of rank n and dimension m is a set of points in \mathbb{R}^m that can be expressed as integer combinations of n linearly independent basis vectors b_1, \dots, b_n . Denote $B = (b_1, \dots, b_n)$ as the basis of the lattice \mathcal{L} , and then $\mathcal{L} = \{Bz : z \in \mathbb{Z}^n\}$.

2.3 Number Theory

A number field \mathbb{K} is a finite extension of the rational numbers \mathbb{Q} . Any such \mathbb{K} is isomorphic to $\mathbb{Q}[X]/(P)$ for an irreducible monic polynomial P . The degree of P matches the degree of the extension. For any extension \mathbb{K} of degree d , there are exactly d embeddings $\sigma_1, \dots, \sigma_d$ from \mathbb{K} into the complex numbers \mathbb{C} . If an embedding sends \mathbb{K} into the real numbers \mathbb{R} , it's called a real embedding. Otherwise, it's called complex. If an embedding is not real, it can be paired with its complex conjugate to give another distinct complex embedding. We use r_1 to denote the count of real embeddings and r_2 to denote the count of complex embeddings up to conjugation. Therefore, the total count of embeddings is $d = r_1 + 2r_2$. When all embeddings are real (i.e. $r_1 = d$), we say the extension $\mathbb{K}|\mathbb{Q}$ is totally real. Conversely, when all embeddings are complex (i.e. $2r_2 = d$), we call it totally imaginary.

CM Number Field. A CM (number) field \mathbb{L} is a number field if it's a quadratic extension \mathbb{L}/\mathbb{K} where the base field \mathbb{K} is totally real but \mathbb{L} is totally imaginary. The extension \mathbb{L}/\mathbb{K} is a Galois extension and we denote the Galois group by $Gal(\mathbb{L}/\mathbb{K})$. There is a complex conjugation in $Gal(\mathbb{L}/\mathbb{K})$, i.e. $\exists \tau \in Gal(\mathbb{L}/\mathbb{K})$ s.t. $\forall x \in \mathbb{L}, \sigma_i(\tau(x)) = \overline{\sigma_i(x)}$. We usually denote $\tau(x)$ by x^* .

Canonical Embedding. We call this map $\sigma : x \in \mathbb{K} \mapsto (\sigma_1(x), \dots, \sigma_d(x))^T \in \mathbb{C}^d$ canonical embedding of number field \mathbb{K} . We will often identify \mathbb{K} with the image underlying its canonical embedding, then $\mathcal{O}_{\mathbb{K}}$ is a lattice. But note that we are not representing elements in \mathbb{K} using the canonical embedding.

The norm map defined over \mathbb{K} is $\mathcal{N}_{\mathbb{K}}(z) = \prod_i \sigma_i(z)$. Similarly, the trace map is $\text{Tr}_{\mathbb{K}}(z) = \sum_i \sigma_i(z)$. Regard $z \in \mathbb{K}$ as \mathbb{Q} -linear map $m_z : x \in \mathbb{K} \mapsto zx \in \mathbb{K}$, then we have $\mathcal{N}_{\mathbb{K}}(z) = \det(m_z)$ and $\text{Tr}_{\mathbb{K}}(z) = \text{Tr}(m_z)$. Especially, if $z \in \mathbb{Q}$, then $\mathcal{N}_{\mathbb{K}}(z), \text{Tr}_{\mathbb{K}}(z)$ belong to \mathbb{Q} . When there is no ambiguity, we drop the subscript.

The \mathbb{R} -algebra $\mathbb{K}_{\mathbb{R}} := \mathbb{K} \otimes_{\mathbb{Q}} \mathbb{R}$ is a real vector space of dimension d . If write \mathbb{K} as $\mathbb{Q}[X]/(P)$, then we can use $\mathbb{R}[X]/(P)$ to denote $\mathbb{K} \otimes_{\mathbb{Q}} \mathbb{R}$. To keep the discussion concise, this paper will not delve deeply into the discussion about $\mathbb{K}_{\mathbb{R}}$.

Rings of Integer. Let $\mathcal{O}_{\mathbb{K}}$ denote the ring of integers of a number field \mathbb{K} . $\mathcal{O}_{\mathbb{K}}$ is a free \mathbb{Z} -module of rank d . The discriminant of \mathbb{K} , denoted $\Delta_{\mathbb{K}}$, is defined as $(\det(\sigma_i(\alpha_j))_{i,j})^2 \in \mathbb{Z}$, where $(\alpha_j)_{1 \leq j \leq d}$ is any basis of $\mathcal{O}_{\mathbb{K}}$. Specifically, there exists some absolute constant $c > 1$ such that $\Delta_{\mathbb{K}} \geq c^d$ for all number fields \mathbb{K} . In particular, we always have $d = \text{poly}(\log \Delta_{\mathbb{K}})$.

When \mathbb{K} is a totally real number field and $\mathbb{L} = \mathbb{K}[X]/(X^2 + 1)$, Mureau et.al. showed $\log \Delta_{\mathbb{L}} = \text{poly}(\log \Delta_{\mathbb{K}})$ and the following lemma in [20, Section 2.2].

Lemma 2.1 ([20, Lemma 2.6]). *Let \mathbb{K} be a totally real number field and $\mathbb{L} := \mathbb{K}[X]/(X^2 + 1)$. There exists a polynomial time algorithm A that, given as input a \mathbb{Z} -basis $B_{\mathbb{K}}$ of $\mathcal{O}_{\mathbb{K}}$, computes a \mathbb{Z} -basis $B_{\mathbb{L}}$ of $\mathcal{O}_{\mathbb{L}}$.*

Lemma 2.2. *Let \mathbb{L} be a CM number field with degree n . Then $\forall x \in \mathcal{O}_{\mathbb{L}} \setminus \{0\}$, we have: $\text{Tr}_{\mathbb{L}}(x^*x) \geq n$, and $\text{Tr}_{\mathbb{L}}(x^*x) = n$ iff x is a root of unity.*

Proof. Note $\forall 1 \leq i \leq n$, $\sigma_i(x^*x) = (\sigma_i(x))^* \sigma_i(x) = |\sigma_i(x)|^2 \geq 0$. So $\text{Tr}_{\mathbb{L}}(x^*x) = \sum_{i=1}^n \sigma_i(x^*x) \geq n(\prod_{i=1}^n |\sigma_i(x^*x)|)^{1/n} = n(\mathcal{N}(x^*x))^{1/n} \geq n$. The last inequality holds for $\forall r \in \mathcal{O}_{\mathbb{L}}$, $\mathcal{N}(r) \geq 1$. This means $\text{Tr}_{\mathbb{L}}(x^*x) = n$ iff all the equals are taken iff $|\sigma_i(x)| = 1$ for all $1 \leq i \leq n$ iff¹ x is a root of unity. \square

Ideals. An (fractional) ideal \mathcal{I} is an finitely generated additive subgroup of \mathbb{K} such that $x \cdot \mathcal{I} \subseteq \mathcal{I}$ for all $x \in \mathcal{O}_{\mathbb{K}}$. When an ideal is contained in $\mathcal{O}_{\mathbb{K}}$, we call it an integral ideal and usually use the fraktur lower-case letter to denote it (e.g. \mathfrak{a}). Principal ideal is an ideal generated by a single element $a \in \mathbb{K}$ i.e. $a\mathcal{O}_{\mathbb{K}}$. The product of two ideals \mathcal{I} and \mathcal{J} is their group product i.e. $\mathcal{I}\mathcal{J} = \{\sum_i x_i y_i | x_i \in \mathcal{I}, y_i \in \mathcal{J}\}$. An ideal \mathcal{I} has the form $\frac{1}{d} \cdot \mathfrak{a}$, where $d \in \mathcal{O}_{\mathbb{K}} \setminus \{0\}$, $\mathfrak{a} \subset \mathcal{O}_{\mathbb{K}}$. Then we can define the algebraic norm $\mathcal{N}(\mathcal{I}) := \sharp(\mathcal{O}_{\mathbb{K}}/\mathfrak{a})/\sharp(\mathcal{O}_{\mathbb{K}}/(d\mathcal{O}_{\mathbb{K}}))$.

Modules. Assume \mathbb{K} is a number field. An $\mathcal{O}_{\mathbb{K}}$ module M is a subset of \mathbb{K}^ℓ of the form $b_1\mathcal{I}_1 + \dots + b_r\mathcal{I}_r$, where the \mathcal{I}_i 's are non-zero fractional ideals of \mathbb{K} and (b_1, \dots, b_r) are \mathbb{K} -linearly independent vectors of \mathbb{K}^ℓ , for some $\ell > 0$. We call $\mathbf{B} = (B, (\mathcal{I}_i)_{1 \leq i \leq r})$ a pseudo-basis for M , where B is the matrix whose columns are the b_i . The integer r is called the rank of the module. When $r = \ell$, we say that the module has full rank.

2.4 Module-LIP

Definition 2.1. *Let $\mathbf{B} = (B, (\mathcal{I}_i)_{1 \leq i \leq \ell})$ be a pseudo-basis of a rank- ℓ module M in $\mathbb{K}_{\mathbb{R}}^k$. The pseudo-Gram matrix associated with \mathbf{B} is denoted by $\mathbf{G} := (G, (\mathcal{I}_i)_{1 \leq i \leq \ell})$, where $G = B^*B$.*

Definition 2.2. *Let $\mathbf{G} = (G, (\mathcal{I}_i)_{1 \leq i \leq \ell})$ and $\mathbf{G}' = (G', (\mathcal{J}_i)_{1 \leq i \leq \ell})$ be two pseudo-Gram matrices. They are said to be congruent if there exists $U = (u_{i,j})_{1 \leq i,j \leq \ell} \in GL_{\ell}(\mathbb{K})$ such that $G' = U^*GU$ and $u_{i,j} \in \mathcal{I}_i\mathcal{J}_j^{-1}$, $v_{i,j} \in \mathcal{J}_i\mathcal{I}_j^{-1}$, where $V = (v_{i,j})_{1 \leq i,j \leq \ell} := U^{-1}$. Such U is called a congruence matrix between \mathbf{G} and \mathbf{G}' . This defines an equivalence relation \sim on the set of pseudo-Gram matrices.*

In [20] Mureau et al. proposed three equivalent definitions of isomorphism between module lattices, and we only elaborate here the one we will use.

¹ This is a basic result in number theory. One can find a argument in [20, Lemma 2.14].

Definition 2.3. Let $M, M' \subset \mathbb{K}_{\mathbb{R}}^{\ell}$ be two modules of rank ℓ with respective pseudo-bases $\mathbf{B} = (B, (\mathcal{I}_i)_{1 \leq i \leq \ell})$ and $\mathbf{B}' = (B', (\mathcal{J}_i)_{1 \leq i \leq \ell})$. Let \mathbf{G} (resp. \mathbf{G}') be the pseudo-Gram matrix associated with \mathbf{B} (resp. \mathbf{B}'). We say that M, M' are isomorphic as module lattices if \mathbf{G} and \mathbf{G}' are congruent.

Definition 2.4 (module-LIP $_{\mathbb{K}}^{\mathbf{B}}$). For \mathbf{B} a pseudo-basis of a module lattice $M \subset \mathbb{K}_{\mathbb{R}}^{\ell}$ with associated pseudo-Gram matrix \mathbf{G} , the (worst-case) search module lattice Isomorphism Problem with parameter \mathbf{K} and \mathbf{B} , denoted by module-LIP $_{\mathbf{K}}^{\mathbf{B}}$, is, given as input any pseudo-Gram matrix $\mathbf{G}' \sim \mathbf{G}$ (see Definition 2.2), to find a congruence matrix between \mathbf{G} and \mathbf{G}' .

2.5 Algorithmic Consideration

Representation of Ideals and Modules. Assume $B_{\mathcal{O}_{\mathbb{K}}} = (\alpha_j)_{j=1, \dots, d}$ is a basis of $\mathcal{O}_{\mathbb{K}}$. We represent elements in \mathbb{K} (resp. $\mathbb{K}_{\mathbb{R}}$) by their coordinates in the basis $B_{\mathcal{O}_{\mathbb{K}}}$, which is a vector in \mathbb{Q}^d (resp. \mathbb{R}^d). For $x \in \mathbb{K}$ represented by the vector $(x_1, \dots, x_d)^T \in \mathbb{Q}^d$, we define $\text{size}(x) := \sum_i \text{size}(x_i)$, where $\text{size}(a/b) := \lceil \log_2 |a| \rceil + \lceil \log_2 |b| \rceil$ for $a, b \in \mathbb{Z}$ coprime. As is customary, we assume that in this paper the $B_{\mathcal{O}_{\mathbb{K}}}$ is always an LLL-reduced basis of $\mathcal{O}_{\mathbb{K}}$, meaning that $\sigma(B_{\mathcal{O}_{\mathbb{K}}})$ forms an LLL-reduced basis of $\mathcal{O}_{\mathbb{K}}$. This choice is made to ensure that the coefficients of $\alpha_i \alpha_j$ under $B_{\mathcal{O}_{\mathbb{K}}}$ representation do not blow up.

In fact, $\sigma(B_{\mathcal{O}_{\mathbb{K}}})$ being LLL-reduced implies that for any integral $x \in \mathcal{O}_{\mathbb{K}}$, $\text{size}(x) = \text{poly}(d, \|\sigma(x)\|)$. Inversely, $\|\sigma(x)\| \leq \sum_i |x_i| \cdot \|\sigma(\alpha_i)\| \leq d^{3/2} \cdot 2^d \cdot (\Delta_{\mathbb{K}}^{1/d}) \cdot \max_i |x_i|$ since $\lambda_d(\mathcal{O}_{\mathbb{K}}) \leq \sqrt{d} \cdot (\Delta_{\mathbb{K}}^{1/d})$. This implies that the arithmetic operations on elements in $\mathcal{O}_{\mathbb{K}}$ are in polynomial time. And then the arithmetic operations on elements in \mathbb{K} are in polynomial time.

A fractional ideal \mathcal{I} is represented by a \mathbb{Z} -basis (y_1, \dots, y_d) of the ideal, such that $(\sigma(y_i))_{1 \leq i \leq d}$ is an LLL-reduced basis of $\sigma(\mathcal{I})$. In particular, we have $\|\sigma(y_i)\| \leq 2^d \cdot \lambda_d(\mathcal{I}) \leq \sqrt{d} \cdot 2^d \cdot \Delta_{\mathbb{K}}^{3/(2d)} \cdot \mathcal{N}(\mathcal{I})^{1/d}$ (see e.g. [20, Section 2.3]). We define $\text{size}(\mathcal{I}) := \sum_i \text{size}(y_i)$.

Lemma 2.3 ([20, Lemma 2.9]). Let $B = (B, (\mathcal{I}_i)_{1 \leq i \leq r})$ be a pseudo-basis of a rank r module M in \mathbb{K}^{ℓ} . Then, one can compute in polynomial time a basis $C \in \mathbb{C}^{d\ell \times dr}$ of $\sigma(M)$ such that the column vectors c_i of C satisfy $\|c_i\| \leq \sqrt{d} \cdot 2^d \cdot (\Delta_{\mathbb{K}}^{3/(2d)}) \cdot \max_{1 \leq j \leq r} \|\sigma(b_j)\| \cdot \mathcal{N}(\mathcal{I}_j)^{1/d}$, where b_j is the j -th column of B .

Basic Algorithms

Lemma 2.4 ([11, Lemma 2.8]). With the representation of ideals as described above, one can sum up two ideals \mathcal{I} and \mathcal{J} in time $\text{poly}(\text{size}(\mathcal{I}), \text{size}(\mathcal{J}))$, multiply two ideals \mathcal{I} and \mathcal{J} in time $\text{poly}(\text{size}(\mathcal{I}), \text{size}(\mathcal{J}), \log \Delta_{\mathbb{K}})$, compute the inverse of an ideal \mathcal{I} in time $\text{poly}(\text{size}(\mathcal{I}), \log \Delta_{\mathbb{K}})$.

As a generalization of the product of ideals, when the bilinear map between two abelian groups satisfies that it runs in polynomial time in the input size and

outputs a lattice vector of polynomial size in the input size, we can compute the group product of these two abelian groups with the \mathbb{Z} -basis of them as input in polynomial time.

The following lemma guarantees that the computation of roots of unity in a given field is efficient.

Lemma 2.5 ([20, Corollary 2.11]). *Let \mathbb{K} be a degree d number field. Then, \mathbb{K} has at most $2d^2$ roots of unity, and there exists a polynomial-time algorithm that, given a basis of the ring of integers $\mathcal{O}_{\mathbb{K}}$, computes the roots of unity in \mathbb{K} .*

Lenstra-Silverberg Algorithm. Gentry and Szydło initially proposed an algorithm in [13] to recover x from x^*x and xR (where R is a certain type of polynomial ring). Later, Lenstra and Silverberg extended this in [17–19]. We describe here the main theorem presented in [19], which will be used later in Sect. 3 and Sect. 4.

Definition 2.5. *An order is a commutative ring of which the additive group is isomorphic to \mathbb{Z}^n for some $n \in \mathbb{Z}_{\geq 0}$. A CM-order A is an order such that:*

1. *A has no non-zero nilpotent elements.*
2. *A is equipped with an conjugate automorphism $x \mapsto \bar{x}$ of A such that $\varphi(\bar{x}) = \overline{\varphi(x)}$ for all $x \in A$ and all ring homomorphisms $\varphi : A \rightarrow \mathbb{C}$.*

Definition 2.6. *Let A be a CM-order. A lattice L is an A -lattice if it's given an A -module structure with the property that for all $a \in A$ and $x, y \in L$ one has $\langle ax, y \rangle = \langle x, \bar{a}y \rangle$.*

An example of an A -lattice is the A -module A itself, with inner product $\langle a, b \rangle = \text{Tr}(\bar{a}b)$; here $\text{Tr} : A \rightarrow \mathbb{Z}$ is the trace function of A as a \mathbb{Z} -algebra. This A -lattice is called the standard A -lattice.

In algorithms, we can represent an order by a system $(b_{ijk})_{i,j,k=1}^n$ of integers with the property that, for some \mathbb{Z} -basis $\alpha_1, \dots, \alpha_n$ of the order, one has $\alpha_i \alpha_j = \sum_{k=1}^n b_{ijk} \alpha_k$ for all $1 \leq i, j \leq n$. In other words, for \mathbb{Z} -basis $\alpha_1, \dots, \alpha_n$ of the order, we specify the order by matrix representation of m_{α_i} under the basis $\alpha_1, \dots, \alpha_n$, where m_{α_i} means the action of multiplying α_i . A lattice is specified by the Gram matrix of a \mathbb{Z} -basis b_1, \dots, b_m . An A -lattice is specified as a lattice and a system of nm^2 integer coefficients that express $\alpha_i b_j$ on b_1, \dots, b_m , where the $(\alpha_i)_{i=1}^n$ and $(b_j)_{j=1}^m$ are as above.

Definition 2.7. *An A -isomorphism $f : L \rightarrow M$ of A -lattices is an isomorphism of A -modules with $\langle f(x), f(y) \rangle = \langle x, y \rangle$ for all $x, y \in L$.*

One can see that if there is an A -isomorphism between an A -lattice L and A -module M which is also a lattice, then M is also an A -lattice.

Theorem 2.1 ([19, Theorem 1.5]). *There is a deterministic polynomial-time algorithm that, given a CM-order A and an A -lattice L , decides whether or not L is A -isomorphic with the standard A -lattice, and if so, computes such an A -isomorphism.*

3 Solving $\mathcal{O}_{\mathbb{L}}^2$ -LIP with Module Symplectic Automorphism

In this section, we focus on the $\mathcal{O}_{\mathbb{L}}^2$ -LIP, in which \mathbf{B} is taken to be $(I_2, (\mathcal{O}_{\mathbb{L}}))$ in module-LIP $_{\mathbb{L}}^{\mathbf{B}}$, and \mathbb{L} is a CM number field.

We firstly present a deterministic polynomial-time algorithm solving $\mathcal{O}_{\mathbb{L}}^2$ -LIP with the help of certain module lattice automorphism of $\mathcal{O}_{\mathbb{L}}^2$. It seems not easy to find such a module lattice automorphism, but we can compute another weak module lattice automorphism. This weak module lattice automorphism will invalidate the omSVP assumption used in the security analysis of HAWK.

3.1 An Algorithm for $\mathcal{O}_{\mathbb{L}}^2$ -LIP with Automorphism of $\mathcal{O}_{\mathbb{L}}^2$

In this subsection, we will mainly give a direct application of the Lenstra-Silverberg algorithm on $\mathcal{O}_{\mathbb{L}}^2$ -LIP as Theorem 3.1. Essentially, this provides a reduction from $\mathcal{O}_{\mathbb{L}}^2$ -LIP to finding certain module lattice automorphisms of $\mathcal{O}_{\mathbb{L}}^2$.

Recall $J_2 := \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$.

Theorem 3.1. *Let \mathbb{L} be a CM number field with degree $2d$ and $B \in GL_2(\mathcal{O}_{\mathbb{L}})$. There is a deterministic polynomial-time algorithm that, given a basis of $\mathcal{O}_{\mathbb{L}}$, B^*B , and $B^{-1}J_2B$, outputs $\mathcal{U}_2(\mathcal{O}_{\mathbb{L}})B$.*

Lemma 3.1. *Let \mathbb{L} be a CM number field with degree $2d$. Then*

$$\mathcal{U}_2(\mathcal{O}_{\mathbb{L}}) = \left\{ \begin{pmatrix} \xi_1 & 0 \\ 0 & \xi_2 \end{pmatrix} \mid \xi_1, \xi_2 \in \mu(\mathbb{L}) \right\} \cup \left\{ \begin{pmatrix} 0 & \xi_1 \\ \xi_2 & 0 \end{pmatrix} \mid \xi_1, \xi_2 \in \mu(\mathbb{L}) \right\}.$$

Furthermore, $\#\mathcal{U}_2(\mathcal{O}_{\mathbb{L}}) \leq 2\#\mu(\mathbb{L})^2 \leq 128d^4$.

Proof. Assume $U = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \mathcal{U}_2(\mathcal{O}_{\mathbb{L}})$, then $2d = \text{Tr}_{\mathbb{L}}(1) = \text{Tr}_{\mathbb{L}}(a^*a + b^*b) = \text{Tr}_{\mathbb{L}}(a^*a) + \text{Tr}_{\mathbb{L}}(b^*b)$. By Lemma 2.2, one of a or b is zero, and the other is a root of unity. Do same discussion for c, d , one of c or d is zero, and the other is a root of unity. Similarity, one of a or c is zero, and the other is a root of unity. \square

Lemma 3.2. *Use the notation in Theorem 3.1. Denote $B^{-1}J_2B$ by J_B , $\mathcal{O}_{\mathbb{L}}$ by R . Take $H := \langle J_B \rangle = \{I_2, J_B, -I_2, -J_B\}$. Then we can define modified group ring $R \langle H \rangle := R[H] / \langle I_2 + (-I_2) \rangle = R \cdot I_2 + R \cdot J_B$. Denote I_2 by e , J_B by σ , $R \langle H \rangle$ by \tilde{R} . Then \tilde{R} is a CM-order. We usually use $ae + b\sigma$ to denote the element in \tilde{R} , where $a, b \in R$. And then $\text{Tr}(ae + b\sigma)$ is just $2\text{Tr}_{\mathbb{L}}(a)$.*

Proof. R, H is commutative, so $R \langle H \rangle$ is commutative and the additive group is isomorphic to \mathbb{Z}^{4d} .

Assume $ae + b\sigma$ is nilpotent in \tilde{R} , in which $a, b \in R$. Then $(ae + b\sigma)^m = 0$ for some $m \in \mathbb{Z}_+$. Consider $(bx + a)^m, x^2 + 1 \in R[x]$. Then $\exists r(x) \in R[x]$ s.t. $\deg(r) \leq 1$, and $(bx + a)^m - r(x) \in \langle x^2 + 1 \rangle$. Assume $r(x) = cx + d$, then we have

$(ae + b\sigma)^m - (c\sigma + de) = 0$ for $\sigma^2 + e = 0$. Therefore, $c\sigma + de = 0$ and this means $c = d = 0$. If $bx + a \neq 0$ then $r(x) \neq 0$ since $x^2 + 1$ doesn't divide $(bx + a)^m$. It's a contradiction. So $b\sigma + ae = 0$.

The conjugate automorphsim is $ae + b\sigma \mapsto a^*e - b^*\sigma$. Then $\forall \varphi : \tilde{R} \rightarrow \mathbb{C}$, $\varphi(\overline{ae}) = \overline{\varphi(ae)}$ since R is a CM-order and its conjugate automorphism is the complex conjugation. And $\varphi(\sigma)^2 = \varphi(\sigma^2) = -1 \Rightarrow \varphi(\sigma) \in \{\pm i\} \Rightarrow \overline{\varphi(\sigma)} = -\varphi(\sigma) = \varphi(\overline{\sigma})$. So $\varphi(\overline{r}) = \overline{\varphi(r)}$ for all $r \in \tilde{R}$.

Thus \tilde{R} is a CM-order. Note $\tilde{R} = Re \oplus R\sigma$ and $Re, R\sigma$ are invariant under $ae \Rightarrow \text{Tr}(ae) = \text{Tr}(ae|_{Re}) + \text{Tr}(ae|_{R\sigma}) = 2\text{Tr}_R(a) = 2\text{Tr}_{\mathbb{L}}(a)$. Here Tr_R means trace on R . Similarly, since $b\sigma(Re) \subseteq R\sigma, b\sigma(R\sigma) \subseteq Re, \text{Tr}(b\sigma) = 0$. \square

Proof (proof of Theorem 3.1). Use the setting in Lemma 3.2. We take $M := \mathcal{O}_{\mathbb{L}}^2$ with the inner product $\langle \cdot, \cdot \rangle_M : (x, y) \in M^2 \mapsto 2\text{Tr}_{\mathbb{L}}(x^*B^*By) \in \mathbb{Z}$. \tilde{R} acts on M as $(ae + b\sigma, m) \in \tilde{R} \times M \mapsto a \cdot m + b \cdot \sigma \cdot m$ (matrix multiplication). It makes M a \tilde{R} -module (Note that the commutativity between rI_2 and σ matrices multiplication is utilized here to ensure the associativity of the ring operation.)

Assume $e_1 = B^{-1} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = e \cdot e_1, e_2 = B^{-1} \begin{pmatrix} 0 \\ -1 \end{pmatrix} = \sigma \cdot e_1 \in M$. Define $f : r \in \tilde{R} \mapsto r \cdot e_1$. Then we have $\forall a_1e + b_1\sigma, a_2e + b_2\sigma \in \tilde{R}$,

$$\begin{aligned} & \frac{1}{2} \langle f(a_1e + b_1\sigma), f(a_2e + b_2\sigma) \rangle_M \\ &= \text{Tr}_{\mathbb{L}}((a_1 \cdot e_1 + b_1 \cdot e_2)^* B^* B (a_2 \cdot e_1 + b_2 \cdot e_2)) \\ &= \text{Tr}_{\mathbb{L}} \left(\begin{pmatrix} a_1^* & -b_1^* \end{pmatrix} (B^*)^{-1} B^* B B^{-1} \begin{pmatrix} a_2 \\ -b_2 \end{pmatrix} \right) \\ &= \text{Tr}_{\mathbb{L}} \left(\begin{pmatrix} a_1^* & -b_1^* \end{pmatrix} \begin{pmatrix} a_2 \\ -b_2 \end{pmatrix} \right) \\ &= \text{Tr}_{\mathbb{L}}(a_1^* a_2 + b_1^* b_2) \\ &= \frac{1}{2} \text{Tr}((a_1^* a_2 + b_1^* b_2)e + (a_1^* b_2 - b_1^* a_2)\sigma) \\ &= \frac{1}{2} \text{Tr}(\overline{(a_1e + b_1\sigma)}(a_2e + b_2\sigma)) \\ &= \frac{1}{2} \langle a_1e + b_1\sigma, a_2e + b_2\sigma \rangle. \end{aligned}$$

Obviously f is homomorphism of \tilde{R} -module. Note $f(ae + b\sigma) = a \cdot e_1 + b \cdot e_2 = B^{-1} \begin{pmatrix} a \\ -b \end{pmatrix}$. f is injective since $f(ae + b\sigma) = 0 \Rightarrow B^{-1} \begin{pmatrix} a \\ -b \end{pmatrix} = 0 \Rightarrow \begin{pmatrix} a \\ -b \end{pmatrix} = 0 \Rightarrow a = b = 0$. f is surjective since $\forall v \in M$, assume $Bv = \begin{pmatrix} a \\ -b \end{pmatrix} \in \mathcal{O}_{\mathbb{L}}^2$, then $f(ae + b\sigma) = B^{-1} \begin{pmatrix} a \\ -b \end{pmatrix} = v$.

In conclusion, we obtain M is A -isomorphic with the standard A -lattice, and then is a A -lattice. Using the polynomial-time algorithm in Theorem 2.1, we can get an A -isomorphsim ϕ between the standard A -lattice and M . Assume

$\phi(e) = B^{-1} \begin{pmatrix} a_1 \\ b_1 \end{pmatrix}$ for some $a, b \in R$. Then $\text{Tr}_{\mathbb{L}}(a_1^* a_1 + b_1^* b_1) = \frac{1}{2} \langle \phi(e), \phi(e) \rangle_M = \frac{1}{2} \langle e, e \rangle = n$. By Lemma 2.2, we have $a_1 \in \mu(R), b_1 = 0$ or $b_1 \in \mu(R), a_1 = 0$.

Assume $\phi(\sigma) = B^{-1} \begin{pmatrix} a_2 \\ b_2 \end{pmatrix}$. Similarly, we obtain $a_2 \in \mu(R), b_2 = 0$ or $b_2 \in \mu(R), a_2 = 0$. Note $M = R\phi(e) \oplus R\phi(\sigma)$, so a_1, a_2 are not both 0. This means $(\phi(e)|\phi(\sigma)) \in B^{-1}\mathcal{U}_2(\mathcal{O}_{\mathbb{L}})$ (by Lemma 3.1).

Then we compute $(\phi(e)|\phi(\sigma)) \cdot \mathcal{U}_2(\mathcal{O}_{\mathbb{L}}) = B^{-1}(\mu(A) \cdot v)\mathcal{U}_2(\mathcal{O}_{\mathbb{L}})$ in polynomial time since $\#\mathcal{U}_2(\mathcal{O}_{\mathbb{L}}) \leq 128d^4$. □

3.2 New Pseudo Lattice Automorphisms of Rank 2 Module Lattices over a CM Number Field

A very simple but important lemma is given below. It's actually the symplectic property of the 2×2 matrix.

Lemma 3.3. $\forall U \in GL_2(\mathbb{K}), U^T J_2 U = \det(U) \cdot J_2$.

Proof. Assume $U = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$, then

$$U^T J_2 U = \begin{pmatrix} a & c \\ b & d \end{pmatrix} \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} 0 & ad - bc \\ -(ad - bc) & 0 \end{pmatrix} = \det(U) \cdot J_2.$$

□

Proposition 3.1. *Let \mathbb{K} be a number field, $B \in GL_2(\mathbb{K})$, and $r \in \mathbb{K}$. Given as input a basis of $\mathcal{O}_{\mathbb{K}}$, $G = B^T B$, and $\det(B)$, we can compute $J_B := B^{-1} J_2 B$ and $m_r := B^{-1}(rI_2)B$ in the time of polynomial of the input size.*

Proof. We claim that

$$J_B = (\det(B)I_2) \cdot G^{-1} \cdot J_2 \text{ and } m_r = (rI_2),$$

and then the time to compute J_B and m_r is polynomial.

It is obvious that $rI_2 = B^{-1}(rI_2)B$ since rI_2 is in center of $M_2(\mathbb{K})$, and we also have

$$\begin{aligned} & (\det(B)I_2) \cdot G^{-1} \cdot J_2 \\ &= (\det(B)I_2)B^{-1}(B^T)^{-1}J_2B^{-1}B \\ &= B^{-1}(\det(B)I_2)((B^{-1})^T J_2 B^{-1}) B \\ &= B^{-1}(\det(B)I_2)(\det(B^{-1}))J_2 B \\ &= B^{-1}J_2 B, \end{aligned}$$

where the third equality holds by Lemma 3.3. □

Lemma 3.4. *Let \mathbb{L} be a CM number field. Define $t_* : \begin{pmatrix} x \\ y \end{pmatrix} \in \mathbb{L}^2 \mapsto \begin{pmatrix} x^* \\ y^* \end{pmatrix} \in \mathbb{L}^2$. It's an \mathbb{Q} linear map. We claim that $\forall U \in GL_2(\mathbb{L}), U^* J_2 t_* U = \det(U)^* \cdot J_2 t_*$.*

Proof. Note that for all $B \in M_2(\mathbb{L}), t_* \circ B = (B^*)^T \circ t_*$ as \mathbb{Q} linear map. Assume $U = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$, then

$$\begin{aligned} U^* J_2 t_* U &= \begin{pmatrix} a^* & c^* \\ b^* & d^* \end{pmatrix} \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \begin{pmatrix} a^* & b^* \\ c^* & d^* \end{pmatrix} t_* \\ &= \begin{pmatrix} 0 & (ad - bc)^* \\ -(ad - bc)^* & 0 \end{pmatrix} t_* \\ &= \det(U)^* \cdot J_2 t_*. \end{aligned}$$

□

Proposition 3.2 *Let \mathbb{L} be a CM number field, $B \in GL_2(\mathbb{L})$, and $r \in \mathbb{L}$. Given as input a basis of $\mathcal{O}_{\mathbb{L}}$, $G = B^* B$, and $\det(B)$, we can compute $B^{-1} J_2 t_* B$ and $m_r := B^{-1}(r I_2) B$ in the time of polynomial of the input size.*

Proof. The computation of m_r is same as Proposition 3.1. Similarly we claim $B^{-1} J_2 t_* B = (\det(B)^* I_2) \cdot G^{-1} \cdot J_2 t_*$, and then the time to compute $B^{-1} J_2 t_* B$ is polynomial.

$$\begin{aligned} (\det(B)^* I_2) \cdot G^{-1} \cdot J_2 t_* &= (\det(B)^* I_2) B^{-1} (B^*)^{-1} J_2 t_* 2 B^{-1} B \\ &= B^{-1} (\det(B)^* I_2) ((B^{-1})^* J_2 t_* B^{-1}) B \\ &= B^{-1} (\det(B)^* I_2) (\det(B^{-1})^*) J_2 t_* B \\ &= B^{-1} J_2 t_* B, \end{aligned}$$

where the third equality holds by Lemma 3.4. □

Remark 1. We refer to $B^{-1} J_2 B$ and $B^{-1} J_2 t_* B$ as pseudo lattice automorphisms. To further distinguish them, we refer to $B^{-1} J_2 B$ as a pseudo module lattice automorphism. We use the term 'pseudo' because when they act on the module lattice generated by the pseudo-basis \mathbf{B} , the resulting elements may not necessarily still belong to the original module lattice. We call them automorphisms because the inner product (over the \mathbb{Q} -vector space) induced by $G = B^* \cdot B$ of a vector remains the same under the pseudo-automorphisms. More precisely, in the case of $B^{-1} J_2 t_* B$ (the case of $B^{-1} J_2 t_* B$ is similar), for any vectors $v_i = B^{-1}(x_i, y_i)^T, i = 1, 2$, their inner product induced by G is $\text{tr}_{L/\mathbb{Q}}(v_1^* G v_2) = \text{tr}_{L/\mathbb{Q}}(x_1^* x_2 + y_1^* y_2)$. Applying the pseudo-automorphism on v_i , the images become $(B^{-1} J_2 t_* B) B^{-1}(x_i, y_i)^T = B^{-1}(y_i^*, -x_i^*)^T$. Consequently, the inner product of the images is $\text{tr}_{L/\mathbb{Q}}(x_1 x_2^* + y_1 y_2^*) = \text{tr}_{L/\mathbb{Q}}(x_1^* x_2 + y_1^* y_2)$, the same as before. If considered on the \mathbb{L} -vector space, $B^{-1} J_2 t_* B$ cannot preserve the inner product, but $B^{-1} J_2 B$ can.

3.3 Impact of Additional Automorphism on HAWK

In this section, we will show the impact of additional automorphism on HAWK [8]. HAWK² is one of the brightest prospects at round one of the NIST for additional digital signatures [21]. HAWK is defined over a degree n , which is a power of two (equal to 256, 512 or 1024). A HAWK private key is a randomly generated basis for the lattice \mathbb{Z}^{2n} , consisting of four polynomials $f, g, F, G \in R_n = \mathbb{Z}[X]/(X^n + 1)$, where f and g have small coefficients and together they satisfy the NTRU equation

$$fG - gF = 1 \pmod{X^n + 1}$$

The lattice secret basis B is $\begin{pmatrix} f & F \\ g & G \end{pmatrix}$ and the public key is

$$Q = B^*B = \begin{pmatrix} f^*f + g^*g & f^*F + g^*G \\ F^*f + G^*g & F^*F + G^*G \end{pmatrix}$$

In order to provide formal justification for the strong unforgeability under chosen message attack of HAWK, they formally introduce omSVP and they provide reductions in the (quantum) random oracle model from HAWK to omSVP³, i.e. if there exists an adversary \mathcal{A} against the (Q)ROM-SUF-CMA game of HAWK, then there exists an adversary \mathcal{B} against the SAMPLE game in Fig. 1. The omSVP is defined as follows.

Definition 3.1 (Average case omSVP [8]). *An average case **omSVP** instance is the pair **ac-omSVP** = (**Init**, **samp**). On input 1^n , **Init** returns a form Q sampled from some distribution over $\mathcal{H}_{\ell_n}(\mathbb{K}_n)$, the roots of unity $\mu(\mathbb{K}_n)$ for \mathbb{K}_n , a length bound L_n , and a Gaussian parameter σ_n . On input Q , **samp** returns a sample from D_{Q, σ_n} . The adversary in Fig. 1 wins whenever it can utilize the form Q and the samples it receives from **samp** to produce a non-trivial new element of $\mathcal{O}_{\mathbb{K}}^{\ell}$ that is sufficiently short.*

SAMPLE _{ac-omSVP, A} (1^n) 1: $\mathcal{L} \leftarrow \{0\}$ 2: $(Q, \mu(\mathbb{K}), L, \sigma) \leftarrow \text{Init}(1^n)$ 3: $x^* \leftarrow \mathcal{A}^{\text{samp}(Q)}(Q)$ 4: return $\llbracket \ x^*\ _Q \leq L \wedge x^* \notin \mathcal{L}_{\text{samples}} \rrbracket$	samp(Q) 1: $x \leftarrow D_{Q, \sigma}$ 2: $\mathcal{L}_{\text{samples}} \leftarrow \mathcal{L}_{\text{samples}} \cup \{\alpha x\}_{\alpha \in \mu(\mathbb{K})}$ 3: return x
--	---

Fig. 1. The SAMPLE game

Here the “non-trivial new” depends on the definition of $\mathcal{L}_{\text{samples}}$ in $\text{samp}(Q)$ in the SAMPLE game. In the SAMPLE game of HAWK, $Q = B^*B$, we can

² see <https://hawk-sign, info>.

³ See chapter 6 of the HAWK specification document for details.

think that the sample x we get has the form $x = B^{-1} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$ and $\| \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \| < L$. However, Proposition 3.2 tells us that in addition to $\{\alpha x\}_{\alpha \in \mu(\mathbb{K})}$, there is another type of trivial new vector that we can obtain efficiently. More specifically, we can compute the automorphism $B^{-1}J_2t_*B$ by Proposition 3.2 and for a given sample $x = B^{-1} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$, applying this automorphism to x we will obtain an element $x^* = B^{-1}J_2t_*B \cdot B^{-1} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = B^{-1} \begin{pmatrix} x_2^* \\ -x_1^* \end{pmatrix}$ in Fig. 1. Note that $\|x^*\| = \|x\| < L$ and $x^* \notin \{\alpha x\}_{\alpha \in \mu(\mathbb{K})}$, thus the $\{\alpha x^*\}_{\alpha \in \mu(\mathbb{K})}$ are non-trivial new elements. In the case of HAWK, $\mu(\mathbb{K}) = \{X^i : i = 0, \dots, 2n - 1\}$, combined with automorphism $B^{-1}J_2t_*B$, we get a subgroup G of $\text{Aut}(Q)$ and G is isomorphic to the dihedral group D_{2n} ⁴. At present, it seems that this automorphism has little impact on HAWK, but whether this automorphism will have a greater impact on HAWK requires further research in the future.

It is worth noting that omSVP and forging signatures on Hawk are not completely equivalent. Intuitively even if one can find an x^* that wins the SAMPLE game, one must also find a message and salt that hashes into a particular coset to make this a successful signature forgery. For more information see HAWK [8].

What’s more, Theorem 3.1 tells us that in secret key recovery of HAWK, given $Q = B^*B$, if we can find $B^{-1}J_2B$, then we can get the secret key B efficiently by Theorem 3.1, or equivalently, we have the following corollary, namely, if we have additional information B^TB , then we can find the secret key B . This provide new perspectives for cryptanalysis of HAWK.

Corollary 3.1. *Let \mathbb{L} be a CM number field and $B \in GL_2(\mathcal{O}_{\mathbb{L}})$. There is a deterministic polynomial-time algorithm that, given a basis of $\mathcal{O}_{\mathbb{L}}$, B^*B , and B^TB , computes $\mathcal{U}_2(\mathcal{O}_{\mathbb{L}})B$.*

Proof. Recall $t_* : \begin{pmatrix} x \\ y \end{pmatrix} \in \mathbb{L}^2 \mapsto \begin{pmatrix} x^* \\ y^* \end{pmatrix} \in \mathbb{L}^2$ and $t_* \circ B = (B^*)^T \circ t_*$ as \mathbb{Q} linear map. We can compute $(B^TB)^{-1}(B^*B)^T t_*$ as \mathbb{Q} -linear map. Then note $(B^TB)^{-1}(B^*B)^T t_* = B^{-1}(B^*)^T t_* = B^{-1}t_*B$. By [20, Theorem 2.15] or Proposition 4.1, we can find $\det(B)$ from $\det(G)$ in polynomial time. Using Proposition 3.2, then we can compute $B^{-1}J_2t_*B$. In conclusion, we can compute $B^{-1}t_*B \cdot B^{-1}J_2t_*B = B^{-1}J_2B$ in polynomial time. Then we use Theorem 3.1. \square

4 An Algorithm for Module-LIP in Rank 2 over Totally Real Number Fields

In this section, let \mathbb{K} be a totally real number field and $\mathbb{L} = \mathbb{K}[X]/(X^2 + 1)$. By Lemma 2.1 and the discussion of size, we can always assume that the input parameter \mathbf{K} and the input parameter \mathbf{L} are equivalent. We can think of X as

⁴ D_{2n} refers to the symmetries of the $2n$ -gon, a group of order $4n$.

the imaginary unit i , but sometime we use X again. We'll use this notation a lot.

In last section, all operations performed on \mathbb{L} in Corollary 3.1 can be directly applied to \mathbb{K} , then we can directly obtain a algorithm for solving $\mathcal{O}_{\mathbb{K}}^2$ -LIP. This is a deterministic polynomial-time algorithm for $\mathcal{O}_{\mathbb{K}}^2$ -LIP, where \mathbb{K} is a totally real number field. In contrast, the result in [20] only offers a heuristic polynomial-time algorithm for it.

In this section, with different approach and more elaborate processing, we present a deterministic polynomial-time algorithm for the module-LIP of rank 2 over a totally real number field as this following theorem.

Theorem 4.1. *Let \mathbb{K} be a totally real number field and $\mathbb{L} = \mathbb{K}[X]/(X^2 + 1)$. $M \subseteq \mathbb{K}^2$ is an module lattice of rank 2 with pseudobasis parameter \mathbf{B} . Algorithm 7 takes as input parameter \mathbf{K} , \mathbf{B} , and \mathbf{G}' an instance of module-LIP $_{\mathbf{K}}^{\mathbf{B}}$, runs in the polynomial time in the size of the input and finds all congruence matrices between \mathbf{G} and \mathbf{G}' .*

To illustrate the structure of our algorithm, we first present an informal version of Algorithm 7 as below.

Algorithm 1: FindU(Informal)

Require: Parameter $\mathbf{B} = (B, (\mathcal{I}_i)_i)$ and \mathbf{K} . An module-LIP $_{\mathbf{K}}^{\mathbf{B}}$ instance $\mathbf{G}' = (G', (\mathcal{J}_i)_i)$.

Ensure: A congruence matrix U between \mathbf{G} (the pseudo-Gram matrix associated with \mathbf{B}) and \mathbf{G}' .

- 1: $\pm J_{BU} \leftarrow \text{ConjOfJ}(\mathbf{B}, \mathbf{K}, \mathbf{G}')$ (Proposition 4.2)
 - 2: $(\mathcal{I}_{\mathbf{B}}, v_{\mathbf{B}}, (BU)^{-1}\mathcal{I}_{\mathbf{B}}v_{\mathbf{B}}) \leftarrow \text{EigenSubLat}(J_{BU}, \mathbf{B}, \mathbf{K}, \mathbf{G}')$ (Proposition 4.3)
 - 3: $\mu(\mathbb{L}) \cdot (BU)^{-1}v_{\mathbf{B}} \leftarrow \text{UseLS}(\mathcal{I}_{\mathbf{B}}, v_{\mathbf{B}}, (BU)^{-1}\mathcal{I}_{\mathbf{B}}v_{\mathbf{B}}, \mathbf{B}, \mathbf{K}, \mathbf{G}')$ (Proposition 4.4)
 - 4: $BU \leftarrow (v_{\mathbf{B}}|\overline{v_{\mathbf{B}}})((BU)^{-1}v_{\mathbf{B}}|(BU)^{-1}\overline{v_{\mathbf{B}}})^{-1}$
 - 5: $U \leftarrow B^{-1}BU$
 - 6: **return** U .
-

Roughly speaking, our algorithm can be divided into three steps.

Firstly, we extract a 'automorphism' of \mathbf{G}' from the information of parameters \mathbf{B} and \mathbf{G}' , denoted by $J_{BU} = (BU)^{-1}J_2(BU)$. We call this step as ConjOfJ that means finding conjugation of J_2 .

Secondly, we identify the intersection of the eigenspace of this automorphism and the direct product of ideals in G , showing that it differs from the eigenspace of J_2 intersected with the lattice defined by B only by a factor of BU . This intersection is essentially a rank 1 module lattice, and with the BU factor accounted for, we can easily compute a pseudo-basis for it. We call this step as EigenSubLat that means computing sublattice composed by eigenvector.

Finally, we multiply the lattice obtained from the intersection by the inverse of the ideal derived from the previously computed pseudo-basis, resulting in a

cyclic module lattice $\mathcal{O}_{\mathbb{L}} \cdot v$. Again, we will also obtain the value BUv . So we can use the Lenstra-Silverberg algorithm to recover v from $\mathcal{O}_{\mathbb{L}} \cdot v$, in the sense of a difference of one root of unity. We call this step as UseLS that simply means using Lenstra-Silverberg algorithm. With BUv and v known, we can easily recover BU .

4.1 Application of Lenstra-Silverberg algorithm

First we use the Theorem 2.1 (Lenstra-Silverberg algorithm) to give two specific algorithms, which will play an important role in the subsequent proofs. The first algorithm in the proposition is just Theorem 2.15 in [20], while the second algorithm can be seen as a high-dimensional generalization of the first algorithm. It is worth noting that the two algorithms in Proposition 4.1 are actually algorithms for solving rank-1 module-LIP in \mathbb{K} and \mathbb{K}^2 , respectively. We can generalize them into a unified form for solving rank-1 module-LIP, but we have chosen this current representation for easier understanding. So far, it seems that only the first algorithm has received attention, both in terms of applications and implementations.

Proposition 4.1. *Let \mathbb{F} be a CM-field or a totally real number field with degree n . Let A be the ring of integers of \mathbb{F} . [19, Examples 3.7(i)(ii)] showed that A is a CM-order. The conjugate automorphism is just the complex conjugation $x \mapsto x^*$, and the trace function is just $Tr_{\mathbb{F}}$.*

1. For $\alpha \in \mathbb{F}$, there is a deterministic polynomial-time algorithm LS1 that, given A , αA and $\alpha^* \alpha$, then we can find $\alpha \mu(A)$ in polynomial time, where $\mu(A)$ means roots of unity in A .
2. For $v = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} \in \mathbb{F}^2$ and $B \in GL_2(\mathbb{F})$, there is a deterministic polynomial-time algorithm LS2 that, given A , $B^* B$, $v^* v = v_1 v_1^* + v_2 v_2^*$, and $B^{-1}(A \cdot v)$, then we can find $B^{-1}(\mu(A) \cdot v)$ in polynomial time, where $\mu(A)$ means roots of unity in A .

Algorithm 2: LS1(informal)

- Require:** $\mathcal{O}_{\mathbb{F}}$; lattice $\alpha \mathcal{O}_{\mathbb{F}}$ for some $\alpha \in \mathbb{F}; \alpha^* \alpha$
Ensure: $\alpha \mu(\mathcal{O}_{\mathbb{F}})$.
- 1: $\langle \cdot \rangle_M \leftarrow ((x, y) \in (\alpha \mathcal{O}_{\mathbb{F}})^2 \mapsto Tr_{\mathbb{F}}(\frac{x^* y}{\alpha^* \alpha}))$
 - 2: $M \leftarrow (\alpha \mathcal{O}_{\mathbb{F}}, \langle \cdot \rangle_M)$
 - 3: $W \leftarrow \mathbf{LS}(\mathcal{O}_{\mathbb{F}}, M)$
 - 4: **return** W .
-

Algorithm 3: LS2(informal)

Require: $\mathcal{O}_{\mathbb{F}}$; lattice $B^{-1}(\mathcal{O}_{\mathbb{F}} \cdot v)$ for some $v \in \mathbb{F}^2; v^*v, B^*B$
Ensure: $B^{-1}(\mu(\mathcal{O}_{\mathbb{F}}) \cdot v)$.
1: $\langle \cdot, \cdot \rangle_M \leftarrow ((x, y) \in (B^{-1}(\mathcal{O}_{\mathbb{F}} \cdot v))^2 \mapsto \text{Tr}_{\mathbb{F}}(\frac{x^*(B^*B)y}{v^*v}))$
2: $M \leftarrow (B^{-1}(\mathcal{O}_{\mathbb{F}} \cdot v), \langle \cdot, \cdot \rangle_M)$
3: $W \leftarrow \text{LS}(\mathcal{O}_{\mathbb{F}}, M)$
4: **return** W .

Proof. 1. Let $M = \alpha A$. It's an (fractional) ideal of A , so is an A -module. Define the inner product in M as $\langle \cdot, \cdot \rangle_M : (x, y) \in M^2 \mapsto \text{Tr}_{\mathbb{F}}(\frac{x^*y}{\alpha^*\alpha}) \in \text{Tr}_{\mathbb{F}}(A) \subseteq \mathbb{Z}$. Then M is a integral lattice. Consider $f : a \in A \mapsto \alpha a \in M$. Obviously it's an isomorphism of A -module, and one can see $\langle f(x), f(y) \rangle_M = \langle x, y \rangle$ for all $x, y \in A$ by the definition of $\langle \cdot, \cdot \rangle_M$. So f is an A -isomorphism, and M is an A -lattice isomorphic to standard A -lattice.

Using the polynomial-time algorithm in Theorem 2.1, we can get an A -isomorphism ϕ between the standard A -lattice and M . Assume $\phi(1) = \alpha \cdot a$ for some $a \in A$. Then $\text{Tr}_{\mathbb{F}}(aa^*) = \langle \phi(1), \phi(1) \rangle_M = \langle 1, 1 \rangle = n$. By Lemma 2.2, we have $a \in \mu(A)$. Then we compute $\phi(1) \cdot \mu(A) = \alpha\mu(A)$ in polynomial time since $\#\mu(A) \leq 2n^2$.

2. Let $M = B^{-1}(A \cdot v) = A \cdot (B^{-1}v)$. It's an A -module. Define the inner product in M as $\langle \cdot, \cdot \rangle_M : (x, y) \in M^2 \mapsto \text{Tr}_{\mathbb{F}}(\frac{x^*(B^*B)y}{v^*v}) \in \text{Tr}_{\mathbb{F}}(A) \subseteq \mathbb{Z}$. Then M is a integral lattice.

Consider $f : a \in A \mapsto B^{-1}(av) = a(B^{-1}v) \in M$. Obviously it's an isomorphism of A -module, and one can see $\langle f(x), f(y) \rangle_M = \langle x, y \rangle$ for all $x, y \in A$ by the definition of $\langle \cdot, \cdot \rangle_M$. So f is an A -isomorphism, and M is an A -lattice isomorphic to standard A -lattice.

Using the polynomial-time algorithm in Theorem 2.1, we can get an A -isomorphism ϕ between the standard A -lattice and M . Assume $\phi(1) = B^{-1}(av)$ for some $a \in A$. Then $\text{Tr}_{\mathbb{F}}(aa^*) = \langle \phi(1), \phi(1) \rangle_M = \langle 1, 1 \rangle = n$. By Lemma 2.2, we have $a \in \mu(A)$. Then we compute $\mu(A) \cdot \phi(1) = B^{-1}(\mu(A) \cdot v)$ in polynomial time since $\#\mu(A) \leq 2n^2$. \square

4.2 Find J_{BU}

Using the Proposition 3.1, we only need to know $\det(BU)$ to obtain J_{BU} using G' . We have known B and thus $\det(B)$. It remains to find $\det(U)$. This can be done by Gentry's algorithm i.e. LS1. To do so, we first need to extract $\det(U) \cdot \mathcal{O}_{\mathbb{K}}$ from (\mathcal{I}_i) and (\mathcal{J}_i) , which is not unexpected since U actually gives an isomorphism from $\bigoplus \mathcal{I}_i$ to $\bigoplus \mathcal{J}_i$.

Lemma 4.1 ([5, Proposition 1.4.2]). *Follow the setup as in Definition 2.2. We have $\det(U) \cdot \mathcal{O}_{\mathbb{K}} = \prod_{k=1}^{\ell} \mathcal{I}_k \prod_{k=1}^{\ell} \mathcal{J}_k^{-1}$.*

Proof. For any permutation $\sigma \in S_\ell$,

$$\prod_{k=1}^{\ell} u_{k\sigma(k)} \in \prod_{k=1}^{\ell} \mathcal{I}_k \mathcal{J}_{\sigma(k)}^{-1} = \prod_{k=1}^{\ell} \mathcal{I}_k \prod_{k=1}^{\ell} \mathcal{J}_k^{-1}.$$

By the definition of determinant, $\det(U) \in \prod_{k=1}^{\ell} \mathcal{I}_k \prod_{k=1}^{\ell} \mathcal{J}_k^{-1}$ i.e.

$$\det(U) \cdot \mathcal{O}_{\mathbb{K}} \subseteq \prod_{k=1}^{\ell} \mathcal{I}_k \prod_{k=1}^{\ell} \mathcal{J}_k^{-1}.$$

Symmetrically, we have $\det(V) \cdot \mathcal{O}_{\mathbb{K}} \subseteq \prod_{k=1}^{\ell} \mathcal{J}_k \prod_{k=1}^{\ell} \mathcal{I}_k^{-1}$. Note that $\det(V) = \det(U)^{-1}$, so

$$\det(U) \cdot \mathcal{O}_{\mathbb{K}} = \prod_{k=1}^{\ell} \mathcal{I}_k \prod_{k=1}^{\ell} \mathcal{J}_k^{-1}.$$

□

Proposition 4.2. *Follow the setup provided in Theorem 4.1. U is a congruence matrix. There is a deterministic polynomial-time algorithm that, given parameter $\mathbf{B}, \mathbf{K}, \mathbf{G}'$, computes $\pm J_{BU}$.*

Proof. We show Algorithm 4 satisfies the requirements.

Algorithm 4: ConjOfJ

Require: Parameter $\mathbf{B} = (B, (\mathcal{I}_i)_i)$ and \mathbf{K} . An module-LIP $_{\mathbb{K}}^{\mathbf{B}}$ instance $\mathbf{G}' = (G', (\mathcal{J}_i)_i)$.

Ensure: $\pm J_{BU}$ for a congruence matrix U between \mathbf{G} (the pseudo-Gram matrix associated with \mathbf{B}) and \mathbf{G}' .

- 1: $\det(U) \cdot \mathcal{O}_{\mathbb{K}} \leftarrow \prod_{k=1}^{\ell} \mathcal{I}_k \prod_{k=1}^{\ell} \mathcal{J}_k^{-1}$
 - 2: $\det(U)^2 \leftarrow \frac{\det(G')}{\det(B)^2}$
 - 3: $W_1 \leftarrow \text{LS1}(\mathbf{K}, \det(U) \cdot \mathcal{O}_{\mathbb{K}}, \det(U)^2)$
 - 4: $W_2 \leftarrow \det(B) \cdot W_1 \cdot G'^{-1} \cdot J_2$
 - 5: **return** W_2 .
-

Correctness: Step 1 is right by Lemma 4.1. Step 2 is right since $\det(G') = \det(U^T B^T B U) = \det(U^T) \det(B^T) \det(B) \det(U) = \det(B)^2 \det(U)^2$. Step 3 is right by Proposition 4.1.

Complexity: At Step 1, both ideals' multiplication and inversion run in polynomial time. At Step 2, both elements' multiplication and inversion run in polynomial time. Step 3 also runs in polynomial time by Proposition 4.1. \square

Remark 2. Here, Algorithm LS1 can be replaced by an algorithm for computing square roots over algebraic number fields.

Remark 3. If we consider B as a matrix over $\mathbb{K}_{\mathbb{R}}$, where $\det(U)$ is still an element of \mathbb{K} , we can therefore obtain an approximate value of $\det(U)^2$ by calculating $\frac{\det(G')}{\det(B)^2}$, and then recover the exact value of $\det(U)^2$.

4.3 Find Sub Module Lattice Composed by Eigenvectors of J_{BU}

For the remainder of this section, we need to transfer the whole setting from \mathbf{K} to \mathbf{L} , since the eigenvalues $\pm i$ of J_2 are not in \mathbb{K} . Specifically, we will consider $(B, (\mathcal{I}_i \mathcal{O}_{\mathbb{L}}))$ instead of $(B, (\mathcal{I}_i))$, $(G', (\mathcal{J}_i \mathcal{O}_{\mathbb{L}}))$ instead of $(G', (\mathcal{J}_i))$. We can compute $\mathcal{I}_i \mathcal{O}_{\mathbb{L}}$ by calculating $\sum_{j=1}^d y_{ij} \mathcal{O}_{\mathbb{L}}$, where $\{y_{ij}\}$ are a LLL-reduced \mathbb{Z} -basis for \mathcal{I}_i . It can be done in time $\text{poly}(\text{size}(\mathcal{I}_i), \log \Delta_{\mathbb{L}})$. We do the same thing for \mathcal{J}_i . So we can assume we input $(B, (\mathcal{I}_i \mathcal{O}_{\mathbb{L}})), (G', (\mathcal{J}_i \mathcal{O}_{\mathbb{L}}))$ when we input \mathbf{B}, \mathbf{G}' . The following lemma tells us that the congruence matrix U does not change.

Lemma 4.2. *Follow the setup provided in Definition 2.2. Define $\mathcal{I}'_k := \mathcal{I}_k \cdot \mathcal{O}_{\mathbb{L}}, \mathcal{J}'_k := \mathcal{J}_k \cdot \mathcal{O}_{\mathbb{L}} 1 \leq k \leq \ell$. Then $U(\bigoplus_{k=1}^{\ell} \mathcal{J}'_k) = \bigoplus_{k=1}^{\ell} \mathcal{I}'_k$. Here, the term "direct sum" refers to the Cartesian product.*

Proof. Assume $e_i \in \mathbb{L}^{\ell}$, and its i -th component is 1, while the rest are 0. Then $\bigoplus_{k=1}^{\ell} \mathcal{I}'_k = \sum_{k=1}^{\ell} \mathcal{O}_{\mathbb{L}} \cdot \mathcal{I}_k \cdot e_k = \mathcal{O}_{\mathbb{L}} \sum_{k=1}^{\ell} \mathcal{I}_k \cdot e_k = \mathcal{O}_{\mathbb{L}} \bigoplus_{k=1}^{\ell} \mathcal{I}_k$. (Note that the products here can all be viewed as group products of Abelian groups, and it is easy to verify the second equality sign from the point of view of Abelian groups.) Similarly, $\bigoplus_{k=1}^{\ell} \mathcal{J}'_k = \mathcal{O}_{\mathbb{L}} \bigoplus_{k=1}^{\ell} \mathcal{J}_k$. Since U and elements in $\mathcal{O}_{\mathbb{L}}$ commute under matrix multiplication, it's enough to show $U(\bigoplus_{k=1}^{\ell} \mathcal{J}_k) = \bigoplus_{k=1}^{\ell} \mathcal{I}_k$. In the setting in Definition 2.2, $U = (u_{ij})$ and $u_{ij} \in \mathcal{I}_i \cdot \mathcal{J}_j^{-1}$. So $\forall v \in \bigoplus_{k=1}^{\ell} \mathcal{J}_k$, Uv 's i -th component is in \mathcal{I}_i i.e. $Uv \in \bigoplus_{k=1}^{\ell} \mathcal{I}_k$. This means $U(\bigoplus_{k=1}^{\ell} \mathcal{J}_k) \subseteq \bigoplus_{k=1}^{\ell} \mathcal{I}_k$. Symmetrically, we have $U^{-1}(\bigoplus_{k=1}^{\ell} \mathcal{I}_k) \subseteq \bigoplus_{k=1}^{\ell} \mathcal{J}_k$. Thus $U(\bigoplus_{k=1}^{\ell} \mathcal{J}_k) = \bigoplus_{k=1}^{\ell} \mathcal{I}_k$. \square

Remark 4. Lemma 4.2 essentially states the following: if $(B, (\mathcal{I}_i)_i)$ and $(B', (\mathcal{J}_i)_i)$ are two pseudo-bases of the same module lattice $M \subset \mathbb{K}^{\ell}$, then $(B, (\mathcal{I}_i \mathcal{O}_{\mathbb{L}})_i)$ and $(B', (\mathcal{J}_i \mathcal{O}_{\mathbb{L}})_i)$ are two pseudo-bases of the same module lattice $\mathcal{O}_{\mathbb{L}} M \subset \mathbb{L}^{\ell}$.

In the following we show by computation the intersection of the eigenspace of J_2 with the module lattice defined by \mathbf{B} . And then consider the analogue after conjugating.

Definition 4.1. *Let \mathbb{K} be a totally real number field and $\mathbb{L} = \mathbb{K}[X]/(X^2+1)$. For parameter \mathbf{B} and \mathbf{K} , assume $B = \begin{pmatrix} a & c \\ b & d \end{pmatrix} \in \mathbb{K}^{2 \times 2}$. Define $\mathcal{I}_{\mathbf{B}} := (b - ia)\mathcal{I}_1 \mathcal{O}_{\mathbb{L}} \cap$*

$(\iota c - d)\mathcal{I}_2\mathcal{O}_{\mathbb{L}}$, and $v_{\mathbf{B}} := \frac{1}{b-\iota a} \begin{pmatrix} a \\ b \end{pmatrix} + \frac{1}{\iota c-d} \begin{pmatrix} c \\ d \end{pmatrix}$. We can see $\mathcal{I}_{\mathbf{B}}$ is an (fractional) ideal of $\mathcal{O}_{\mathbb{L}}$, $v_{\mathbf{B}} \in \mathbb{L}^2$, and $\mathcal{I}_{\mathbf{B}}, v_{\mathbf{B}}, \mathcal{I}_{\mathbf{B}}^{-1}$ can all be computed in polynomial time with inputs of the parameters \mathbf{B}, \mathbf{K} .

Lemma 4.3. *Follow the setup provided in Theorem 4.1. U is a congruence matrix. Denote BU by \tilde{B} . Assume $J_{\tilde{B}} = \tilde{B}^{-1}J_2\tilde{B}$ and $m_i = \iota I_2 = \tilde{B}^{-1}\iota I_2\tilde{B}$. Then $\ker(J_2 - \iota I_2) \cap B(\mathcal{I}_1\mathcal{O}_{\mathbb{L}} \oplus \mathcal{I}_2\mathcal{O}_{\mathbb{L}}) = \mathcal{I}_{\mathbf{B}} \cdot v_{\mathbf{B}}$ and $\ker(J_{\tilde{B}} - m_i) \cap (\mathcal{J}_1\mathcal{O}_{\mathbb{L}} \oplus \mathcal{J}_2\mathcal{O}_{\mathbb{L}}) = \tilde{B}^{-1}(\mathcal{I}_{\mathbf{B}} \cdot v_{\mathbf{B}})$.*

Proof. Firstly,

$$\begin{aligned} & \ker(J_2 - \iota I_2) \cap B(\mathcal{I}_1\mathcal{O}_{\mathbb{L}} \oplus \mathcal{I}_2\mathcal{O}_{\mathbb{L}}) \\ &= \{r_1 \begin{pmatrix} a \\ b \end{pmatrix} + r_2 \begin{pmatrix} c \\ d \end{pmatrix} \mid r_j \in \mathcal{I}_j\mathcal{O}_{\mathbb{L}}, J_2(r_1 \begin{pmatrix} a \\ b \end{pmatrix} + r_2 \begin{pmatrix} c \\ d \end{pmatrix}) = \iota(r_1 \begin{pmatrix} a \\ b \end{pmatrix} + r_2 \begin{pmatrix} c \\ d \end{pmatrix})\} \\ &= \{r_1 \begin{pmatrix} a \\ b \end{pmatrix} + r_2 \begin{pmatrix} c \\ d \end{pmatrix} \mid r_j \in \mathcal{I}_j\mathcal{O}_{\mathbb{L}}, \begin{pmatrix} r_1b + r_2d \\ -r_1a - r_2c \end{pmatrix} = \begin{pmatrix} \iota(r_1a + r_2c) \\ \iota(r_1b + r_2d) \end{pmatrix}\} \\ &= \{r_1 \begin{pmatrix} a \\ b \end{pmatrix} + r_2 \begin{pmatrix} c \\ d \end{pmatrix} \mid r_j \in \mathcal{I}_j\mathcal{O}_{\mathbb{L}}, r_1b + r_2d = \iota(r_1a + r_2c)\} \\ &= \{r_1 \begin{pmatrix} a \\ b \end{pmatrix} + r_2 \begin{pmatrix} c \\ d \end{pmatrix} \mid r_j \in \mathcal{I}_j\mathcal{O}_{\mathbb{L}}, r_1(b - \iota a) = r_2(\iota c - d)\} \\ &= \left\{ \frac{r}{b - \iota a} \begin{pmatrix} a \\ b \end{pmatrix} + \frac{r}{\iota c - d} \begin{pmatrix} c \\ d \end{pmatrix} \mid r \in (b - \iota a)\mathcal{I}_1\mathcal{O}_{\mathbb{L}} \cap (\iota c - d)\mathcal{I}_2\mathcal{O}_{\mathbb{L}} \right\} \\ &= (b - \iota a)\mathcal{I}_1\mathcal{O}_{\mathbb{L}} \cap (\iota c - d)\mathcal{I}_2\mathcal{O}_{\mathbb{L}} \cdot \left\{ \frac{1}{b - \iota a} \begin{pmatrix} a \\ b \end{pmatrix} + \frac{1}{\iota c - d} \begin{pmatrix} c \\ d \end{pmatrix} \right\} \\ &= \mathcal{I}_{\mathbf{B}} \cdot v_{\mathbf{B}}, \end{aligned}$$

and then

$$\begin{aligned} & \ker(J_{\tilde{B}} - m_i) \cap (\mathcal{J}_1\mathcal{O}_{\mathbb{L}} \oplus \mathcal{J}_2\mathcal{O}_{\mathbb{L}}) \\ &= \ker(\tilde{B}^{-1}(J_2 - \iota I_2)\tilde{B}) \cap (\mathcal{J}_1\mathcal{O}_{\mathbb{L}} \oplus \mathcal{J}_2\mathcal{O}_{\mathbb{L}}) \\ &= \tilde{B}^{-1}(\ker(J_2 - \iota I_2) \cap \tilde{B}(\mathcal{J}_1\mathcal{O}_{\mathbb{L}} \oplus \mathcal{J}_2\mathcal{O}_{\mathbb{L}})) \\ &= \tilde{B}^{-1}(\ker(J_2 - \iota I_2) \cap B(\mathcal{I}_1\mathcal{O}_{\mathbb{L}} \oplus \mathcal{I}_2\mathcal{O}_{\mathbb{L}})) \text{ (use Lemma 4.2)} \\ &= \tilde{B}^{-1}(\mathcal{I}_{\mathbf{B}} \cdot v_{\mathbf{B}}) \text{ (by above)}. \end{aligned}$$

□

The following lemma guarantees that we can compute the intersection above efficiently (without knowing \tilde{B}). In fact, the integer version of intersection is already well known, we just need to modify it slightly.

Lemma 4.4. *Let \mathbb{L} be a number field with degree n , and $B_{\mathcal{O}_{\mathbb{L}}}$ be a basis of $\mathcal{O}_{\mathbb{L}}$. Then for $A \in \mathbb{L}^{2 \times 2}$ and a lattice $\mathcal{L} \subseteq \mathbb{L}^2$, there is a deterministic polynomial-time algorithm that, given $B_{\mathcal{O}_{\mathbb{L}}}, A$, and a basis $B_{\mathcal{L}}$ of \mathcal{L} , outputs $\ker(A) \cap \mathcal{L}$.*

Proof. Assume $\text{rank}(\mathcal{L}) = r$. Note that $\ker(A) \cap \mathcal{L} = B_{\mathcal{L}}(\ker(A \cdot B_{\mathcal{L}}) \cap \mathbb{Z}^r)$. Since $A \cdot B_{\mathcal{L}} \in \mathbb{L}^{2 \times r}$, we can compute some $U \in \mathbb{Q}^{2n \times r}$ in polynomial time such that $A \cdot B_{\mathcal{L}} = \begin{pmatrix} B_{\mathcal{O}_{\mathbb{L}}} & 0 \\ 0 & B_{\mathcal{O}_{\mathbb{L}}} \end{pmatrix} U$. Then $\ker(A) \cap \mathcal{L} = B_{\mathcal{L}}(\ker(U) \cap \mathbb{Z}^r)$, and $\ker(U) \cap \mathbb{Z}^r$ can be computed by using Hermit Normal Form (or Smith Normal Form) in polynomial time. \square

Combining the lemmas and definitions in this subsection, we can directly obtain the following proposition.

Proposition 4.3. *Follow the setup provided in Theorem 4.1. U is a congruence matrix. There is a deterministic polynomial-time Algorithm 5 that, given J_{BU} and parameters $\mathbf{B}, \mathbf{K}, \mathbf{G}'$, output the ideal $\mathcal{I}_{\mathbf{B}}$, the vector $v_{\mathbf{B}}$, and the rank 1 module lattice $(BU)^{-1}\mathcal{I}_{\mathbf{B}}v_{\mathbf{B}}$.*

Algorithm 5: EigenSubLat

Require: Parameter $\mathbf{B} = (B, (\mathcal{I}_i)_i)$ and \mathbf{K} . An module-LIP $_{\mathbf{K}}^{\mathbf{B}}$ instance $\mathbf{G}' = (G', (\mathcal{J}_i)_i)$. J_{BU} for a congruence matrix U between \mathbf{G} (the pseudo-Gram matrix associated with \mathbf{B}) and \mathbf{G}' . Write $B = \begin{pmatrix} a & c \\ b & d \end{pmatrix}$.

Ensure: $\mathcal{I}_{\mathbf{B}}, v_{\mathbf{B}}, (BU)^{-1}\mathcal{I}_{\mathbf{B}}v_{\mathbf{B}}$.
 1: $\mathcal{I}_{\mathbf{B}} \leftarrow (b - ia)\mathcal{I}_1\mathcal{O}_{\mathbb{L}} \cap (ic - d)\mathcal{I}_2\mathcal{O}_{\mathbb{L}}$
 2: $v_{\mathbf{B}} \leftarrow \frac{1}{b-ia} \begin{pmatrix} a \\ b \end{pmatrix} + \frac{1}{ic-d} \begin{pmatrix} c \\ d \end{pmatrix}$
 3: $(BU)^{-1}\mathcal{I}_{\mathbf{B}}v_{\mathbf{B}} \leftarrow \ker(J_{BU} - m_i) \cap (\mathcal{J}_1\mathcal{O}_{\mathbb{L}} \oplus \mathcal{J}_2\mathcal{O}_{\mathbb{L}})$
 4: **return** $\mathcal{I}_{\mathbf{B}}, v_{\mathbf{B}}, (BU)^{-1}\mathcal{I}_{\mathbf{B}}v_{\mathbf{B}}$.

Remark 5. Here we have computed a concrete pseudo-basis of $\mathcal{I}_{\mathbf{B}}v_{\mathbf{B}}$ directly. In fact, if we first figure out $\mathcal{I}_{\mathbf{B}}v_{\mathbf{B}}$ by finding $\ker(J_2 - \iota I_2) \cap B(\mathcal{I}_1\mathcal{O}_{\mathbb{L}} \oplus \mathcal{I}_2\mathcal{O}_{\mathbb{L}})$, and then find any pseudo-basis of $\mathcal{I}_{\mathbf{B}}v_{\mathbf{B}}$, it will not affect the following operations. From this point of view it is better to generalize our algorithm to the case on $\mathbb{K}_{\mathbb{R}}$ (in this case our $\mathcal{I}_{\mathbf{B}}$ is not a fractional ideal of $\mathcal{O}_{\mathbb{K}}$).

4.4 Use Lenstra-Silverberg Algorithm

In the last subsection we ended up with a rank 1 module lattice. Just turn it into a cyclic module and we can use the Proposition 4.1.

Proposition 4.4. *Follow the setup provided in Theorem 4.1. U is a congruence matrix. There is a deterministic polynomial-time Algorithm 6 that, given $\mathcal{I}_{\mathbf{B}}, v_{\mathbf{B}}, (BU)^{-1}\mathcal{I}_{\mathbf{B}}v_{\mathbf{B}}$ and parameters $\mathbf{B}, \mathbf{K}, \mathbf{G}'$, output $\mu(\mathbb{L}) \cdot (BU)^{-1}v_{\mathbf{B}}$, where $\mu(\mathbb{L})$ is the roots of unity contained in \mathbb{L} .*

Proof. Denote BU by \tilde{B} and $(BU)^{-1}\mathcal{I}_{\mathbf{B}}v_{\mathbf{B}}$ by \mathcal{L}' .

Algorithm 6: UseLS

Require: Parameters $\mathbf{B} = (B, (\mathcal{I}_i)_i)$ and \mathbf{K} . An module-LIP $_{\mathbf{K}}^{\mathbf{B}}$ instance $\mathbf{G}' = (G', (\mathcal{J}_i)_i)$. The ideal $\mathcal{I}_{\mathbf{B}}$, vector $v_{\mathbf{B}}$. The rank 1 module lattice $(BU)^{-1}\mathcal{I}_{\mathbf{B}}v_{\mathbf{B}}$ for a congruence matrix U between \mathbf{G} (the pseudo-Gram matrix associated with \mathbf{B}) and \mathbf{G}' .

Ensure: $\mu(\mathbb{L}) \cdot (BU)^{-1}v_{\mathbf{B}}$.
 1: $\mathcal{L} \leftarrow \mathcal{I}_{\mathbf{B}}^{-1}(BU)^{-1}\mathcal{I}_{\mathbf{B}}v_{\mathbf{B}}$ (regard $r \in \mathcal{I}_{\mathbf{B}}^{-1}$ as rI_2)
 2: $\omega \leftarrow v_{\mathbf{B}}^*v_{\mathbf{B}}$
 3: $W \leftarrow \text{LS2}(\mathbf{L}, G', \omega, \mathcal{L})$
 4: **return** W .

Correctness: We have $\mathcal{L} = \mathcal{I}_{\mathbf{B}}^{-1} \cdot (\tilde{B})^{-1}\mathcal{I}_{\mathbf{B}}v_{\mathbf{B}} = \tilde{B}^{-1}\mathcal{I}_{\mathbf{B}}^{-1} \cdot \mathcal{I}_{\mathbf{B}} \cdot v_{\mathbf{B}} = \tilde{B}^{-1}(\mathcal{O}_{\mathbb{L}} \cdot v_{\mathbf{B}})$. By Proposition 4.1, we can use $\mathbf{L}, G' = \tilde{B}^*\tilde{B}, \mathcal{L} = \tilde{B}^{-1}(\mathcal{O}_{\mathbb{L}} \cdot v_{\mathbf{B}}), \omega = v_{\mathbf{B}}^*v_{\mathbf{B}}$ to find $\tilde{B}^{-1}\mu(\mathbb{L}) \cdot v_{\mathbf{B}}$.

Complexity: On Step 1, one can compute $\mathcal{I}_{\mathbf{B}}^{-1}\pi_1(\mathcal{L}')$ firstly, where π_1 is the projection of vectors onto their first component (WLOG, we can assume $\pi_1(\mathcal{L}') \neq \{0\}$ i.e. $\pi_1(\tilde{B}^{-1}v_{\mathbf{B}}) \neq 0$). Note $\pi_1(\mathcal{L}')$ is fractional ideal of $\mathcal{O}_{\mathbb{L}}$. (we have shown $\mathcal{L}' = \mathcal{I}_{\mathbf{B}}(\tilde{B}^{-1}v_{\mathbf{B}})$, then $\pi_1(\mathcal{L}') = \mathcal{I}_{\mathbf{B}}\pi_1(\tilde{B}^{-1}v_{\mathbf{B}})$.) So it's a product of fraction ideals and can be computed in polynomial time. Then take one vector $\begin{pmatrix} x_0 \\ y_0 \end{pmatrix}$ of the reduced basis of \mathcal{L}' and consider embedding $\iota : x_1 \in \mathcal{I}_{\mathbf{B}}^{-1}\pi_1(\mathcal{L}') \mapsto \begin{pmatrix} x_1 \\ x_1(x_0^{-1}y_0) \end{pmatrix} \in \mathbb{L}^2$. We have $\iota(\mathcal{I}_{\mathbf{B}}^{-1}\pi_1(\mathcal{L}')) = \mathcal{L}'$. ($\mathcal{L}' = \mathcal{I}_{\mathbf{B}}(\tilde{B}^{-1}v_{\mathbf{B}}) \Rightarrow \forall \begin{pmatrix} x \\ y \end{pmatrix} \in \mathcal{L}', x^{-1}y$ is a constant.) Therefore the image of a basis of $\mathcal{I}_{\mathbf{B}}^{-1}\pi_1(\mathcal{L}')$ under ι is a basis of \mathcal{L}' and can be computed in polynomial time. So step 7 runs in polynomial time. Step 2 is just conjugation and matrix multiplication. Step 3 runs in polynomial time by Proposition 4.1. □

Remark 6. If the reader can accept the language of group products, then Step 1 is just a group product that corresponds matrix multiplications as the bilinear map and lattice vectors in \mathbb{L}^2 as output.

4.5 The Algorithm

Proof (proof of Theorem 4.1). We prove the correctness and the time complexity as below.

Correctness: Denote BU by \tilde{B} . Assume U is a congruence matrix U between \mathbf{G} and \mathbf{G}' . By Proposition 4.2 we can assume $\sigma = (\tilde{B})^{-1}J_2(\tilde{B})$ after step 3. This time we have $(\mathcal{I}, v, \mathcal{L}')$ is just $(\mathcal{I}_{\mathbf{B}}, v_{\mathbf{B}}, (\tilde{B})^{-1}\mathcal{I}_{\mathbf{B}}v_{\mathbf{B}})$ by Proposition 4.3. Next, by Proposition 4.4 we have $W = \tilde{B}^{-1}\mu(\mathbb{L}) \cdot v_{\mathbf{B}}$, where $\mu(\mathbb{L})$ is the roots of unity contained in \mathbb{L} .

Algorithm 7: FindU

Require: Parameter $\mathbf{B} = (B, (\mathcal{I}_i)_i)$ and \mathbf{K} . An module-LIP $_{\mathbf{K}}^{\mathbf{B}}$ instance $\mathbf{G}' = (G', (\mathcal{J}_i)_i)$.

Ensure: A congruence matrix U between \mathbf{G} (the pseudo-Gram matrix associated with \mathbf{B}) and \mathbf{G}' .

- 1: $P \leftarrow \text{ConjOfJ}(\mathbf{B}, \mathbf{K}, \mathbf{G}')$
- 2: $S \leftarrow \emptyset$
- 3: **for** $\sigma \in P$ **do**
- 4: $(\mathcal{I}, v, \mathcal{L}') \leftarrow \text{EigenSubLat}(\sigma, \mathbf{B}, \mathbf{K}, \mathbf{G}')$
- 5: $W \leftarrow \text{UseLS}(\mathcal{I}, v, \mathcal{L}', \mathbf{B}, \mathbf{K}, \mathbf{G}')$
- 6: **for** $w \in W$ **do**
- 7: $D \leftarrow (v|\bar{v})(w|\bar{w})^{-1}$
- 8: $V \leftarrow B^{-1}D$
- 9: **if** V is a congruence matrix between \mathbf{G} and \mathbf{G}' **then**
- 10: $S \leftarrow S \cup \{V\}$
- 11: **end if**
- 12: **end for**
- 13: **end for**
- 14: **return** S .

Then we assume $w = \tilde{B}^{-1}v$ after Step 6. In this time, $\bar{w} = \overline{\tilde{B}^{-1}v} = \overline{\tilde{B}^{-1}\bar{v}} = \tilde{B}^{-1}\bar{v}$ (note $\tilde{B} \in \mathbb{K}^{2 \times 2}$) and v, \bar{v} are \mathbb{L} -linear independent (note $\langle v, \bar{v} \rangle = 0$). So $(w|\bar{w}) = \tilde{B}^{-1}(v|\bar{v})$ and then $D = \tilde{B}$. Finally, we get $V = B^{-1}\tilde{B} = U$.

Complexity: We can compute matrix products and inverses over \mathbb{K} (resp. \mathbb{L}) in polynomial time. By Proposition 4.2, 4.3, 4.4, Step 1, 4, 5 all run in polynomial time of size $(\mathbf{B}, \mathbf{K}, \mathbf{G}')$. And $\sharp(P) = 2$, $\sharp(W) = \sharp(\mathcal{U}_2(\mathcal{O}_{\mathbb{L}}))$ are in poly(degree (\mathbb{K})). In summary, the whole algorithm runs in polynomial time. \square

Remark 7. We could do argument similar to the proof of Corollary 3.1 to show that: under the additional condition that a hint $\tilde{B}^T \tilde{B}$ is given, Theorem 4.1 still holds for CM number fields.

5 Conclusion

In this paper, we introduce a new tool called (pseudo) symplectic automorphism of the module, with which we can solve $\mathcal{O}_{\mathbb{L}}^2$ -LIP efficiently for a CM number field \mathbb{L} . Although we do not know how to find such automorphism efficiently in general, a weak one can always be computed in polynomial time, which is enough to invalidate the omSVP assumptions utilized in HAWK's security proof and directly results in a provable deterministic polynomial-time algorithm solving module-LIP for rank-2 modules in \mathbb{K}^2 where \mathbb{K} is a totally real number field.

Acknowledgments. Thank the anonymous referees for their valuable suggestions on improving the presentation of this paper. Hengyi Luo and Yanbin Pan were supported in part by National Key Research and Development Project (No. 2020YFA0712300), National Natural Science Foundation of China (No. 62372445,

62032009, 12226006) and Innovation Program for Quantum Science and Technology under Grant 2021ZD0302902. Anyu Wang is partially supported by Tsinghua University Dushi Program.

References

1. Aggarwal, D., Dadush, D., Regev, O., Stephens-Davidowitz, N.: Solving the shortest vector problem in 2^n time using discrete gaussian sampling: Extended abstract. In: Servedio, R.A., Rubinfeld, R. (eds.) Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14–17, 2015. pp. 733–742. ACM (2015), <https://doi.org/10.1145/2746539.2746606>
2. Benčina, B., Budroni, A., Chi-Domínguez, J.J., Kulkarni, M.: Properties of lattice isomorphism as a cryptographic group action. In: Saarinen, M.J., Smith-Tone, D. (eds.) Post-Quantum Cryptography. pp. 170–201. Springer Nature Switzerland, Cham (2024)
3. Bennett, H., Ganju, A., Peetathawatchai, P., Stephens-Davidowitz, N.: Just how hard are rotations of z^n ? algorithms and cryptography with the simplest lattice. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 252–281. Springer (2023)
4. Blanks, T.L., Miller, S.D.: Generating cryptographically-strong random lattice bases and recognizing rotations of Z^n . In: Post-Quantum Cryptography: 12th International Workshop, PQCrypto 2021, Daejeon, South Korea, July 20–22, 2021, Proceedings 12. pp. 319–338. Springer (2021)
5. Cohen, H.: Advanced topics in computational number theory, chap. The Hermite Normal Form Algorithms in Dedekind Domains, p. 27. Springer (2000), <https://api.semanticscholar.org/CorpusID:118279568>
6. Ducas, L.: Provable lattice reduction of z^n with blocksize $n/2$. Designs, Codes and Cryptography pp. 1–8 (2023)
7. Ducas, L., Gibbons, S.: Hull attacks on the lattice isomorphism problem. In: IACR International Conference on Public-Key Cryptography. pp. 177–204. Springer (2023)
8. Ducas, L., Postlethwaite, E.W., Pulles, L.N., van Woerden, W.P.J.: Hawk: Module LIP makes lattice signatures fast, compact and simple. In: Agrawal, S., Lin, D. (eds.) Advances in Cryptology - ASIACRYPT 2022 - 28th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, December 5–9, 2022, Proceedings, Part IV. Lecture Notes in Computer Science, vol. 13794, pp. 65–94. Springer (2022), https://doi.org/10.1007/978-3-031-22972-5_3
9. Ducas, L., van Woerden, W.P.J.: On the lattice isomorphism problem, quadratic forms, remarkable lattices, and cryptography. In: Dunkelman, O., Dziembowski, S. (eds.) Advances in Cryptology - EUROCRYPT 2022 - 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 - June 3, 2022, Proceedings, Part III. Lecture Notes in Computer Science, vol. 13277, pp. 643–673. Springer (2022), https://doi.org/10.1007/978-3-031-07082-2_23
10. Dutour Sikirić, M., Haensch, A., Voight, J., van Woerden, W.P.: A canonical form for positive definite matrices. Open Book Series 4(1), 179–195 (2020)
11. Felderhoff, J., Pellet-Mary, A., Stehlé, D., Wesolowski, B.: Ideal-SVP is Hard for Small-Norm Uniform Prime Ideals. In: Theory of Cryptography, TCC 2023.

- Lecture Notes in Computer Science, vol. 14372, pp. 63–92. Springer Nature Switzerland, Taipei (Taiwan), Taiwan (Dec 2023). https://doi.org/10.1007/978-3-031-48624-1_3, <https://hal.science/hal-04326750>
12. Geißler, K., Smart, N.P.: Computing the $M = U U^t$ integer matrix decomposition. In: Paterson, K.G. (ed.) *Cryptography and Coding, 9th IMA International Conference, Cirencester, UK, December 16-18, 2003, Proceedings*. Lecture Notes in Computer Science, vol. 2898, pp. 223–233. Springer (2003), https://doi.org/10.1007/978-3-540-40974-8_18
 13. Gentry, C., Szydło, M.: Cryptanalysis of the revised NTRU signature scheme. In: Knudsen, L.R. (ed.) *Advances in Cryptology - EUROCRYPT 2002, International Conference on the Theory and Applications of Cryptographic Techniques, Amsterdam, The Netherlands, April 28 - May 2, 2002, Proceedings*. Lecture Notes in Computer Science, vol. 2332, pp. 299–320. Springer (2002), https://doi.org/10.1007/3-540-46035-7_20
 14. Haviv, I., Regev, O.: On the lattice isomorphism problem. In: Chekuri, C. (ed.) *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*. pp. 391–404. SIAM (2014), <https://doi.org/10.1137/1.9781611973402.29>
 15. Hoffstein, J., Howgrave-Graham, N., Pipher, J., Silverman, J.H., Whyte, W.: NTRUSIGN: digital signatures using the NTRU lattice. In: Joye, M. (ed.) *Topics in Cryptology - CT-RSA 2003, The Cryptographers' Track at the RSA Conference 2003, San Francisco, CA, USA, April 13-17, 2003, Proceedings*. Lecture Notes in Computer Science, vol. 2612, pp. 122–140. Springer (2003), https://doi.org/10.1007/3-540-36563-X_9
 16. Jiang, K., Wang, A., Luo, H., Liu, G., Yu, Y., Wang, X.: Exploiting the symmetry of \mathbb{Z}^n : Randomization and the automorphism problem. In: Guo, J., Steinfeld, R. (eds.) *Advances in Cryptology – ASIACRYPT 2023*. pp. 167–200. Springer Nature Singapore, Singapore (2023)
 17. Jr., H.W.L., Silverberg, A.: Revisiting the gentry-szydło algorithm. In: Garay, J.A., Gennaro, R. (eds.) *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I*. Lecture Notes in Computer Science, vol. 8616, pp. 280–296. Springer (2014), https://doi.org/10.1007/978-3-662-44371-2_16
 18. Jr., H.W.L., Silverberg, A.: Lattices with symmetry. *J. Cryptol.* **30**(3), 760–804 (2017), <https://doi.org/10.1007/s00145-016-9235-7>
 19. Lenstra Jr, H.W., Silverberg, A.: Testing isomorphism of lattices over cm-orders. *SIAM Journal on Computing* **48**(4), 1300–1334 (2019)
 20. Mureau, G., Pellet-Mary, A., Pliatsok, G., Wallet, A.: Cryptanalysis of rank-2 module-lip in totally real number fields. In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. pp. 226–255. Springer (2024)
 21. NIST: Post-quantum cryptography: digital signature schemes. round 1 additional signatures (2023)
 22. Plesken, W., Souvignier, B.: Computing isometries of lattices. *Journal of Symbolic Computation* **24**(3-4), 327–334 (1997)
 23. Szydło, M.: Hypercubic lattice reduction and analysis of GGH and NTRU signatures. In: Biham, E. (ed.) *Advances in Cryptology - EUROCRYPT 2003, International Conference on the Theory and Applications of Cryptographic Techniques, Warsaw, Poland, May 4-8, 2003, Proceedings*. Lecture Notes in Computer Science, vol. 2656, pp. 433–448. Springer (2003), https://doi.org/10.1007/3-540-39200-9_27



Dense and Smooth Lattices in Any Genus

Wessel van Woerden^(✉) 

Univ. Bordeaux, CNRS, Inria, Bordeaux INP, IMB, Talence, France
wessel.van-woerden@math.u-bordeaux.fr

Abstract. The Lattice Isomorphism Problem (LIP) was recently introduced as a new hardness assumption for post-quantum cryptography. The strongest known efficiently computable invariant for LIP is the genus of a lattice. To instantiate LIP-based schemes one often requires the existence of a lattice that (1) lies in some fixed genus, and (2) has some good geometric properties such as a high packing density or small smoothness parameter.

In this work we show that such lattices exist. In particular, building upon classical results by Siegel (1935), we show that essentially any genus contains a lattice with a close to optimal packing density, smoothing parameter and covering radius. We present both how to efficiently compute concrete existence bounds for any genus, and asymptotically tight bounds under weak conditions on the genus.

The introduction of the lattice isomorphism problem (LIP) as a hardness assumption for cryptography raises a new family of interesting questions [3, 10, 11]. LIP asks to determine if two lattices are isomorphic, i.e., if one is an orthonormal transformation of the other. One way to answer this question in the negative is using invariants, and the *genus* of a lattice, gives the strongest known efficiently computable invariant for LIP. If two lattices fall into distinct genera LIP thus becomes easy. Therefore, in the context of LIP, one often works inside a certain genus or a family of genera. In particular, notions like randomness, reductions and hardness questions are suddenly restricted to within a genus.

In this work we study the geometric properties of random lattices in a fixed genus \mathcal{G} and use that to show the existence of a lattice $\mathcal{L} \in \mathcal{G}$ with a good packing density, smoothing parameter or covering radius. We show that the strong condition of being in a fixed genus does in fact not change much to the behavior we are used to from random lattices. This is both interesting from a theoretic perspective, but in addition it allows to tightly instantiate cryptographic schemes that are based on LIP. Previously, the existence of such lattices was assumed heuristically [1, 3, 11].

Random Lattices and the Existence of Good Packings. Within cryptography families of random lattices play an important role, for example we often consider random q -ary lattices $q\mathbb{Z}^n \subset \mathcal{L} \subset \mathbb{Z}^n$ which are related to LWE, NTRU or SIS. Within cryptanalysis random lattices often play the role of worst-case instances and therefore their properties are used in heuristic analysis of algorithms. For

example, heuristically we assume that lattices we encounter follow the Gaussian Heuristic which in full generality says that the number of nonzero lattice points in a ‘nice’ volume S is about $\text{vol}(S)/\text{vol}(\mathcal{L})$. What is precisely meant here by ‘random’ is often not so important as these heuristics give good estimates in practice.

On the mathematical side however, the notion of a random lattice is more explicitly defined. Here we look at the whole space of lattices $\mathcal{L}_{[n,D]}$ of some fixed dimension $n \geq 2$ and (co)volume $D > 0$. There exists a natural and finite Haar measure on $\mathcal{L}_{[n,D]}$, which thereby induces a natural probability distribution $\mathcal{D}(\mathcal{L}_{[n,D]})$. While the space $\mathcal{L}_{[n,D]}$ and the distribution $\mathcal{D}(\mathcal{L}_{[n,D]})$ can be quite complicated to understand, it allows for remarkably clean and provable statements about the expected behavior of lattices following this distribution. For example, for any star-shaped volume S the expected number $\mathbb{E}[|S \cap \mathcal{L} \setminus \{0\}|]$ of nonzero lattice points in S over $\mathcal{D}(\mathcal{L}_{[n,D]})$ is precisely equal to $\text{vol}(S)/\text{vol}(\mathcal{L})$, i.e., what the Gaussian Heuristic prescribes. However, in this case it is a provable result.

Now by choosing $S \subset \mathbb{R}^n$ to be a ball of radius λ we can determine the expected number of nonzero lattice points of length at most λ . If this expectation is strictly less than 2 we know that there must exist a lattice $\mathcal{L} \in \mathcal{L}_{[n,D]}$ that has strictly less than 2 vectors of length λ . Because any lattice point occurs in a pair $\pm x$ of the same length we thus know that this lattice contains no lattice points of length at most λ and thus that its minimum distance $\lambda_1(\mathcal{L}) := \min\{\|v\| : v \in \mathcal{L}\}$ satisfies $\lambda_1(\mathcal{L}) > \lambda$. By picking the appropriate λ and by slightly refining this argument one gets the Minkowski-Hlawka Theorem, which says that there exists a lattice $\mathcal{L} \subset \mathbb{R}^n$ with $\lambda_1(\mathcal{L}) \geq (2\zeta(n)\text{vol}(\mathcal{L})/\omega_n)^{1/n} \approx \sqrt{n/2\pi e} \cdot \text{vol}(\mathcal{L})^{1/n}$, where ω_n is the volume of the n -dimensional unit ball. This is close to optimal as Minkowski’s Theorem says that $\lambda_1(\mathcal{L}) \leq \text{mk}(\mathcal{L}) := 2 \cdot (\text{vol}(\mathcal{L})/\omega_n)^{1/n}$.

So from the average-case behavior of random lattices one can show the existence of a lattice with a large minimum distance, i.e., that of a good lattice packing. More generally, random lattices are known to have other good geometric properties. Besides a good packing density, they also have a large covering radius and a small smoothing parameter in expectation. And again, this immediately results in a proof of existence for lattices with such good properties.

Random Lattices in a Fixed Genus. Our question is if the same can be said when we add the seemingly strong restriction of falling in some fixed genus. Two (full-rank) integral lattices $\mathcal{L}_1, \mathcal{L}_2 \subset \mathbb{R}^n$ fall into the same genus if they are equivalent over the p -adic integers \mathbb{Z}_p for all primes p , i.e., if $\mathcal{L}_1 \otimes_{\mathbb{Z}} \mathbb{Z}_p \cong \mathcal{L}_2 \otimes_{\mathbb{Z}} \mathbb{Z}_p$ ¹ Computing if two lattices are equivalent over \mathbb{Z}_p is efficient, essentially because one can (block) diagonalize (gram) matrices over this local ring, and the equivalence class can then simply be read from the (block) diagonalized form. Furthermore,

¹ One could view this as formally replacing the lattice $B \cdot \mathbb{Z}^n$ by $B \cdot \mathbb{Z}_p^n$, and the standard Euclidean inner product with image \mathbb{Z} becomes the bi-linear form $(x, y) \mapsto \sum_i x_i y_i$ with image \mathbb{Z}_p . An isomorphism has to preserve both the \mathbb{Z}_p structure as the bi-linear form.

assuming that $\text{vol}(\mathcal{L}_1) = \text{vol}(\mathcal{L}_2)$, one only has to check this equivalence over the finite number of primes p dividing $2 \text{vol}(\mathcal{L}_i)^2$. Given the prime factorization of $\text{vol}(\mathcal{L}_i)^2$ the genus equivalence is thus efficiently computable.

Minkowski showed that any genus only contains a finite number of isomorphism classes $[\mathcal{L}_1], \dots, [\mathcal{L}_m]$ [20]. Furthermore, if one restrict the usual Haar measure to a fixed genus \mathcal{G} we obtain a natural distribution $\mathcal{D}(\mathcal{G})$ on these classes, where each $[\mathcal{L}_i]$ is sampled relative to its mass $m([\mathcal{L}_i]) = 1/|\text{Aut}(\mathcal{L}_i)|$, where $\text{Aut}(\mathcal{L}_i)$ is the automorphism group of \mathcal{L}_i .

Now that we have a natural notion of randomness on a genus \mathcal{G} , the question is if we can say something about the expected behavior of certain lattice properties. It turns out that the answer is yes, and in fact most of the theory for this was already developed almost 90 years ago by Siegel [26] in the form of *mass formulas*.

For an integer $k \geq 1$ and an integral lattice \mathcal{L} we denote the number of lattice point with squared norm k by $N_{\mathcal{L}}(k) := |\{x \in \mathcal{L} : \|x\|^2 = k\}|$. Generally, computing $N_{\mathcal{L}}(k)$ is a very hard problem, however Siegel showed that its expected value over a genus is essentially equal to a converging product of local densities at each prime p . Each local density is efficiently computable and similarly as for the genus one only really has to compute them for the primes p dividing $2k \text{vol}(\mathcal{G})^2$. Siegel’s mass formula thus implies, that for a genus \mathcal{G} we can efficiently compute the expectation $N_{\mathcal{G}}(k) := \mathbb{E}_{[\mathcal{L}] \leftarrow \mathcal{D}(\mathcal{G})}[N_{\mathcal{L}}(k)]$ (given the prime factorization of $2k \text{vol}(\mathcal{G})^2$).

We can now make a similar argument as for the Minkowski-Hlawka Theorem. For an integer $\lambda \geq 1$ consider the sum $S_{\lambda} := \sum_{k=1}^{\lambda} N_{\mathcal{G}}(k)$. This sum represents the expected number of nonzero lattice vectors of squared norm at most λ for a lattice \mathcal{L} sampled from $\mathcal{D}(\mathcal{G})$. Now if $S_{\lambda} < 2$ we know that there must exist a lattice $\mathcal{L} \in \mathcal{G}$ with strictly less than 2 and thus precisely 0 nonzero vectors of squared norm less than λ . So we have that $\lambda_1(\mathcal{L})^2 > \lambda$, and by appropriately picking λ this can show the existence of a good lattice packing in \mathcal{G} . This reasoning was already used in an unpublished work by Conway and Thompson, and written down by Milnor [19], to show the existence of odd unimodular lattices, integral lattices \mathcal{L} with volume 1 which contain vectors of odd squared norm, with $\lambda_1(\mathcal{L})^2 \geq \lfloor (\frac{3}{5}\omega_n)^{-2/n} \rfloor$. For the case of even unimodular lattices, that only exist when $8|n$, Milnor [19], using computations of Serre [25], claims a similar bound of $\lambda_1(\mathcal{L})^2 \geq 2 \lfloor \frac{1}{2} (\frac{3}{5}\omega_n)^{-2/n} \rfloor$. Note that for both odd and even unimodular lattices the existence bound is only slightly weaker than the Minkowski-Hlawka Theorem, and they quickly converge to each other for large n . The additional restriction of falling in a fixed genus therefore does not seem strong enough to influence the good geometric properties of random lattices too much.

Contributions. The first aim of this work is to survey these classical and maybe surprising results related to the genus. The existing literature however only seems to consider the unimodular case and the packing density². In this work, we therefore extend these results in two ways that are of interest to cryptography.

² As far as we know.

Firstly, we extend the Minkowski-Hlawka-like Theorem to almost any genus. In particular, under light conditions³ on the genus \mathcal{G} , we show the existence of a good lattice packing $\mathcal{L} \in \mathcal{G}$, with minimum distance equivalent to the Minkowski-Hlawka Theorem up to a small factor $O(1)^{1/n}$. We achieve this by bounding the local densities and thus the expected number of lattice vectors $N_{\mathcal{G}}(k)$ of squared norm k .

Theorem 1 (General packing). *For any integral genus \mathcal{G} in dimension $n \geq 6$ such that $\text{rk}_p(\mathcal{G}) \geq 6$ for all primes p , and any constant $0 < c \leq 1$, we have*

$$\Pr_{[\mathcal{L}] \leftarrow \mathcal{D}(\mathcal{G})} \left[\lambda_1(\mathcal{L})^2 \geq \left[c^2 \cdot \left(\frac{7\zeta(3)}{9\zeta(2)} \cdot \frac{\text{vol}(\mathcal{L})}{\omega_n} \right)^{2/n} \right] \right] > 1 - c^n.$$

In particular, there exists a lattice $\mathcal{L} \in \mathcal{G}$ with

$$\lambda_1(\mathcal{L})^2 \geq \left[\left(\frac{7\zeta(3)}{9\zeta(2)} \cdot \frac{\text{vol}(\mathcal{L})}{\omega_n} \right)^{2/n} \right] \approx n/2\pi e \cdot \text{vol}(\mathcal{L})^{2/n}.$$

Note that in fact we show something stronger, i.e., by roughly lowering the bound on the first minimum by a constant factor c we show that it is attained with a probability of at least $1 - c^n$ over $\mathcal{D}(\mathcal{G})$. This follows directly from an application of Markov’s inequality in the proof and the result closely matches the behavior of the first minimum for random lattices [2]. Furthermore, this allows us to show the existence of a lattice $\mathcal{L} \in \mathcal{G}$ for which both \mathcal{L} and its dual \mathcal{L}^* have a good packing density.

Secondly, we show that the reasoning can be extended to prove the existence of lattices with a good covering radius $v(\mathcal{L}) := \min\{\lambda > 0 : \text{dist}(\mathcal{L}, x) \leq \lambda \forall x \in \mathbb{R}^n\}$ and a good smoothing parameter $\eta_\epsilon(\mathcal{L})$, even for the relatively large values $\epsilon \gg e^{-n}$ that are of interest in cryptography.

Theorem 2 (General smoothing). *For any integral genus \mathcal{G} in dimension $n \geq 6$ such that $\text{rk}_p(\mathcal{G}) \geq 6$ for all primes p , constants $C = 26.1$ and $0 < c \leq 1$, and $\epsilon \geq C \cdot (ce)^{-n} \cdot \text{vol}(\mathcal{G})^{-1}$, we have*

$$\Pr_{[\mathcal{L}] \leftarrow \mathcal{D}(\mathcal{G})} \left[\eta_\epsilon(\mathcal{L}^*) \leq \frac{1}{c} \cdot \left(\frac{C \cdot \text{vol}(\mathcal{L}^*)}{\epsilon} \right)^{1/n} \right] > 1 - c^n.$$

In particular, there exists a lattice $\mathcal{L} \in \mathcal{G}$ such that $\eta_\epsilon(\mathcal{L}^) \leq (C \cdot \text{vol}(\mathcal{L}^*)/\epsilon)^{1/n}$.*

³ We require that the rank $\text{rk}_p(\mathcal{G})$ of a Gram matrix $G \bmod p$ over \mathbb{F}_p of any lattice $\mathcal{L} \in \mathcal{G}$ is at least 6. Note that this property only has to be checked for primes $p | \det(G)$, and is (after normalization) true for most integral lattices of sufficiently large dimension. In particular it is true for \mathbb{Z}^n , SIS, LWE and NTRU lattices with a sufficiently high dimension and number of samples.

Theorem 3 (General covering radius). *For any integral genus \mathcal{G} in dimension $n \geq 6$ such that $\text{rk}_p(\mathcal{G}) \geq 6$ for all primes p , and constants $C = 26.1$ and $e^{-1}(2C/(3 \text{vol}(\mathcal{G})))^{1/n} \leq c \leq 1$, we have*

$$\Pr_{[\mathcal{L}] \sim \mathcal{D}(\mathcal{G})} \left[v(\mathcal{L}^*) \leq \frac{1}{c} \cdot \left(\sqrt{n/2\pi} + 1 \right) \cdot \left(\frac{2}{3} C \cdot \text{vol}(\mathcal{L}^*) \right)^{1/n} \right] > 1 - c^n.$$

In particular, when additionally $n \geq 7$, there exists a lattice $\mathcal{L} \in \mathcal{G}$ such that

$$v(\mathcal{L}^*) \leq \left(\sqrt{n/2\pi} + 1 \right) \cdot \left(\frac{2}{3} C \cdot \text{vol}(\mathcal{L}^*) \right)^{1/n} \approx \sqrt{e} \cdot \sqrt{n/2\pi e} \cdot \text{vol}(\mathcal{L}^*)^{1/n}.$$

Besides these essentially tight asymptotic bounds we also explain how to efficiently compute concrete existence bounds for any fixed genus.

Applications. Finally, we give some applications of these results for the instantiation of LIP-based schemes. Suppose we have an efficiently decodable lattice \mathcal{L} with unique decoding radius $\rho = \Theta(\lambda_1(\mathcal{L}))$, and suppose that

$$\text{gap}(\mathcal{L}) := \max\{\text{mk}(\mathcal{L})/\lambda_1(\mathcal{L}), \text{mk}(\mathcal{L}^*)/\lambda_1(\mathcal{L}^*)\} \leq f,$$

i.e., the primal and dual minimum distances and the decoding radius are within a factor $O(f)$ from optimal. Heuristically, the larger f is the easier it is to decode or find short vectors in this lattice (or its dual). Given such a lattice [11] shows how to instantiate an encryption scheme where the security is solely based on distinguishing between some isomorphism classes $[\mathcal{L}_1], [\mathcal{L}_2]$ constructed from \mathcal{L} that lie in the same genus. However in this construction the geometric gaps blow up to $\text{gap}(\mathcal{L}_i) = O(f^3)$ which reduces the concrete security significantly. Our results show the existence of a lattice \mathcal{L}' in the same genus as \mathcal{L} but such that $\text{gap}(\mathcal{L}') = O(1)$. This can in turn be used to create a suitable pair $\mathcal{L}_1, \mathcal{L}_2$ for which $\text{gap}(\mathcal{L}_i) = O(f)$, and thus we reduce the cubic loss to only a small constant loss. The encryption scheme from [4] based on the same framework benefits from the same improvement. Similarly, we show how to instantiate the signature scheme from [11] with a constant loss $O(f)$ instead of a quadratic loss $O(f^2)$.

Another interesting work [3] introduces constructions based on LIP for the unimodular lattice \mathbb{Z}^n . To instantiate their scheme the authors assume that there exists a lattice \mathcal{L} in the odd unimodular genus \mathcal{G}_{odd} of \mathbb{Z}^n with $\lambda_1(\mathcal{L}) \geq \Omega(n/\log(n))$ and $\eta_\varepsilon(\mathbb{Z}^n) \leq \eta_\varepsilon(\mathcal{L})/\sqrt{\log(n)}$ for $\varepsilon < n^{-\omega(1)n}$. Similarly, the encryption scheme [1] based on LIP requires the existence of an even unimodular lattice \mathcal{L} such that $\lambda_1(\mathcal{L}) \geq \sqrt[4]{72n}$, and this is conjectured to be true for $n \geq 85$. We raise that the first claim for the first minimum is already answered by [19, 25], and the second claim for the smoothing parameter follows from Lemma 4. In fact, this shows a much stronger result than required.

1 Preliminaries

Notation. Vectors are *column vectors*. For a ring R we denote $\mathcal{GL}_n(R)$ as the general linear group of $n \times n$ invertible matrices over R . For $R \subset \mathbb{R}$ we denote $\mathcal{S}_n^{>0}(R)$ as the space of positive-definite symmetric matrices over R . We denote $\mathcal{O}_n(\mathbb{R})$ for the group of orthonormal transformations over the reals \mathbb{R} . We denote $\zeta(\cdot)$ for the Riemann zeta function.

1.1 Lattices and Quadratic Forms

Lattices. A lattice \mathcal{L} is a discrete additive subgroup of the Euclidean space \mathbb{R}^n . We call the dimension of the real span $\text{Span}_{\mathbb{R}}(\mathcal{L}) \subset \mathbb{R}^n$ the *rank* $\text{rk}(\mathcal{L})$ of a lattice, and say that $\mathcal{L} \subset \mathbb{R}^n$ has *full-rank* if $\text{rk}(\mathcal{L}) = n$. In this work we restrict ourselves to full-rank lattices. Full-rank lattices $\mathcal{L} \subset \mathbb{R}^n$ can be represented by a full-rank basis $B \in \mathcal{GL}_n(\mathbb{R})$ such that

$$\mathcal{L} = \mathcal{L}(B) := B \cdot \mathbb{Z}^n = \{Bx : x \in \mathbb{Z}^n\}.$$

Such a basis representation is not unique, i.e., for any basis $B \in \mathcal{GL}_n(\mathbb{R})$ and any unimodular matrix $U \in \mathcal{GL}_n(\mathbb{Z})$ we have $\mathcal{L}(B) = \mathcal{L}(B \cdot U)$.

For a basis B we call $G_B := B^T B \in \mathcal{S}_n^{>0}(\mathbb{R})$ the *gram* matrix of B and a gram matrix of the lattice $\mathcal{L}(B)$. Note that a gram matrix does not uniquely define a lattice, in particular for any basis $B \in \mathcal{GL}_n(\mathbb{R})$ and any orthonormal transformation $O \in \mathcal{O}_n(\mathbb{R})$ we have $G_B = G_{OB}$ while $\mathcal{L}(B)$ and $\mathcal{L}(OB)$ are usually distinct. From a gram matrix $G \in \mathcal{S}_n^{>0}(\mathbb{R})$ one can always construct a corresponding lattice basis by computing the unique Cholesky decomposition $G = C^T C$ where C is an upper-triangular matrix with positive diagonal. Generally however, the Cholesky decomposition of G_B does not return the basis B of $\mathcal{L}(B)$, but some basis $C = O \cdot B$ of $O \cdot \mathcal{L}(B)$ for some $O \in \mathcal{O}_n(\mathbb{R})$.

For a lattice \mathcal{L} we write \mathcal{L}^* for its *dual* lattice given by

$$\mathcal{L}^* := \{x \in \mathbb{R}^n : \forall y \in \mathcal{L}, \langle x, y \rangle \in \mathbb{Z}\}.$$

As expected we have that $(\mathcal{L}^*)^* = \mathcal{L}$. If B is a basis and G a gram matrix of \mathcal{L} , then $(B^{-1})^T$ is a basis and G^{-1} a gram matrix of \mathcal{L}^* .

Lattice Properties. Due to the discrete and additive nature of a lattice there exists a positive minimum (Euclidean) distance $\lambda_1(\mathcal{L})$ called the *first minimum* between any two distinct lattice points. Equivalently, this can be defined as $\lambda_1(\mathcal{L}) := \min_{x \in \mathcal{L} \setminus \{0\}} \|x\|$. For a (full-rank) lattice $\mathcal{L} = \mathcal{L}(B)$ we define its *(co)volume* $\text{vol}(\mathcal{L})$ as $|\det(B)|$ which is independent of the chosen basis. Equivalently, we have $\text{vol}(\mathcal{L}) = \det(G)^{\frac{1}{2}}$ for any gram matrix $G \in \mathcal{S}_n^{>0}(\mathbb{R})$ of \mathcal{L} . Furthermore, note that $\text{vol}(\mathcal{L}^*) = \text{vol}(\mathcal{L})^{-1}$. The first minimum and the volume of a lattice $\mathcal{L} \subset \mathbb{R}^n$ are related to each-other by Minkowski's Theorem which says that

$$\lambda_1(\mathcal{L}) \leq \text{mk}(\mathcal{L}) := 2 \cdot \frac{\text{vol}(\mathcal{L})^{1/n}}{\omega_n^{1/n}} \approx \sqrt{2n/\pi e} \cdot \text{vol}(\mathcal{L})^{1/n},$$

where ω_n is the volume of the n -dimensional unit ball. The *covering radius* $v(\mathcal{L})$ of a (full-rank) lattice $\mathcal{L} \subset \mathbb{R}^n$ is the minimum radius $r > 0$ such that any target $t \in \mathbb{R}^n$ is at distance at most r from the lattice, i.e., such that $\mathcal{L} + r\mathcal{B}^n = \mathbb{R}^n$.

We call a lattice *integral* or *rational*, if all for all pair-wise $x, y \in \mathcal{L}$ the inner product $\langle x, y \rangle \in \mathbb{Z}$ is integer or rational, respectively. Equivalently this means that for any basis B of \mathcal{L} the gram matrix $G_B = B^\top B$ has integer or rational coefficients, i.e., $G_B \in \mathcal{S}_n^{>0}(\mathbb{Z})$ or $G_B \in \mathcal{S}_n^{>0}(\mathbb{Q})$, respectively. Note that a lattice being *integral* is a weaker condition than being *integer* $\mathcal{L} \subset \mathbb{Z}^n$, and many well-known lattices are integral but not integer. We define the *scale* of a rational lattice \mathcal{L} by $\text{scale}(\mathcal{L}) := \max\{0 < s < \infty : \frac{1}{s}\mathcal{L} \text{ is integral}\}$, which can efficiently be computed from any gram matrix G of \mathcal{L} . We call an integral lattice \mathcal{L} *normalized* if $\text{scale}(\mathcal{L}) = 1$. Note that every rational lattice can be normalized to an integral lattice as $\mathcal{L}/\text{scale}(\mathcal{L})$. If a lattice \mathcal{L} is rational, then its dual is also rational and thus integral up to scaling. We define the *parity* of an integral lattice \mathcal{L} by $\text{par}(\mathcal{L}) := \gcd(\{\|x\|^2 : x \in \mathcal{L}\}) / \gcd(\{\langle x, y \rangle : x, y \in \mathcal{L}\}) \in \{1, 2\}$, which can efficiently be computed from any gram matrix G of \mathcal{L} . For an integral lattice $\mathcal{L} \subset \mathbb{R}^n$ with gram matrix $G \in \mathcal{S}_n^{>0}(\mathbb{Z})$ and any prime p we define its p -rank by $\text{rk}_p(\mathcal{L}) := \text{rk}_{\mathbb{F}_p}(G)$, which is independent of the choice of gram matrix.

Besides the first minimum $\lambda_1(\mathcal{L})$ of a lattice we can also ask how many lattice vectors exists of a certain length. This information is usually denoted by the theta series of a lattice.

Definition 1 (Theta series). *Let \mathcal{L} be a lattice, the theta series $\theta_{\mathcal{L}}(q)$ of \mathcal{L} is the formal q -series*

$$\theta_{\mathcal{L}}(q) = \sum_{v \in \mathcal{L}} q^{\|v\|^2}.$$

For integral lattices \mathcal{L} we obtain the formal power series $\theta_{\mathcal{L}}(q) = 1 + \sum_{k=1}^{\infty} N_{\mathcal{L}}(k) \cdot q^k$, where $N_{\mathcal{L}}(k) := |\{v \in \mathcal{L} : \|v\|^2 = k\}|$ is the number of vectors with squared norm k .

Note that for an integral lattice \mathcal{L} we have $N_{\mathcal{L}}(k) = 0$ for all $0 < k < \lambda_1(\mathcal{L})^2$, i.e., the first $\lambda_1(\mathcal{L})^2 - 1$ non-trivial coefficients of the theta series are 0. Theta series and their relation to the first minimum of a lattice will play an important role in this work. Another property that is important in lattice-based cryptography is the smoothing parameter.

Definition 2 (smoothing parameter). *For $\varepsilon > 0$ the smoothing parameter $\eta_{\varepsilon}(\mathcal{L})$ of a lattice \mathcal{L} is given by the minimum $s > 0$ such that $\theta_{\mathcal{L}^*}(\exp(-\pi s^2)) = 1 + \varepsilon$.*

While this might not immediately be clear from the definition, the smoothing parameter indicates how large the standard deviation of a centered (continuous) Gaussian must be such that it becomes ε -close to uniform over the quotient \mathbb{R}^n/\mathcal{L} . The latter property can for example be used in security proofs of signature schemes to show that the signatures sampled from a discrete Gaussian with standard deviation $\sigma \geq \eta_{\varepsilon}(\mathcal{L})$ do not leak any information. Preferably we thus want the smoothing parameter $\eta_{\varepsilon}(\mathcal{L})$ to be small, something which we informally

call *good smoothing*. A lattice with a good smoothing automatically also has a small covering radius.

Lemma 1 ([24, Lemma 6.1]). *For any lattice $\mathcal{L} \subset \mathbb{R}^n$ we have*

$$v(\mathcal{L}) \leq \left(\sqrt{n/2\pi} + 1\right) \cdot \eta_{\frac{3}{2}}(\mathcal{L}).$$

Note that computing the first minimum, (part of) the theta series, the covering radius, or the smoothing parameter is generally a hard problem for which the best algorithms take at least $2^{\Omega(n)}$ time.

Quadratic Forms. In the literature on post-quantum cryptography and cryptanalysis it is common to work with bases and lattices. On the contrary, in the mathematical study of lattices it is quite common to work with gram matrices and (positive definite) quadratic forms. We discuss how those are related and how they essentially give a different view on the same object.

Consider a basis $B \in \mathcal{GL}_n(\mathbb{R})$ and its gram matrix $G = B^\top B$. A gram matrix is positive definite and naturally defines a positive definite real quadratic form

$$f_G : \mathbb{R}^n \rightarrow \mathbb{R}, \quad x \mapsto x^\top Gx = \sum_{i=1}^n \sum_{j=1}^n G_{ij}x_i x_j.$$

From now on we will simply identify f_G with G and call a gram matrix G a *quadratic form* or simply a *form*. Due to the positive-definiteness such a quadratic form defines a norm by $\|x\|_G := \sqrt{x^\top Gx}$. Note that for $v = Bx$ we have the following identity:

$$\|v\|_2^2 = (Bx)^\top Bx = x^\top B^\top Bx = x^\top Gx =: \|x\|_G^2.$$

More generally, G defines an inner product $\langle x, y \rangle_G := x^\top Gy$, and for Bx, By we have that

$$\langle Bx, By \rangle = (Bx)^\top By = x^\top B^\top By = x^\top Gy = \langle x, y \rangle_G.$$

In terms of geometry it is thus equivalent to consider the vector Bx under the Euclidean geometry or the vector x under the geometry induced by G . Every lattice point of $\mathcal{L}(B)$ can be written as Bx for an integer vector $x \in \mathbb{Z}^n$, thus on the quadratic form side we always consider the lattice \mathbb{Z}^n (but we change its geometry). For a quadratic form G one could thus similarly define its *first minimum* by $\lambda_1(G) = \min_{x \in \mathbb{Z}^n \setminus \{0\}} \|x\|_G$, or its (co)volume $\text{vol}(G) := \sqrt{\det(G)}$ matching those of any corresponding lattice.

Throughout this work when we talk about a lattice we always implicitly assume it is represented by some basis or some gram matrix. Generally, we will stick to the lattice terminology to present our main results, but for the proofs we will often switch to quadratic forms as they are more natural to work with in this setting.

1.2 Random Lattices

The space of full-rank lattices in \mathbb{R}^n of volume 1 can be identified by the quotient $\mathcal{L}_{[n]} := \mathcal{GL}_n(\mathbb{R})/\mathcal{GL}_n(\mathbb{Z})$. Here $\mathcal{GL}_n(\mathbb{R})$ represents all the bases of volume 1, and $\mathcal{GL}_n(\mathbb{Z})$ represents the basis transformations that turn one basis into another basis of the same lattice.

The space $\mathcal{GL}_n(\mathbb{R})$ has a natural invariant Haar measure, and Siegel proved in 1945 [27] that the mass of $\mathcal{L}_{[n]}$ is finite under the projection of this Haar measure. After normalization this yields a probability distribution μ_n over $\mathcal{L}_{[n]}$. By construction this distribution is invariant under both orthonormal and basis transformations, i.e., for any measurable set $\mathcal{A} \subset \mathcal{L}_{[n]}$ and all $O \in \mathcal{GL}_n(\mathbb{R})$, and $U \in \mathcal{GL}_n(\mathbb{Z})$ we have $\mu_n(OAU) = \mu_n(\mathcal{A})$. A *random lattice* is thus a unit lattice $\mathcal{L} \in \mathcal{L}_{[n]}$ sampled under the probability distribution μ_n . More generally, simply by scaling, we also speak of random lattices of some fixed volume $D > 0$.

In 1943, Hlawka proved the following, maybe surprising result about the expectation of a function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ over a random lattice.

Theorem 4 ([17,27]). *Let $n \geq 2$ and let $f : \mathbb{R}^n \rightarrow \mathbb{R}$ be an Riemann-integrable function such that $\|x\|^{n+c} f(x)$ is bounded on \mathbb{R}^n for some fixed $c > 0$. Then*

$$\int_{\mathcal{L} \in \mathcal{L}_{[n]}} \sum_{x \in \mathcal{L} \setminus \{0\}} f(x) d\mu_n = \int_{\mathbb{R}^n} f(x) dx.$$

In particular, for a star-shaped volume S the expected number of nonzero lattice vectors in S for a random lattice of volume D is $\text{vol}(S)/D$. Furthermore, the expected number of primitive lattice vectors is $\frac{\text{vol}(S)}{\zeta(n)D}$.

Knowing the expected number of (primitive) lattice points in a certain volume is enough to show the existence of a lattice with a good packing.

Corollary 1 (Minkowski-Hlawka theorem [17,21]). *For any dimension n and volume D there exists a lattice $\mathcal{L} \in \mathcal{L}_{[n,D]}$ with*

$$\lambda_1(\mathcal{L}) \geq (2\zeta(n) \text{vol}(\mathcal{L})/\omega_n)^{1/n} \approx \sqrt{n/2\pi e} \cdot \text{vol}(\mathcal{L})^{1/n}.$$

Proof. For any $(2\zeta(n)D/\omega_n)^{1/n} > \varepsilon > 0$ let $\lambda = (2\zeta(n)D/\omega_n)^{1/n} - \varepsilon > 0$ and let $S_\lambda \subset \mathbb{R}^n$ be the n -dimensional ball with radius λ . Then by construction

$$\frac{\text{vol}(S_\lambda)}{\zeta(n)D} = \frac{\lambda^n \cdot \omega_n}{\zeta(n)D} < 2,$$

and thus the expected number of primitive lattice vectors in S_λ is strictly less than 2. There thus exists a lattice $\mathcal{L} \in \mathcal{L}_{[n,D]}$ such that $|S_\lambda \cap \mathcal{L}| < 2$ which implies that $|S_\lambda \cap \mathcal{L}| = 0$ as any lattice vectors occurs as a pair $\pm x$. So $\lambda_1(\mathcal{L}) > (2\zeta(n) \text{vol}(\mathcal{L})/\omega_n)^{1/n} - \varepsilon$. In particular, letting $\varepsilon \rightarrow 0$ shows that

$$\sup_{\mathcal{L} \in \mathcal{L}_{[n,D]}} \lambda_1(\mathcal{L}) \geq (2\zeta(n) \text{vol}(\mathcal{L})/\omega_n)^{1/n}.$$

It is a classical result (see e.g. [29, p. 29-31]) that this supremum is attained by some lattice $\mathcal{L} \in \mathcal{L}_{[n,D]}$, from which the Theorem follows. □

Ignoring the small factor⁴ $(2\zeta(n))^{1/n}$ the quantity $\text{gh}(\mathcal{L}) := (\text{vol}(\mathcal{L})/\omega_n)^{1/n} \approx \sqrt{n}/2\pi e \cdot \text{vol}(\mathcal{L})^{1/n}$ is often called the *Gaussian heuristic* of a lattice \mathcal{L} . Besides the existence of a lattice $\mathcal{L} \subset \mathbb{R}^n$ with $\lambda_1(\mathcal{L}) \geq \text{gh}(\mathcal{L})$ one can also show concentration results that show that the first minimum of a random lattice becomes heavily concentrated around $\text{gh}(\mathcal{L})$ for growing n (see [2] for a survey). In cryptanalysis, this is often also heuristically assumed to be the case for ‘random’ lattices in a more broader sense, hence the name.

Beyond the existence of a good packing one can also show the existence of a lattice with a good (small) smoothing parameter for any $\varepsilon > 0$.

Corollary 2 (Random smoothing). *For any dimension n and volume D and any $\varepsilon > 0$ there exists a lattice $\mathcal{L} \in \mathcal{L}_{[n,D]}$ with*

$$\eta_\varepsilon(\mathcal{L}) \leq \left(\frac{\text{vol}(\mathcal{L})}{\varepsilon} \right)^{\frac{1}{n}}.$$

Proof. Without loss of generality we normalize to have volume $D = 1$, and consider the function $f(x) = e^{-\pi s^2 \|x\|^2}$ for $s > 0$ and for which $\|x\|^{n+1} f(x)$ is clearly bounded on \mathbb{R}^n . Applying Theorem 4 we obtain that

$$\int_{\mathcal{L}^* \in \mathcal{L}_{[n]}} \sum_{x \in \mathcal{L}^* \setminus \{0\}} e^{-\pi s^2 \|x\|^2} d\mu_n = \int_{\mathbb{R}^n} e^{-\pi s^2 \|x\|^2} dx = s^{-n}.$$

Let $s := \varepsilon^{-\frac{1}{n}}$, by the above there exists a lattice $\mathcal{L}^* \in \mathcal{L}_{[n]}$ such that

$$\theta_{\mathcal{L}^*}(\exp(-\pi s^2)) = 1 + \sum_{x \in \mathcal{L}^* \setminus \{0\}} e^{-\pi s^2 \|x\|^2} \leq 1 + s^{-n} = 1 + \varepsilon.$$

Then by definition for the dual $\mathcal{L} \in \mathcal{L}_{[n]}$ of \mathcal{L}^* we have $\eta_\varepsilon(\mathcal{L}) \leq s$. □

Note that Corollary 2 goes beyond just combining Corollary 1 with bounds based in the (dual) minimal distance like $\eta_\varepsilon(\mathcal{L}) \leq \sqrt{\ln(1/\varepsilon)}/\lambda_1(\mathcal{L}^*)$ for $\varepsilon \in (0, e^{-n}]$. In particular, it gives a better and tighter bound for large $\varepsilon > e^{-n}$, which is precisely the regime interesting for cryptography. This bound is represented in a different setting in [8, Proposition 4.].

1.3 Lattice Isomorphism Problem

The lattice isomorphism problem asks if two lattices \mathcal{L}_1 and \mathcal{L}_2 are related to each-other by an orthonormal transformation. In terms of bases this means there exists both an orthonormal transformation on the left and a unimodular (basis) transformation on the right that transforms one basis into the other. In the setting of gram matrices or quadratic forms the orthonormal transformation is irrelevant and only the unimodular transformation remains but is applied (transposed) to both sides.

⁴ One could argue that for the actual Gaussian Heuristic this factor should not be neglected, but it is often ignored as it quickly converges to 1 as n grows.

Definition 3 (Lattice Isomorphism). We call two full-rank lattices $\mathcal{L}_1, \mathcal{L}_2 \subset \mathbb{R}^n$ isomorphic and write $\mathcal{L}_1 \cong \mathcal{L}_2$ if there exists an orthonormal transformation $O \in \mathcal{O}_n(\mathbb{R})$ such that $O \cdot \mathcal{L}_1 = \mathcal{L}_2$. If $\mathcal{L}_i = \mathcal{L}(B_i)$ for bases $B_1, B_2 \in \mathcal{GL}_n(\mathbb{R})$ then $\mathcal{L}_1 \cong \mathcal{L}_2$ if and only if:

1. there exist $O \in \mathcal{O}_n(\mathbb{R}), U \in \mathcal{GL}_n(\mathbb{Z})$ such that $O \cdot B_1 \cdot U = B_2$, or equivalently,
2. there exist $U \in \mathcal{GL}_n(\mathbb{Z})$ such that $U^\top G_1 U = G_2$ where $G_i = B_i^\top B_i$.

In the quadratic form setting the gram matrices G_1, G_2 are called \mathbb{Z} -equivalent or simply *equivalent* if they represent isomorphic lattices. Given two isomorphic lattices it is computationally a hard problem to find the isomorphism between them.

Definition 4 (Search LIP). Given a pair of isomorphic lattices $\mathcal{L}_1, \mathcal{L}_2 \subset \mathbb{R}^n$, compute an orthonormal transformation $O \in \mathcal{O}_n(\mathbb{R})$ such that $O \cdot \mathcal{L}_1 = \mathcal{L}_2$.

We allow for the lattice to be either represented by a basis or by a gram matrix. In the case of a gram matrices G_1, G_2 we rephrase search LIP as finding an unimodular transformation $U \in \mathcal{GL}_n(\mathbb{Z})$ such that $U^\top G_1 U = G_2$. This makes the orthonormal transformation irrelevant. Furthermore note that if we restrict to integral lattices, then this formulation of LIP only involves integer arithmetic.

The best provable algorithm to solve search LIP runs in time $n^{O(n)}$ [15]. Furthermore, the general algorithms for solving LIP [12, 22, 23] require as a first step the computation of short lattice vectors, which takes time $2^{O(n)}$. The high complexity of these algorithm is what makes LIP interesting as a hardness assumption.

Often however, the security proof of LIP based cryptographic schemes is not based on the search variant, but on a distinguishing variant. For any lattice \mathcal{L} we will denote its isomorphism class by $[\mathcal{L}] = \{O \cdot \mathcal{L} : O \in \mathcal{O}_n(\mathbb{R})\}$. We can then ask to distinguish between different isomorphism classes.

Definition 5 (Distinguish LIP). Let $\mathcal{L}_1, \mathcal{L}_2$ be non-isomorphic lattices. Given any lattice $\mathcal{L} \in [\mathcal{L}_b]$ for a uniformly random $b \leftarrow \mathcal{U}(\{1, 2\})$. Recover b .

2 The Genus of a Lattice

For any two isomorphic lattices $\mathcal{L}_1 \cong \mathcal{L}_2$ we have that $\text{vol}(\mathcal{L}_1) = \text{vol}(\mathcal{L}_2)$. The (co)volume of the lattice is thus an efficiently computable *invariant* for lattice isomorphisms. Other examples of this are the $\text{scale}(\mathcal{L})$ or parity $\text{par}(\mathcal{L})$ of an integral lattice. If two lattices have distinct invariants we can use that to solve distinguish LIP, simply by computing the same invariant for $\mathcal{L} \in [\mathcal{L}_b]$ and see if it matches that of \mathcal{L}_1 or \mathcal{L}_2 . When instantiating a cryptographic scheme based on distinguishing LIP [1, 3, 4, 11] we thus should make sure that the non-isomorphic lattices $\mathcal{L}_1, \mathcal{L}_2$ match on all efficiently computable invariants.

In terms of quadratic forms the lattice isomorphism problem boils down to \mathbb{Z} -equivalence, which is seemingly hard. It is natural however to look at weaker forms of equivalence over larger rings $R \supset \mathbb{Z}$ which might be more efficient to compute.

Definition 6 (*R*-equivalence). Let $R \supset \mathbb{Z}$ be any ring containing \mathbb{Z} . We say that two integral lattices $\mathcal{L}_1, \mathcal{L}_2 \subset \mathbb{R}^n$ are *R*-equivalent if $\mathcal{L}_1 \otimes_{\mathbb{Z}} R \cong \mathcal{L}_2 \otimes_{\mathbb{Z}} R$. Alternatively, two integral quadratic forms $G_1, G_2 \in \mathcal{S}_n^{>0}(\mathbb{Z})$ are *R*-equivalent if there exists a $U \in \mathcal{GL}_n(R)$ such that $U^\top G_1 U = G_2$ over R .

One such weaker form of equivalence is that over the p -adic integers \mathbb{Z}_p . In contrast to \mathbb{Z} -equivalence it is efficient to compute if two integral lattices are \mathbb{Z}_p -equivalent for any prime p . In short, it follows from the fact that forms are (block-)diagonalizable over \mathbb{Z}_p , after which the equivalence is relatively easy to determine. See [6, Chapter 15.7] for more information on this computation and how to determine a complete set of invariants for \mathbb{Z}_p -equivalence. Furthermore, assuming that $\text{vol}(\mathcal{L}_1) = \text{vol}(\mathcal{L}_2)$, we only have to focus on those primes p that divide $2 \text{vol}(\mathcal{L}_i)^2$, as for all other primes the \mathbb{Z}_p -equivalence follows directly. Assuming that we know the factorization of $\text{vol}(\mathcal{L}_i)^2$ we can thus determine the \mathbb{Z}_p -equivalence of \mathcal{L}_1 and \mathcal{L}_2 for all primes p .

Definition 7 (Genus [6, Chapter 15]). The genus $\text{gen}(\mathcal{L})$ of an integral lattice $\mathcal{L} \subset \mathbb{R}^n$ consists of all (integral) lattices of dimension n that are \mathbb{Z}_p -equivalent to \mathcal{L} for all primes p . Given an integral lattice \mathcal{L} , and the prime factorization of $\text{vol}(\mathcal{L})^2$, we can efficiently compute a canonical label of the genus it corresponds to.

In case we are not only considering full-rank lattices there is an extra condition that the lattices must be equivalent over the reals. However, two full-rank lattices of the same dimension are always equivalent over \mathbb{R} so we can safely ignore this condition.

Two lattices $\mathcal{L}_1 \cong \mathcal{L}_2$ that are \mathbb{Z} -equivalent are also \mathbb{Z}_p equivalent for any prime p given that $\mathbb{Z} \subset \mathbb{Z}_p$, and thus $\text{gen}(\mathcal{L}_1) = \text{gen}(\mathcal{L}_2)$. In particular, if we have lattices $\mathcal{L}_1, \mathcal{L}_2$ such that $\text{gen}(\mathcal{L}_1) \neq \text{gen}(\mathcal{L}_2)$ it follows directly that they cannot be equivalent, and this can be efficiently computed. In this way, the genus of a lattice gives us a strong invariant for lattice isomorphisms. Note that as a result it is also well defined to speak about the genus of a \mathbb{Z} -equivalence class $[\mathcal{L}]$, and denote $\text{gen}([\mathcal{L}]) := \text{gen}(\mathcal{L})$. As far as we know the genus covers all the known efficiently computable invariants, which makes it interesting for us to study. As a result we can also simply define $\text{vol}(\mathcal{G}) := \text{vol}(\mathcal{L}), \text{scale}(\mathcal{G}) := \text{scale}(\mathcal{L})$ and $\text{rk}_p(\mathcal{G}) := \text{rk}_p(\mathcal{L})$ for $\mathcal{L} \in \mathcal{G}$ which is independent of the chosen representative.

Remark 1. For simplicity we only consider here the genus of *integral* lattices. Because the structure of the genus is invariant under integer scaling one could easily extend these notions to rational lattices. In particular, $\text{scale}(\mathcal{L})$ is an invariant of the (rational) genus. More generally, similar notions exist for quadratic forms over the ring of integers of number fields [18, 30].

2.1 Randomness over the Genus

Every genus consists of a finite number of \mathbb{Z} -equivalence classes [20] and thus one could consider a uniform distribution $\mathcal{U}(\mathcal{G})$ over it. However, mathematically,

this is not the most natural distribution and we have to give each equivalence class a slightly different weight depending on the size of their automorphism group.

Definition 8 (Randomness over a genus). *For a genus \mathcal{G} we define the probability distribution $\mathcal{D}(\mathcal{G})$ which samples $[\mathcal{L}] \in \mathcal{G}$ with relative mass $m(\mathcal{L}) := 1/|\text{Aut}(\mathcal{L})|$. In particular, for any $[\mathcal{L}] \in \mathcal{G}$ we have*

$$\Pr_{[\mathcal{L}'] \leftarrow \mathcal{D}(\mathcal{G})} [\mathcal{L}' \cong \mathcal{L}] = \frac{m(\mathcal{L})}{\sum_{[\mathcal{L}'] \in \mathcal{G}} m(\mathcal{L}')}.$$

Just as in the case of fully random lattice the measure $m(\mathcal{L}) = 1/\text{Aut}(\mathcal{L})$ again follows naturally, this time from the Haar measure on $\mathcal{O}_n(\mathbb{R})$. More precisely, equip $\mathcal{O}_n(\mathbb{R})$ with its volume 1 Haar measure. The isometry class $[\mathcal{L}]$ is then endowed with an $\mathcal{O}_n(\mathbb{R})$ -invariant measure m with total measure $m([\mathcal{L}]) = m(\mathcal{O}_n(\mathbb{R}) \cdot \mathcal{L}) = \frac{1}{|\text{Aut}(\mathcal{L})|}$, because \mathcal{L} is left invariant by precisely $|\text{Aut}(\mathcal{L})|$ orthonormal transformations. Furthermore, this is precisely the distribution one gets when restricting the general probability distribution $\mu_{n,D}$ to the genus \mathcal{G} . In particular, if we sample a random $\mathcal{L}' \in \mu_{n,D}$ under the restriction that $\mathcal{L}' \in \mathcal{G}$, then the probability that $\mathcal{L}' \cong \mathcal{L}$ for some $\mathcal{L} \in \mathcal{G}$ is precisely $m(\mathcal{L})/\sum_{[\mathcal{L}']} m(\mathcal{L}')$.

Given one representative $\mathcal{L} \in \mathcal{G}$ in a genus there is a natural notion of p -neighbours within the same genus for any prime $p \nmid 2 \text{vol}(\mathcal{G})^2$, namely all those lattices \mathcal{L}' in the same genus for which $\mathcal{L} \cap \mathcal{L}'$ has index p in both \mathcal{L} and \mathcal{L}' . These connections turn the genus into a graph where the nodes are isomorphism classes $[\mathcal{L}]$ and the edges are p -neighbours. For large enough primes p this graph is furthermore connected⁵. By picking any of those (finite) p -neighbours uniformly at random one obtains a random walk over this graph. Another reason for the distribution $\mathcal{D}(\mathcal{G})$ to be natural is that for large enough p it is the natural limit or stationary distribution for this random walk, and this precisely allows us to sample efficiently from $\mathcal{D}(\mathcal{G})$ as stepping through this graph is efficient [16]. In fact for large enough p , a single step is enough to be negligibly close to the distribution of $\mathcal{D}(\mathcal{G})$ [5], i.e., any isomorphism class is reached with relative weight $w(\mathcal{L})$.

Theorem 5 ([5, 16]). *There exists an efficient algorithm to sample from $\mathcal{D}(\mathcal{G})$.*

2.2 Siegel’s Mass Formulas

Given that the genus is an invariant under \mathbb{Z} -equivalence we can view any particular genus as a set of \mathbb{Z} -equivalence classes. This set is always finite, but more surprisingly we can even compute a notion of its size, i.e., its *mass*.

⁵ We ignore here the rare case that the genus splits into multiple spinor-genera.

Theorem 6 (Smith-Siegel-Minkowski Mass formula [26]). *Let*

$$M(\mathcal{G}) := \sum_{[\mathcal{L}] \in \text{gen}(\mathcal{G})} \frac{1}{|\text{Aut}(\mathcal{L})|}$$

be the mass of \mathcal{G} , where the sum is over all equivalence classes in the genus. Given as input a gram matrix G of any lattice $\mathcal{L} \in \mathcal{G}$, and the prime factorization of $\det(\mathcal{G})^2$, the mass $M(\mathcal{G})$ can be computed in polynomial time in the input, n and $\log(\text{vol}(\mathcal{G}))$.

This directly gives an estimate for the number of equivalence classes.

Corollary 3. *Let \mathcal{G} be a non-empty genus of dimension n and let $M(\mathcal{G})$ be its mass. Then the genus contains at least $2M(\mathcal{G})$ and for $n > 10$ at most $2^n \cdot n! \cdot M(\mathcal{G})$ distinct equivalence classes.*

Proof. Clearly $\{\pm I_n\} \subset \text{Aut}(\mathcal{L})$ and thus $|\text{Aut}(\mathcal{L})| \geq 2$ for any lattice. Furthermore, Feit [13] showed in 1996 that with some exception for $n \leq 10$, \mathbb{Z}^n has the largest automorphism group for any n -dimensional lattice \mathcal{L} and thus $|\text{Aut}(\mathcal{L})| \leq |\text{Aut}(\mathbb{Z}^n)| = 2^n \cdot n!$ for any $n > 10$. The latter is one of the first significant results based on the classification of finite simple groups in an unpublished manuscript from 1984 of Weisfeiler [31]. The result follows immediately from these bounds and the definition of the mass formula. □

We remark that asymptotically most lattices have a trivial automorphism group and thus we expect the number of equivalence classes to be quite close to $2M(\mathcal{G})$. While the existence of such a mass formula might already be surprising, one can go even further than this.

Note that computing the first minimum $\lambda_1(\mathcal{L})^2 = \arg \min_{k \geq 1} \{N_{\mathcal{L}}(k) > 0\}$ is already a hard problem. So to say anything about the theta series of a given lattice is very hard. However, once we start to look at the expected theta series over a genus, we suddenly are able to compute it efficiently.

Theorem 7 (Siegel’s mass formula [26]). *For a non-empty genus \mathcal{G} we define the expected theta series as*

$$\Theta_{\mathcal{G}}(q) = \mathbb{E}_{[\mathcal{L}] \leftarrow \mathcal{D}(\mathcal{G})} [\theta_{\mathcal{L}}(q)] = \frac{\sum_{[\mathcal{L}] \in \mathcal{G}} \frac{1}{|\text{Aut}(\mathcal{L})|} \cdot \theta_{\mathcal{L}}(q)}{\sum_{[\mathcal{L}] \in \mathcal{G}} \frac{1}{|\text{Aut}(\mathcal{L})|}} =: 1 + \sum_{k=1}^{\infty} N_{\mathcal{G}}(k) \cdot q^k.$$

Given the prime factorizations of $\text{vol}(\mathcal{G})^2$ and $k > 0$ the coefficient $N_{\mathcal{G}}(k)$ of $\Theta_{\mathcal{G}}(q)$ can be efficiently computed.

Remark 2. While these mass formulas are indeed computable in polynomial time, it is not necessarily an easy task to do it. In particular, the computations are very prone to errors. For an extensive explanation on how to compute the Smith-Siegel-Minkowski mass formula see [7]. In Sect. 4.1 we explain (partly) how to compute Siegel’s mass formula. Both mass formulas have been implemented in Sagemath [28]. For example, the total mass of a genus can be computed by calling `Q.conway_mass()` on a `QuadraticForm Q`.

3 On the Existence of Lattices with Good Properties

3.1 Lattices with Good Properties

The theta series give a lot of information about the geometric properties of a lattice. Due to this connection we can also hope that from the expected theta series over a genus, we can derive the existence of a lattice with good geometric properties within this genus.

One such example is to derive a good lattice packing. Given that the genus already fixed the determinant this means we want to find a lattice with a large first minimum $\lambda_1(\mathcal{L})$. Note that for such a lattice the theta series coefficients $N_1(\mathcal{L}), \dots, N_{\lambda_1(\mathcal{L})^2-1}(\mathcal{L})$ are zero. In case the first few coefficients of the expected theta series are small we can conclude by a counting argument that at least one of the lattices must have only zeros there. Beyond existence, recall that for a non-negative random variable X and $a > 0$, Markov's inequality states that $\Pr[X < a] \geq 1 - \frac{\mathbb{E}[X]}{a}$, which can directly gives us a lower bound on the probability density of such lattices. This leads to a density analogue of the Minkowski-Hlawka Theorem restricted to a fixed genus.

Lemma 2 (Good packing density). *Let \mathcal{G} be a genus with expected theta series $\Theta_{\mathcal{G}}(q) = 1 + \sum_{k=1}^{\infty} N_{\mathcal{G}}(k)q^k$. If $\sum_{k=1}^{\lambda-1} N_{\mathcal{G}}(k) < 2r$ for some $0 < r \leq 1$ and some integer $\lambda \geq 1$, then $\Pr_{[\mathcal{L}] \leftarrow \mathcal{D}(\mathcal{G})}[\lambda_1(\mathcal{L})^2 \geq \lambda] > 1 - r$. In particular, then there exists a lattice $\mathcal{L} \in \mathcal{G}$ such that $\lambda_1(\mathcal{L})^2 \geq \lambda$.*

Proof. We consider the non-negative random variable $\sum_{k=1}^{\lambda-1} N_{\mathcal{L}}(k)$ where $[\mathcal{L}] \leftarrow \mathcal{D}(\mathcal{G})$. By definition its expectation is given by $\sum_{k=1}^{\lambda-1} N_{\mathcal{G}}(k)$. By Markov's inequality we then obtain

$$\Pr_{[\mathcal{L}] \leftarrow \mathcal{D}(\mathcal{G})} \left[\sum_{k=1}^{\lambda-1} N_{\mathcal{L}}(k) < 2 \right] \geq 1 - \frac{\sum_{k=1}^{\lambda-1} N_{\mathcal{G}}(k)}{2} > 1 - \frac{2r}{2} = 1 - r,$$

from which the result follows as $\sum_{k=1}^{\lambda-1} N_{\mathcal{L}}(k) < 2$ if and only if $\lambda_1(\mathcal{L})^2 \geq \lambda$. For the existence result note that the probability is strictly positive for $r \leq 1$. \square

Remark 3. One can observe that Lemma 2 proves a stronger statement than merely the existence of a good packing. It also gives a lower bound on the probability that any lattice sampled from $\mathcal{D}(\mathcal{G})$ achieves a certain minimum distance. Moreover, this can be turned into a quantitative statement on the number of such lattices by considering the mass $M(\mathcal{G})$ of the genus. In particular, if $\sum_{k=1}^{\lambda-1} N_{\mathcal{G}}(k) < 2r$ for $0 < r \leq 1$ then there exist at least $2(1 - r)M(\mathcal{G})$ non-isomorphic lattices $\mathcal{L} \in \mathcal{G}$ such that $\lambda_1(\mathcal{L})^2 \geq \lambda$.

Remark 4. Just as for general random lattices there exists a variant of Siegel's mass formula that computes the number of *primitive* vectors of squared norm k (see e.g. [14]). Clearly, Lemma 2 works just as well with these quantities. For large n however, it does not seem to make a large difference (just as for the Hlawka-Minkowski Theorem), so we do not consider this small improvement.

To obtain a good dual lattice packing one can apply the same Lemma to the (scaled) dual theta series. Note that one could even apply the same proof to the primal and dual theta series simultaneously to obtain a single lattice with both a good primal and dual packing.

Lemma 3 (Good primal and dual packing). *Let \mathcal{G} be a genus with expected theta series $\Theta_{\mathcal{G}}(q) = 1 + \sum_{k=1}^{\infty} N_{\mathcal{G}}(k)q^k$, and let $c\mathcal{G}^{-1}$ for $c = \text{scale}(\mathcal{G}^{-1})^{-1} \in \mathbb{Q}$ be the integral scaled dual genus with expected theta series $\Theta_{c\mathcal{G}^{-1}}(q) = 1 + \sum_{k=1}^{\infty} N_{c\mathcal{G}^{-1}}(k)q^k$. Let $0 < r \leq 1$ and let $\lambda, \lambda' \geq 1$ be integers. If $\sum_{k=1}^{\lambda-1} N_{\mathcal{G}}(k) + \sum_{k=1}^{\lambda'-1} N_{c\mathcal{G}^{-1}}(k) < 2r$, then $\Pr_{[\mathcal{L}] \leftarrow \mathcal{D}(\mathcal{G})} \left[\lambda_1(\mathcal{L})^2 \geq \lambda \text{ and } \lambda_1(\mathcal{L}^*)^2 \geq \frac{\lambda'}{c} \right] > 1 - r$. In particular, then there exists a lattice $\mathcal{L} \in \mathcal{G}$ with $\lambda_1(\mathcal{L})^2 \geq \lambda$ and $\lambda_1(\mathcal{L}^*)^2 \geq \frac{\lambda'}{c}$.*

The existence of a good dual packing immediately also implies the existence of a lattice with good smoothing as for $\varepsilon \in (0, e^{-n}]$ we have $\eta_{\varepsilon}(\mathcal{L}) \leq \sqrt{\ln(1/\varepsilon)}/\lambda_1(\mathcal{L}^*)$. Usually however, we are interested in the smoothing for larger values of ε . In that case we can consider the following result.

Lemma 4 (Good smoothing parameter). *Let $\varepsilon > 0$, $0 < r \leq 1$ and let $s > 0$ be such that $\Theta_{\mathcal{G}}(\exp(-\pi s^2)) < 1 + r\varepsilon$, then $\Pr_{[\mathcal{L}] \leftarrow \mathcal{D}(\mathcal{G})} [\eta_{\varepsilon}(\mathcal{L}^*) < s] > 1 - r$. In particular, then there exists a lattice $\mathcal{L} \in \mathcal{G}$ such that $\eta_{\varepsilon}(\mathcal{L}^*) < s$.*

Proof. Note that $\Theta_{\mathcal{G}}(\exp(-\pi s^2)) - 1$ is the expectation of the non-negative random variable $\Theta_{\mathcal{L}}(\exp(-\pi s^2)) - 1$ where $[\mathcal{L}] \leftarrow \mathcal{D}(\mathcal{G})$. By Markov's inequality we then obtain

$$\Pr_{[\mathcal{L}] \leftarrow \mathcal{D}(\mathcal{G})} [\Theta_{\mathcal{L}}(\exp(-\pi s^2)) - 1 < \varepsilon] \geq 1 - \frac{\Theta_{\mathcal{G}}(\exp(-\pi s^2)) - 1}{\varepsilon} > 1 - \frac{r\varepsilon}{\varepsilon} = 1 - r,$$

from which the result follows as by definition $\Theta_{\mathcal{L}}(\exp(-\pi s^2)) < 1 + \varepsilon$ if and only if $\eta_{\varepsilon}((\mathcal{L}')^*) < s$. For the existence result note that the probability is strictly positive for $r \leq 1$. □

3.2 Example: Unimodular Lattices

Let us consider the easiest, but in some sense also most interesting genera, those of unimodular lattices. These lattices are self-dual, which protects them from Hull attacks like [9]. In addition, due to their small determinant one could in principle describe them using small matrix entries, potentially leading to smaller keys. Because unimodular lattices have determinant 1 we only have to focus on the p -adic equivalence for $p = 2$. Furthermore, because 2 does not divide the determinant the full 2-adic equivalence is determined by the parity $\text{par}(\mathcal{L}) \in \{1, 2\}$ of these unimodular lattices. We thus obtain two genera, the *odd* and *even* one. Here we consider as an example the even case for which all the vectors $v \in \mathcal{L}$ have an even squared norm $\|v\|^2$.

The even case includes the famous root lattice E_8 and the Leech lattice A_{24} . Even unimodular lattices only exist in dimensions that are a multiple of 8 (see e.g. [25, p. 53]). For even unimodular lattices the expected theta series is given

by a rational scaling of the q -expansion of the Eisenstein series [19]. To be more precise, let $n \geq 8$ with $8|n$, and let $\mathcal{G}_{n,e}$ be the genus of even unimodular lattices of dimension $n = 8m$, then we have

$$\Theta_{\mathcal{G}_{8m,e}}(q) = E_{4m}(q^2) = 1 + \frac{-8m}{B_{4m}} \sum_{k=1}^{\infty} \sigma_{4m-1}(k)q^{2k},$$

where B_i is the i -th Bernoulli number, and $\sigma_z(m) = \sum_{d|m} d^z$ is the sum of positive divisors function.

Using the above expected theta series and Lemma 2 we can prove the existence of an even unimodular lattice with first minimum essentially as indicated by the Gaussian Heuristic.

Lemma 5 (Even packing). *Let $n = 8m \geq 8$ with $m \in \mathbb{N}$, then there exists an n -dimensional even unimodular lattice \mathcal{L} with $\lambda_1(\mathcal{L})^2 \geq 2 \left[\frac{1}{2} \cdot \left(\frac{3\zeta(n/2)}{2\omega_n} \right)^{2/n} \right] \approx n/2\pi e$.*

Proof. Let $k' = \left\lceil \frac{1}{2} \cdot \left(\frac{3\zeta(4m)}{2\omega_{8m}} \right)^{1/4m} \right\rceil$. To apply Lemma 2 we need to show that the sum of the first $k' - 1$ non-trivial coefficients of $\Theta_{\mathcal{G}_{8m,e}}(q)$ is bounded by 2. Recall that these have values $\frac{-8m}{B_{4m}}\sigma_{4m-1}(k)$ for $k = 1, \dots, k' - 1$. First, note that

$$\begin{aligned} \sum_{k=1}^{k'-1} \sigma_{4m-1}(k) &= \sum_{k=1}^{k'-1} \sum_{d|k} d^{4m-1} = \sum_{d=1}^{k'-1} \left\lfloor \frac{k'-1}{d} \right\rfloor d^{4m-1} \leq \sum_{d=1}^{k'-1} (k'-1) \cdot d^{4m-2} \\ &\leq (k'-1) \cdot \frac{1}{4m-1} \left(k' - \frac{1}{2} \right)^{4m-1} < \frac{(k' - \frac{1}{2})^{4m}}{4m-1} \\ &\leq \frac{3\zeta(4m) \cdot \omega_{8m}^{-1}}{2^{4m+1} \cdot (4m-1)}, \end{aligned}$$

where we use that $d^{4m-2} \leq \int_{d-\frac{1}{2}}^{d+\frac{1}{2}} x^{4m-2} dx$. Secondly, by using the common identity for even Bernoulli numbers and the volume ω_{8m} of an $8m$ -dimensional unit ball, we get

$$\frac{-8m}{B_{4m}} = 8m\omega_{8m} \cdot \frac{2^{4m-1}}{\zeta(4m)}.$$

Combining the two we get that

$$\frac{-8m}{B_{4m}} \cdot \sum_{k=1}^{k'-1} \sigma_{4m-1}(k) < 8m \cdot \omega_{8m} \cdot \frac{2^{4m-1}}{\zeta(4m)} \cdot \frac{3\zeta(4m) \cdot \omega_{8m}^{-1}}{2^{4m+1} \cdot (4m-1)} = \frac{3}{2} \cdot \frac{4m}{4m-1} \leq 2.$$

We can conclude by Lemma 2 and the even parity that there exists a lattice $\mathcal{L} \in \mathcal{G}_{n,e}$ with minimum $\lambda_1(\mathcal{L})^2 \geq 2k'$. □

Remark 5. We want to emphasize that this result is not novel. The bound in Lemma 5 is essentially the same as claimed by Milnor [19, p. 47], where a lower bound of $2 \cdot \lceil \frac{1}{2}(\frac{3}{5}\omega_n)^{-2/n} \rceil$ is given based on computations in [25]. The proof here uses a different representation of the Eisenstein series and is more elementary. For a concrete comparison see Fig. 1. For odd unimodular lattices Milnor [19, p. 46] gives a full proof for a lower bound of $\lambda_1(\mathcal{L})^2 \geq \lceil (\frac{3}{5}\omega_n)^{-2/n} \rceil$.

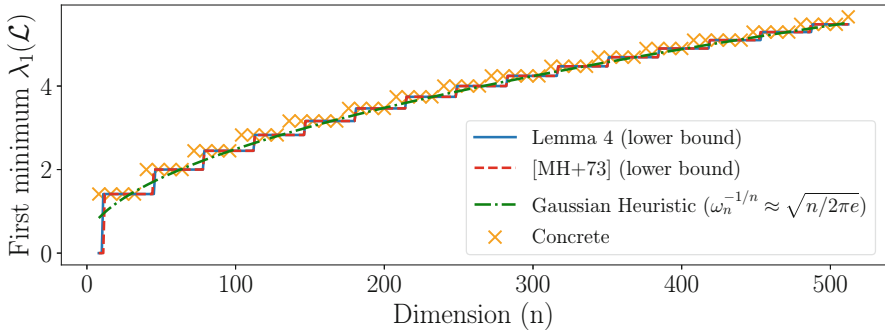


Fig. 1. First minimum guarantee for even unimodular lattices as given by Lemma 5 and [19], compared to concrete values directly computed using $\Theta_{\mathcal{G}_{n,e}}$ and Lemma 2.

For the smoothing parameter we can also prove a result that is essentially tight, and similar to Corollary 2 for random lattices. We restrict ourselves to $\varepsilon \geq \Omega(e^{-n})$ as the general bounds based on the dual minimum distance often fail to give tight results in this important regime for cryptography.

First we require two small technical lemmas.

Lemma 6 (Technical lemma I). *For $x \geq 2$ and any integer $y \geq 1$ we have $\sigma_x(y) \leq \zeta(x) \cdot y^x$, where $\sigma_x(y) = \sum_{d|y} d^x$ is the sum of positive divisors function.*

Proof. Let $y = p_1^{a_1} \cdots p_n^{a_n}$, be the prime factorization of y with p_1, \dots, p_n distinct primes and $n \geq 0, a_i > 0$. Now for any prime power p^a we have $\sigma_x(p^a) = \sum_{d|p^a} d^x = 1 + p^x + \dots + p^{ax} = p^{ax} \cdot \frac{1 - p^{-(a+1)x}}{1 - p^{-x}} \leq p^{ax} \cdot \frac{1}{1 - p^{-x}}$. The sum of divisors function σ_x is multiplicative for coprime inputs and thus we get

$$\sigma_x(y) = \prod_{i=1}^n \sigma_x(p_i^{a_i}) \leq \prod_{i=1}^n p_i^{a_i x} \cdot \frac{1}{1 - p_i^{-x}} \leq y^x \cdot \prod_{p \text{ prime}} \frac{1}{1 - p^{-x}} = \zeta(x) \cdot y^x.$$

□

Lemma 7 (Technical lemma II). *Let $0 < c \leq C$ for some constant $C > 0$, then we have for $x \geq 2$ that*

$$\text{Li}_{-x}(\exp(-c)) \leq \left(1 + 2 \cdot \sum_{k=1}^{\infty} (1 + 4k^2\pi^2/C^2)^{-3/2} \right) \cdot \Gamma(1 + x) \cdot c^{-x-1},$$

where $\text{Li}_y(z) = \sum_{k=1}^{\infty} \frac{z^k}{k^y}$ is the polylogarithm function.

Proof. For negative $y < 0$ we have the following identity by Wood [32, (13.1)]:

$$\begin{aligned} \text{Li}_{-x}(\exp(-c)) &= \Gamma(x + 1) \cdot \sum_{k=-\infty}^{\infty} (2k\pi i + c)^{-x-1} \\ &= \Gamma(x + 1) \cdot c^{-x-1} \cdot \sum_{k=-\infty}^{\infty} (2k\pi i/c + 1)^{-x-1}. \end{aligned}$$

The summation is real-valued as the terms $\pm k$ are conjugates, and we have

$$\begin{aligned} \sum_{k=-\infty}^{\infty} (2k\pi i/c + 1)^{-x-1} &\leq 1 + 2 \cdot \sum_{k=1}^{\infty} |2k\pi i/c + 1|^{-x-1} \\ &= 1 + 2 \cdot \sum_{k=1}^{\infty} (1 + 4k^2\pi^2/c^2)^{-(x+1)/2} \\ &\leq 1 + 2 \cdot \sum_{k=1}^{\infty} (1 + 4k^2\pi^2/C^2)^{-1.5}. \end{aligned}$$

□

Lemma 8 (Even smoothing). *Let $n = 8m \geq 8$ with $m \in \mathbb{N}$, $C = 17.8$, and let $\varepsilon > C \cdot e^{-n}$, then there exists an n -dimensional even unimodular lattice $\mathcal{L} \in \mathcal{G}_{n,e}$ such that $\eta_\varepsilon(\mathcal{L}) \leq (C/\varepsilon)^{1/n}$.*

Proof. Let $s = (C/\varepsilon)^{1/8m} \leq e$. To apply Lemma 4 we have to show that the following sum is bounded by ε :

$$\mu = \frac{-8m}{B_{4m}} \cdot \sum_{k=1}^{\infty} \sigma_{4m-1}(k) \cdot \exp(-2\pi s^2 k).$$

First, using Lemma 6 we have the bound $\sigma_{4m-1}(k) \leq \zeta(4m - 1) \cdot k^{4m-1} \leq \zeta(3)k^{4m-1}$. This gives us that

$$\mu \leq \frac{-8m\zeta(3)}{B_{4m}} \cdot \sum_{k=1}^{\infty} k^{4m-1} \cdot \exp(-2\pi s^2 k) = \frac{-8m\zeta(3)}{B_{4m}} \text{Li}_{1-4m}(e^{-2\pi s^2}),$$

where $\text{Li}_p(z) = \sum_{k=1}^{\infty} \frac{z^k}{k^p}$ is the polylogarithm function. From Lemma 7 we get that $\text{Li}_{1-4m}(e^{-2\pi s^2}) \leq 14.78 \cdot \Gamma(4m) \cdot (2\pi s^2)^{-4m}$, which combined with the identity for the even Bernoulli numbers gives us

$$\begin{aligned} \mu &\leq \frac{-8m\zeta(3)}{B_{4m}} \text{Li}_{1-4m}(e^{-2\pi s^2}) \leq \zeta(3) \cdot 8m \cdot \frac{(2\pi)^{4m}}{2 \cdot (4m)!} \cdot 14.78 \cdot \Gamma(4m) \cdot (2\pi s^2)^{-4m} \\ &= \zeta(3) \cdot 14.78 \cdot s^{-8m} \leq C \cdot \frac{\varepsilon}{C} = \varepsilon. \end{aligned}$$

We conclude by Lemma 4 and the fact that unimodular lattices are self-dual. □

We note that in principle the above Lemma is not restrained to $\varepsilon \geq Ce^{-n}$ and could easily be adapted to handle $\varepsilon < C \cdot e^{-n}$ at the cost of a larger constant C . In particular, by slightly adapting Lemma 7 we can obtain a bound of $(3\zeta(3)/\varepsilon)^{1/(n-2)}$ that is valid for any $3\zeta(3) \geq \varepsilon > 0$. Numerical evidence indicates that the bound from Lemma 7 could be improved further leading to a lower constant C both here and for Theorem 2. However in this regime the bound obtained from a good dual packing is better than the one given here. In Fig. 2 we can see that the bound in Lemma 8 is rather tight compared to a direct application of Lemma 4.

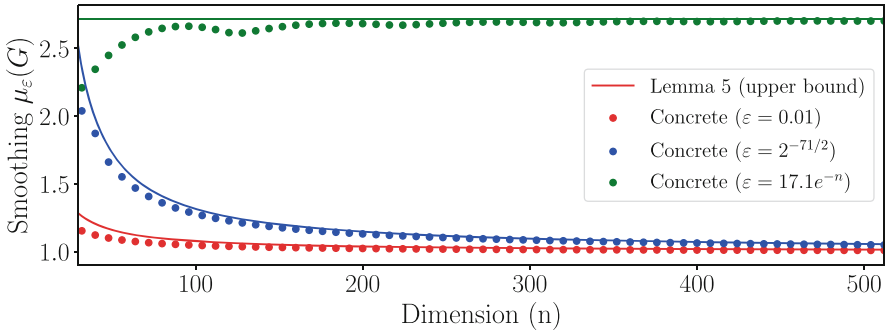


Fig. 2. Smoothing bound for even unimodular lattices as given by Lemma 8, compared to concrete values directly computed using $\Theta_{\mathcal{G}_n, \varepsilon}$ and Lemma 4. The value $\varepsilon = 2^{-71/2} = 1/\sqrt{q_s \cdot \lambda}$ is common in hash-and-sign schemes with $\lambda = 128$ bits of security that can sign 2^{64} signatures.

4 A General Result

We now consider the general case for almost all genera. By our knowledge existing literature only show packing results for the (simpler) unimodular case, but the results do generalize.

Theorem 1 (General packing). *For any integral genus \mathcal{G} in dimension $n \geq 6$ such that $\text{rk}_p(\mathcal{G}) \geq 6$ for all primes p , and any constant $0 < c \leq 1$, we have*

$$\Pr_{[\mathcal{L}] \leftarrow \mathcal{D}(\mathcal{G})} \left[\lambda_1(\mathcal{L})^2 \geq \left[c^2 \cdot \left(\frac{7\zeta(3)}{9\zeta(2)} \cdot \frac{\text{vol}(\mathcal{L})}{\omega_n} \right)^{2/n} \right] \right] > 1 - c^n.$$

In particular, there exists a lattice $\mathcal{L} \in \mathcal{G}$ with

$$\lambda_1(\mathcal{L})^2 \geq \left[\left(\frac{7\zeta(3)}{9\zeta(2)} \cdot \frac{\text{vol}(\mathcal{L})}{\omega_n} \right)^{2/n} \right] \approx n/2\pi e \cdot \text{vol}(\mathcal{L})^{2/n}.$$

Similarly, we can show the existence of a lattice with a good smoothing parameter and covering radius in any genus.

Theorem 2 (General smoothing). *For any integral genus \mathcal{G} in dimension $n \geq 6$ such that $\text{rk}_p(\mathcal{G}) \geq 6$ for all primes p , constants $C = 26.1$ and $0 < c \leq 1$, and $\epsilon \geq C \cdot (ce)^{-n} \cdot \text{vol}(\mathcal{G})^{-1}$, we have*

$$\Pr_{[\mathcal{L}] \leftarrow \mathcal{D}(\mathcal{G})} \left[\eta_\epsilon(\mathcal{L}^*) \leq \frac{1}{c} \cdot \left(\frac{C \cdot \text{vol}(\mathcal{L}^*)}{\epsilon} \right)^{1/n} \right] > 1 - c^n.$$

In particular, there exists a lattice $\mathcal{L} \in \mathcal{G}$ such that $\eta_\epsilon(\mathcal{L}^) \leq (C \cdot \text{vol}(\mathcal{L}^*)/\epsilon)^{1/n}$.*

The proofs of Theorems 1 and 2 are stated in Sect. 4.3 after some preliminary definitions and results in Sects. 4.1 and 4.2.

Theorem 3 (General covering radius). *For any integral genus \mathcal{G} in dimension $n \geq 6$ such that $\text{rk}_p(\mathcal{G}) \geq 6$ for all primes p , and constants $C = 26.1$ and $e^{-1}(2C/(3 \text{vol}(\mathcal{G})))^{1/n} \leq c \leq 1$, we have*

$$\Pr_{[\mathcal{L}] \leftarrow \mathcal{D}(\mathcal{G})} \left[v(\mathcal{L}^*) \leq \frac{1}{c} \cdot \left(\sqrt{n/2\pi} + 1 \right) \cdot \left(\frac{2}{3} C \cdot \text{vol}(\mathcal{L}^*) \right)^{1/n} \right] > 1 - c^n.$$

In particular, when additionally $n \geq 7$, there exists a lattice $\mathcal{L} \in \mathcal{G}$ such that

$$v(\mathcal{L}^*) \leq \left(\sqrt{n/2\pi} + 1 \right) \cdot \left(\frac{2}{3} C \cdot \text{vol}(\mathcal{L}^*) \right)^{1/n} \approx \sqrt{e} \cdot \sqrt{n/2\pi e} \cdot \text{vol}(\mathcal{L}^*)^{1/n}.$$

Proof. We combine Theorem 2 for $\epsilon = \frac{3}{2}$ and Lemma 1. The constraint on c is equivalent to the constraint on $\epsilon = \frac{3}{2}$ in Theorem 2. For the existence claim note that if $n \geq 7$ then $e^{-1}(2C/(3 \text{vol}(\mathcal{G})))^{1/n} < 1$ and the result follows from the strictly positive probability for $c = 1$. □

Remark 6. We expect that the condition on $\text{rk}_p(\mathcal{G})$ could be removed at the cost of a minor loss in the bound and a more tedious proof. Note that the condition is not satisfied by a genus with $\text{scale}(\mathcal{G}) > 1$, however one can always circumvent this by first normalizing the genus before applying the result.

Remark 7. By choosing $c = 3^{-n}$ in Theorems 1 to 3 we get a probability of strictly more than $\frac{2}{3}$ for each property. In particular, this implies the existence of a lattice $\mathcal{L} \in \mathcal{G}$ having a good packing density and good dual smoothing and covering

4.1 Computing Siegel’s Mass Formula

What makes testing equivalence over \mathbb{Z}_p easy is the fact that we can efficiently (block) diagonalize forms over \mathbb{Z}_p .

Lemma 9 ([6, p.370]). For $p \neq 2$ every integral form $G \in \mathcal{S}_n^{>0}(\mathbb{Z})$ is \mathbb{Z}_p -equivalent to a diagonal matrix. For $p = 2$ every integral form $G \in \mathcal{S}_n^{>0}(\mathbb{Z})$ is \mathbb{Z}_2 -equivalent to a block diagonal matrix with blocks

$$(qx), \quad \begin{pmatrix} qa & qb \\ qb & qc \end{pmatrix},$$

where q is a power of 2, a and c are divisible by 2, but x, b and $d = ac - b^2$ are not. As a corollary, for any prime p the form $G \in \mathcal{S}_n^{>0}(\mathbb{Z})$ is equivalent over \mathbb{Z}_p to a decomposition

$$G_1 \oplus pG_p \oplus p^2G_{p^2} \oplus \dots \oplus qG_q \oplus \dots$$

where $p \nmid \det(G_q)$ for all $q = p^i$, all but a finite number of the G_{p^i} have dimension 0, and each G_q is (block) diagonalized.

The (block) diagonalization is not necessarily unique but can be made canonical with some additional rules [6]. Testing for \mathbb{Z}_p -equivalence then simply becomes testing for equality. Note that because $\text{scale}(\cdot)$ is a genus invariant we can normalize (all forms in) a genus just as we can normalize individual forms. Then for any form in a normalized genus \mathcal{G} the first block G_1 has by construction a nonzero dimension which coincides precisely with having a p -rank $\text{rk}_p(\mathcal{G}) \geq 1$. For our main results we will require that $\text{rk}_p(\mathcal{G}) \geq 6$, which simply states that the first block isn't too small.

Recall that the k -th coefficient of Siegel's mass formula computes the expected number of integer solutions to $x^\top Gx$ for a random form G in a fixed genus. By a local-global principle the number of such solutions is related to the density of such solutions over the localization \mathbb{Z}_p .

Definition 9 (Local density [26]). For an integral form $G \in \mathcal{S}_n^{>0}(\mathbb{Z})$, prime p and integers $k, j \geq 0$ we denote

$$N_G(k \bmod p^j) := |\{x \in (\mathbb{Z}/p^j\mathbb{Z})^n : x^\top Gx \equiv k \bmod p^j\}|.$$

for the number of distinct solutions of $x^\top Gx = k \bmod p^j$. The average number of solutions over all values $k = 0, \dots, p^j - 1$ is given by $p^{(n-1)j}$ and we denote

$$\delta_G(k \bmod p^j) := \frac{N_G(k \bmod p^j)}{p^{(n-1)j}}$$

for the relative density of solutions. For $k \geq 1$ the following limit exists and is finite

$$\delta_{G,p}(k) := \lim_{j \rightarrow \infty} \delta_G(k \bmod p^j).$$

For $k = 0$ the limit does not always exist so we define

$$\delta_{G,p}(0) := \limsup_{j \rightarrow \infty} \delta_G(0 \bmod p^j),$$

which might be ∞ . We call $\delta_{G,p}(k)$ the local density over \mathbb{Z}_p at k .

The number of local solutions and therefore the density over \mathbb{Z}_p is invariant under \mathbb{Z}_p -equivalence. We therefore also denote $\delta_{\mathcal{G},p}(k) := \delta_{G,p}(k)$ for any form G in the genus \mathcal{G} . We also consider one last local density over the reals, which is often also called ‘the prime at infinity’ or at -1 .

Definition 10 (Local density at $p = \infty$). *For an integral genus \mathcal{G} of dimension n , and any $k \geq 1$ we denote*

$$\delta_{\mathcal{G},\infty}(k) = \text{vol}(\mathcal{G})^{-1} \cdot \frac{1}{2} n \omega_n k^{n/2-1},$$

for the local density over the reals \mathbb{R} (or $p = \infty$).

We can now state the local-global result of Siegel that allows us to express the coefficients of Siegel’s mass formula in terms of the local densities.

Theorem 8 (Siegel [26]). *Let \mathcal{G} be a genus of dimension $n \geq 3$, and let $\Theta_{\mathcal{G}}(q) = 1 + \sum_{k=1}^{\infty} N_{\mathcal{G}}(k)q^k$ be the expected theta series over \mathcal{G} . Then for all $k \geq 1$ we have*

$$N_{\mathcal{G}}(k) = \prod_{p=2,3,5,\dots,\infty} \delta_{\mathcal{G},p}(k).$$

This product converges and is 0 if and only if at least one of the factors is 0. Furthermore, this product is efficiently computable given the prime factorization of k and $\text{vol}(\mathcal{G})^2$.

Recall that to determine if two integral forms lie in the same genus we only have to compute something for each prime divisor of $2 \text{vol}(\mathcal{G})^2$, as they are automatically equivalent over the other primes (if their volume matches). We have a similar property here that we only have to compute the local densities for $p = \infty$ and the primes dividing $2k \text{vol}(\mathcal{G})^2$. For these primes we can get a (block) diagonalized representative over \mathbb{Z}_p , and from this there are efficient recursive formulas to compute the local density $\delta_{\mathcal{G},p}(k)$ (see [14]). For the other primes the local density is easy to express and their total infinite product can be computed efficiently using a series identity.

Siegel’s Theorem is also valid when restricting to primitive (global and local) solutions which could in theory give slightly better packing bounds. For simplicity we do not consider this case.

Remark 8. These formulas are implemented in Sagemath [28] by Hanke [14] and others. For a `QuadraticForm` object `Q`, one can call `Q.local_density(p, k)` to compute $\delta_{Q,p}(k)$.⁶ Furthermore, the whole product at $k \geq 1$ is computed

⁶ The current Sagemath implementation for computing the local density $\delta_{Q,p}(k)$ at $p = 2$ follows a naive brute-force approach and therefore becomes infeasible to compute for dimensions as low as $n \geq 8$. In our artifact available at https://github.com/WvanWoerden/siegel_asiacrypt_artifact we supply a patch that resolves this issue and we aim at integrating this fix into Sagemath.

as `Q.siegel_product(k)`. Note, for lattices with even parity this actually computes the local density for $Q/2$ because Sagemath uses a different normalization. These functions allow to compute Siegel’s mass formula explicitly and get better concrete bounds directly based on Lemmas 2 and 4.

4.2 Bounding the Local Densities

We will use Theorem 8 to bound the coefficients $N_G(k)$ of the expected theta series over the genus \mathcal{G} . We will show that under light conditions the local density at each finite prime is bounded sufficiently such that the magnitude of the product of local densities is mostly driven by $\delta_{G,\infty}$.

The general idea is to use the orthogonal decomposition we get from Lemma 9 and show that if we have $G = G_1 \oplus G_2$ we only need to bound the local density of G_1 to bound the local density of G . The reason for this is that all solutions to $x^\top Gx = k \pmod{p^j}$ come from solutions $x_i^\top G_i x_i = k_i \pmod{p^j}$ for $k_1 + k_2 = k \pmod{p^j}$. The local densities of G are thus an averaged out version of the local densities of G_1 and G_2 .

Lemma 10 (Decompose and conquer). *Let p be a prime, G a form of dimension n over the p -adic integers, and suppose that $G = G_1 \oplus G_2$ can be written as an orthogonal sum of non-trivial G_1 and G_2 . Then for a constant $C > 0$ we have*

$$\forall \text{ integers } k' \geq 0, \delta_{G_1,p}(k') \leq C \implies \forall \text{ integers } k \geq 0, \delta_{G,p}(k) \leq C.$$

Proof. Note that we can express the number of solutions of G in terms of G_1 and G_2 as follows

$$N_G(k \pmod{p^j}) = \sum_{\substack{k_1, k_2 \in \mathbb{Z}/p^j\mathbb{Z}, s.t. \\ k_1 + k_2 \equiv k \pmod{p^j}}} N_{G_1}(k_1 \pmod{p^j}) \cdot N_{G_2}(k_2 \pmod{p^j}).$$

Dividing both sides by $p^{(n-1)j}$ gives us

$$\begin{aligned} \delta_G(k \pmod{p^j}) &= p^{-j} \sum_{\substack{k_1, k_2 \in \mathbb{Z}/p^j\mathbb{Z}, s.t. \\ k_1 + k_2 \equiv k \pmod{p^j}}} \delta_{G_1}(k_1 \pmod{p^j}) \cdot \delta_{G_2}(k_2 \pmod{p^j}) \\ &\leq \max_{k_1} \delta_{G_1}(k_1 \pmod{p^j}) \cdot \left(p^{-j} \sum_{k_2 \in \mathbb{Z}/p^j\mathbb{Z}} \delta_{G_2}(k_2 \pmod{p^j}) \right) \\ &= \max_{k_1 \in \mathbb{Z}/p^j\mathbb{Z}} \delta_{G_1}(k_1 \pmod{p^j}) \end{aligned}$$

Taking the limit $j \rightarrow \infty$ we obtain a supremum over $\delta_{G_1,p}(k_1)$ for all $k_1 \geq 0$ each of which is bounded by C . So $\delta_G(k) \leq C$. □

Lemma 11 (Classification of local normal forms [6, Chapter 15.7]). *Let p be a prime. Let $G \in \mathcal{S}_n^0(\mathbb{Z})$ be an integral form of dimension $n \geq 4$ and such that $p \nmid \det(G)$. Then G is equivalent over \mathbb{Z}_p to*

$$(I_n), \text{ or } \left[\begin{array}{c|c} u & \\ \hline & I_{n-1} \end{array} \right], \text{ with } \left(\frac{u}{p} \right) = -1, \text{ when } p \text{ is an odd prime, or}$$

$$\left[\begin{array}{c|c} -I_3 & \\ \hline & I_{n-3} \end{array} \right], \text{ or } \left[\begin{array}{cc|c} a & 0 & \\ \hline 0 & b & \\ & & I_{n-2} \end{array} \right], \text{ with } a, b \in \{\pm 1, \pm 3\}, \text{ when } p = 2, \text{ par}(G) = 1, \text{ or}$$

$$\left[\begin{array}{c|c|c|c} B & & & \\ \hline & B & & \\ \hline & & A & \\ \hline & & & \ddots \\ \hline & & & & A \end{array} \right], \text{ or } \left[\begin{array}{c|c|c} B & & \\ \hline & A & \\ \hline & & \ddots \\ \hline & & & A \end{array} \right], \text{ when } p = 2, \text{ and } \text{par}(G) = 2, \text{ where}$$

$$A := \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad B := \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}.$$

Proof. This follows from the classification of a complete set of invariants for \mathbb{Z}_p -equivalence extensively discussed in [6, Chapter 15.7]. We shortly repeat it here for the case that $p \nmid \det(G)$. If $p \neq 2$ then the \mathbb{Z}_p equivalence is fully determined by the dimension and the *sign* given by the Legendre symbol $\left(\frac{\det(G)}{p} \right) \in \{\pm 1\}$. For any dimension n we thus have two cases which can be represented by the forms in the statement.

The case $p = 2$ is a bit more complicated, first the sign is 1 if $\det(G) = \pm 1 \pmod 8$ and -1 if $\det(G) = \pm 3 \pmod 8$. Then we have the parity of G which is 1 if there is at least one odd entry on the diagonal, and otherwise 2. If the parity is 1 then we can fully diagonalize the form and we can consider the *oddity* t that is the sum of the diagonal modulo 8. Note that $t = n \pmod 2$ as the diagonal consists of odd entries, so there are 4 possibilities left. Combined with the sign there are thus 8 pairs of values for each dimension n and one can check that each of those are attained by the representatives in the statement.

Finally, for parity 2 we only have to consider the sign, giving the two cases in the statement. □

Now if we assume that $\text{rk}_p(G) \geq 4$ then we can assume without loss of generality that our form G takes the shape $G = G_1 \oplus G_2$ where G_1 is one of the forms given in Lemma 11. What remains is to bound the local densities of these forms. For this we have to make a distinction between the prime $p = 2$ and primes $p \geq 3$.

Lemma 12 ($p = 2$). *For all $k \geq 0$ we have $\delta_{B,2}(k) \leq 3$ and $\delta_{I_2,2}(k) \leq 2$.*

Proof. For I_2 and $k \geq 1$ Milnor [19, Lemma 9.1, p. 43] states that $\delta_{I_2,2}(k) \in \{0, 2\}$ and thus is bounded by 2. What remains is the case $k = 0$. For every solution to $x^2 + y^2 \equiv 0 \pmod{2^j}$ for $j \geq 3$ one can verify that necessarily $x, y \equiv 0 \pmod{2^{\lfloor j/2 \rfloor}}$. This leaves at most $(2^{j-\lfloor j/2 \rfloor})^2 \leq 2^{j+1}$ solutions and thus a density of at most $2^{j+1}/2^j = 2$. We conclude that $\delta_{I_2,2}(k) \leq 2$ for all $k \geq 0$.

We now consider B , i.e., the number of solutions to the equation $2x^2 + 2xy + 2y^2 \equiv k \pmod{2^j}$ for $j \geq 3$. Clearly $\delta_{B,2}(k) = 0$ if $k \equiv 1 \pmod 2$. So we assume

that $k = 2k'$ and normalize by 2 to obtain the equation $f(x, y) = x^2 + xy + y^2 = k' \pmod{2^{j-1}}$. Note that if $2 \mid x, y$, then $4 \mid x^2 + xy + y^2$, and thus we need $k' \equiv 0 \pmod{4}$. Else either $2 \nmid x$ or $2 \nmid y$, and we see modulo 2 that $k' \equiv 1 \pmod{2}$. If $k' \equiv 2 \pmod{4}$ there are thus no solutions. We now consider the case that $k' \equiv 1 \pmod{2}$. In this case the Jacobian $(2x + y, 2y + x) \equiv (y, x) \pmod{2}$ of f at any solution (x, y) is nonzero modulo 2. So by a quantitative Hensel's Lemma every solution modulo 2 lifts to precisely 2^{j-1} solutions modulo 2^j , i.e., the density remains unchanged. One can simply count 3 solutions $(0, 1), (1, 0), (1, 1)$ modulo 2 and thus we have a density of 1.5 for all $k' \equiv 1 \pmod{2}$. Now we consider the case that $k' \equiv 0 \pmod{4}$. Note that x, y are both divisible by 2 and thus we can divide both sides of the equation by 4. The density of the number of solutions is thus equal to that of $f(x, y) \equiv k'/4 \pmod{2^{j-2}}$. More generally we can divide by a power of 4 until we obtain a number equal to ± 1 or 2 modulo 4. By the previous result we thus obtain that the density is 1.5 if $v_2(k') \equiv 0 \pmod{2}$, and 0 if $v_2(k') \equiv 1 \pmod{2}$. Lastly, for $k' = 0$ and $j = 2j'$ we have the 2^j solutions $(a \cdot 2^{j'}, b \cdot 2^{j'}) \pmod{2^j}$, and thus a density of 1. Note that we have only shown lower bounds for the densities, but one can quickly verify that we have accounted for all solutions as for $j = 2j'$ there are 2^{j-2i-1} elements in $1, \dots, 2^j - 1$ with valuation $2i$, and thus we obtain a total density modulo 2^j of

$$1 + \sum_{i=0}^{j'-1} 2^{2j'-2i-1} \cdot \frac{3}{2} = 2^j.$$

To conclude we note that the density scales by a factor 2 after scaling back. \square

Lemma 13 ($p \geq 3$). *Let p be an odd prime and consider the 6-dimensional form $D_u = uI_1 \oplus I_5$ with $\left(\frac{u}{p}\right) \in \{\pm 1\}$. Then for all $k \geq 0$ we have*

$$\delta_{D_u,p}(k) \leq \frac{1 - p^{-3}}{1 - p^{-2}}.$$

Additionally,

$$\prod_{p'=3,5,7,\dots} \frac{1 - p'^{-3}}{1 - p'^{-2}} = \frac{6\zeta(2)}{7\zeta(3)} \leq 1.173.$$

Proof. Note that $\det(D_u) = u$, and let $\epsilon = \left(\frac{-u}{p}\right) \in \{\pm 1\}$. Let $k = p^l v \geq 1$ where p^l is the highest power of p dividing k . Then by [26, Hilfssatz 16,p.544] we have for $j > l$ that

$$\delta_{D_u,p}(p^l v \pmod{p^j}) = (1 - \epsilon p^{-3})(1 + \epsilon p^{-2} + \epsilon^2 p^{-4} + \dots + \epsilon^l p^{-2l}).$$

Note that the above local density is maximized if $\epsilon = 1$ and $l \rightarrow \infty$, i.e., we have

$$\delta_{D_u,p}(p^l v \pmod{p^j}) \leq (1 - p^{-3}) \sum_{i=0}^{\infty} p^{-2i} = \frac{1 - p^{-3}}{1 - p^{-2}},$$

and thus in particular $\delta_{D_{u,p}}(k) \leq \frac{1-p^{-3}}{1-p^{-2}}$. Furthermore, because the density only depends on the largest power p^l dividing $k > 0$ we also have $\delta_{D_{u,p}}(0) = \lim_{l \rightarrow \infty} \delta_{D_{i,p}}(p^l) = \frac{1-\epsilon p^{-3}}{1-\epsilon p^{-2}} \leq \frac{1-p^{-3}}{1-p^{-2}}$. Finally, we use the identity $\prod_{p=2,3,5,\dots} (1 - p^{-i}) = 1/\zeta(i)$ for $i \geq 2$, and that $\frac{1-2^{-3}}{1-2^{-2}} = 7/6$. \square

Corollary 4 (Finite places are bounded). *For any integral genus \mathcal{G} of dimension $n \geq 6$ such that $\text{rk}_p(\mathcal{G}) \geq 6$ for all primes p , we have for all $k \geq 0$*

$$\prod_{p=2,3,5,\dots} \delta_{\mathcal{G},p}(k) \leq \frac{18\zeta(2)}{7\zeta(3)} < 3.52.$$

Proof. Let G be a form in \mathcal{G} and consider a finite prime p . Because $\text{rk}_p(\mathcal{G}) \geq 6$ we assume by Lemma 9 without loss of generality that G decomposes as $G = G_1 \oplus pG_2$ with $p \nmid \det(G_1)$ and $\dim(G_1) \geq 6$. For $p \neq 2$ an odd prime Lemma 11 shows that we can assume without loss of generality that $G_1 = D_u \oplus I_l$ where $D_u = uI_1 \oplus I_5$ for a unit $u \in \mathbb{Z}_p$. Lemma 13 gives that $\delta_{D_{u,p}}(k') \leq \frac{1-p^{-3}}{1-p^{-2}}$ for all $k' \geq 0$ and thus we can conclude that $\delta_{G,p}(k) \leq \frac{1-p^{-3}}{1-p^{-2}}$ by Lemma 10.

For $p = 2$ we either have $G_1 = I_2 \oplus G'_1$ or $G_1 = B \oplus G'_1$ and we can again conclude that $\delta_{G,2}(k) \leq 3$ by Lemmas 12 and 10. \square

4.3 Proving the Main Result

Taking Theorem 8 and Corollary 4 together gives a bound on the coefficients $N_{\mathcal{G}}(k)$ of the expected theta series over a genus \mathcal{G} . This bound is sufficient to prove our main results.

Proof (Theorem 1). Let \mathcal{G} and $0 < c \leq 1$ be as in the theorem statement and let $\Theta_{\mathcal{G}}(q) = 1 + \sum_{k=1}^{\infty} N_{\mathcal{G}}(k)q^k$ be the expected theta series of \mathcal{G} . By Theorem 8, Corollary 4 and Lemma 10 we have for all $k \geq 1$ that

$$N_{\mathcal{G}}(k) = \prod_{p=2,3,\dots,\infty} \delta_{\mathcal{G},p}(k) \leq \frac{9\zeta(2)}{7\zeta(3)} \cdot \frac{n\omega_n}{\text{vol}(\mathcal{G})} \cdot k^{n/2-1}.$$

Let $\lambda = \left[c^2 \cdot \left(\frac{7\zeta(3)}{9\zeta(2)} \omega_n^{-1} \text{vol}(\mathcal{G}) \right)^{2/n} \right]$, then using the inequality $j^k \leq \int_{j-\frac{1}{2}}^{j+\frac{1}{2}} t^k dt$ for

$k \geq 1$ we get

$$\sum_{k=1}^{\lambda-1} N_{\mathcal{G}}(k) < \frac{9\zeta(2)}{7\zeta(3)} \cdot \frac{n\omega_n}{\text{vol}(\mathcal{G})} \cdot \int_0^{\lambda-\frac{1}{2}} t^{n/2-1} dt = \frac{18\zeta(2)}{7\zeta(3)} \cdot \frac{\omega_n}{\text{vol}(\mathcal{G})} \cdot (\lambda - \frac{1}{2})^{n/2},$$

which by the choice of λ is bounded by $2c^n$. The result then follows from Lemma 2. \square

Proof (Theorem 2). Let \mathcal{G} be as in the theorem statement and let $\Theta_{\mathcal{G}}(q) = 1 + \sum_{k=1}^{\infty} N_{\mathcal{G}}(k)q^k$ be the expected theta series of \mathcal{G} . By Theorem 8, Corollary 4 and Lemma 10 we have for all $k \geq 1$ that

$$N_{\mathcal{G}}(k) = \prod_{p=2,3,\dots,\infty} \delta_{\mathcal{G},p}(k) \leq \frac{9\zeta(2)}{7\zeta(3)} \cdot \frac{n\omega_n}{\text{vol}(\mathcal{G})} \cdot k^{n/2-1}.$$

Let $C = 26.1$, $\varepsilon \geq C \cdot (ce)^{-n} \cdot \text{vol}(\mathcal{G})^{-1}$ and $s = \frac{1}{c} \cdot \left(\frac{C}{\varepsilon \cdot \text{vol}(\mathcal{G})}\right)^{1/n} \leq e$. Then we have

$$\begin{aligned} \Theta_{\mathcal{G}}(\exp(-\pi s^2)) - 1 &= \sum_{k=1}^{\infty} N_{\mathcal{G}}(k) \cdot \exp(-\pi s^2 k) \\ &\leq \sum_{k=1}^{\infty} \frac{9\zeta(2)}{7\zeta(3)} \cdot \frac{n\omega_n}{\text{vol}(\mathcal{G})} \cdot k^{n/2-1} \cdot \exp(-\pi s^2 k) \\ &= \frac{9\zeta(2)}{7\zeta(3)} \cdot \frac{n\omega_n}{\text{vol}(\mathcal{G})} \cdot \text{Li}_{1-n/2}(\exp(-\pi s^2)) = (*). \end{aligned}$$

Now by Lemma 7 and using that $\omega_n = \frac{\pi^{n/2}}{\Gamma(n/2+1)}$ we get

$$\begin{aligned} (*) &\leq 7.39 \cdot \frac{9\zeta(2)}{7\zeta(3)} \cdot \frac{n}{\text{vol}(\mathcal{G})} \cdot \frac{\pi^{n/2}}{\Gamma(n/2+1)} \cdot \Gamma(n/2) \cdot (\pi s^2)^{-n/2} \\ &= 7.39 \cdot \frac{9\zeta(2)}{7\zeta(3)} \cdot \text{vol}(\mathcal{G})^{-1} \cdot 2 \cdot s^{-n} < C \cdot \text{vol}(\mathcal{G})^{-1} \cdot s^{-n} = c^n \varepsilon. \end{aligned}$$

So $\Theta_{\mathcal{G}}(\exp(-\pi s^2)) < 1 + c^n \varepsilon$ and we can conclude the proof by applying Lemma 4, where we recall that $\text{vol}(\mathcal{L}^*) = \text{vol}(\mathcal{G})^{-1}$. □

5 Applications

Instantiating [11] Without Increasing the Geometric Gap. The LIP framework introduced in [11] turns a decodable lattice \mathcal{L} into a Key Encapsulation Scheme based on the hardness of distinguishing LIP between two auxiliary lattices $\mathcal{L}_1, \mathcal{L}_2$ in the same genus. The concrete hardness of this distinguishing problem is directly related to the geometry of these lattices, in particular, assuming the lattices are normalized to have determinant 1, the best known attacks seems to be driven by the *gap* $\max\{\text{mk}(\mathcal{L})/\lambda_1(\mathcal{L}), \text{mk}(\mathcal{L}^*)/\lambda_1(\mathcal{L}^*)\}$ between their first minimum (or their dual’s) and the Minkowski bound. In the example instantiation given in [11], an efficiently decodable lattice with primal and dual distance, and decoding gap bounded by f , leads to auxiliary lattices with geometric gaps bounded by $O(f^3)$, and thus this leads to a significant reduction in concrete security. The authors already hint that knowledge of good lattices in the same genus can decrease this blowup in the construction. Here we show how Theorem 1 could be used to only have a constant blowup, from f to $O(f)$.

Lemma 14. *Let \mathcal{L} be any $n \geq 6$ -dimensional lattice with primal, dual and efficient decoding gap bounded by $O(f)$ and such that $\text{rk}_p(\mathcal{L}) \geq 6$ for all primes p . Then there exists a KEM which security reduces to a $2n$ -dimensional instance of distinguish LIP with geometric gaps bounded by $O(f)$.*

Proof. Let $g \in \mathbb{Z}_{>0}$ be some scaling factor to be determined later. We need to construct an appropriate pair of lattices $\mathcal{L}_S, \mathcal{L}_Q$ to instantiate [11, Theorem 5.2]. Let $\mathcal{L}' \in \text{gen}(\mathcal{L})$ be a lattice with $\text{gap}(\mathcal{L}') = O(1)$ as exists according to Theorem 1. We set $\mathcal{L}_S := g\mathcal{L} \oplus (g+1)\mathcal{L}$ as our well decodable lattice with decoding radius $\rho' = O(g/f \cdot \text{gh}(\mathcal{L}))$, and $\mathcal{L}_Q = \mathcal{L}' \oplus g(g+1)\mathcal{L}'$ as our auxiliary lattice that has a dense sublattice $\mathcal{L}' \subset \mathcal{L}_Q$. Note that $\text{gap}(\mathcal{L}_S) = O(f)$ and $\text{gap}(\mathcal{L}_Q) = O(g)$. To satisfy the conditions of [11, Theorem 5.2] we require that $\eta_{\frac{1}{2}}(\mathcal{L}') \leq \rho'/(2\sqrt{2n})$. Now note that because $\text{gap}(\mathcal{L}') = O(1)$ we have

$$\eta_{\frac{1}{2}}(\mathcal{L}') \leq \eta_{2^{-n}}(\mathcal{L}') \leq \frac{\sqrt{n}}{\lambda_1((\mathcal{L}')^*)} = \theta(\det(\mathcal{L}')^{1/n}).$$

Furthermore, we have $\rho'/2\sqrt{2n} = \theta(g/f \cdot \text{gh}(\mathcal{L})/\sqrt{n})$ and thus it is sufficient to pick g that satisfies

$$\theta(\det(\mathcal{L}')^{1/n}) \leq \theta(g/f \cdot \text{gh}(\mathcal{L})/\sqrt{n}) = \theta(g/f \cdot \det(\mathcal{L}')^{1/n}),$$

from which it is clear that $g = \Theta(f)$ suffices. We conclude by noting that then $\text{gap}(\mathcal{L}_S) = O(f)$ and $\text{gap}(\mathcal{L}_Q) = \Theta(g) = O(f)$. □

The encryption scheme from [4] based on the same framework benefits from the same improvement. For the signature scheme from [11] we also improve the blowup from $O(f^2)$ to $O(f)$.

Lemma 15. *Let \mathcal{L} be any $n \geq 6$ -dimensional lattice with primal, dual and efficient Gaussian sampling gap bounded by $O(f)$ and such that $\text{rk}_p(\mathcal{L}) \geq 6$ for all primes p . Then there exists a signature scheme which security reduces to a $2n$ -dimensional instance of distinguish LIP with geometric gaps bounded by $O(f)$.*

Proof. The proof follows similarly as the construction in [11] and the proof of Lemma 14, but with $\mathcal{L}_S := g\mathcal{L} \oplus (g+1)\mathcal{L}$ and $\mathcal{L}_{Q^{-1}} = \mathcal{L}' \oplus g(g+1)\mathcal{L}'$. □

Instantiating for \mathbb{Z}^n . Another interesting concurrent work [3], that arrived shortly after [11], introduces some cryptographic constructions based on LIP for \mathbb{Z}^n . In this work the authors raised some open questions about the instantiability of their schemes, as for this they require the existence of a lattice with certain geometric properties in the same genus as \mathbb{Z}^n . In particular, it is asked if there exist a lattice \mathcal{L} in the genus of \mathbb{Z}^n that have $\lambda_1^2(\mathcal{L}) \geq \Omega(n/\log n)$, or $\eta_\epsilon(\mathcal{L}) \leq \eta_\epsilon(\mathbb{Z}^n)/\sqrt{\log n} \approx O(\sqrt{\log(1/\epsilon)/\log n})$ for $\epsilon < n^{-\omega(1)}$. Note that the genus of \mathbb{Z}^n is that of all odd unimodular lattices of dimension n . Therefore, the first question is in fact already answered by [19], which shows that there exists an

odd unimodular lattice with $\lambda_1(\mathcal{L})^2 \geq \Omega(n) > \Omega(n/\log n)$ for growing n . For the second, we can instantiate Theorem 2 with for example $\epsilon = 2^{-n}$, which implies the existence of \mathcal{L} in the genus of \mathbb{Z}^n with $\eta_\epsilon(\mathcal{L}) = O(1) < \eta_\epsilon(\mathbb{Z}^n)/\sqrt{\log(n)} \approx \Theta(\sqrt{n/\log(n)})$.

Instantiation of the Encryption Scheme of [1]. In [1] a public-key encryption scheme based on LIP is presented. To instantiate the scheme however an auxiliary lattice is needed that satisfies some geometric properties. In the instantiation the authors would like to use unimodular lattices and they conjecture that for $n \geq 85$ there exists at least one unimodular lattice of dimension n such that $\lambda_1(\mathcal{L}) \geq \sqrt[4]{72n}$. We see that Lemma 5 is enough to answer this conjecture positively, and even show the existence of much better packings. More generally, given a decodable lattice \mathcal{L} of dimension n they require another lattice $\mathcal{L}' \in \mathcal{G}$ with large minimum distance $\lambda_1(\mathcal{L}')$. For most genera Theorem 1 shows the existence of such a lattice.

6 Open Questions

We discuss some open questions that could be interesting for further work.

In this work we have restricted ourselves for simplicity to unstructured lattices. However, in cryptography we often use structured module lattices to decrease storage and improve efficiency. The genus theory extends to these cases and mass formulas can also be extended in certain settings [18,30]. One has to be careful when considering the existing literature as it often considers the quadratic form $B^\top B$ for some module-lattice basis B . Except for the case of totally real number fields, the matrix $B^\top B$ does not correspond to the geometry of the module lattice via the canonical embedding. For example, in the common case of CM-fields, which is relevant for HAWK [10], this information is instead captured by the Hermitian form B^*B . Luckily, most of the genus theory and the Smith-Siegel-Minkowski mass formula has been generalized to hermitian forms over CM-fields by [18]. We therefore expect that Siegel’s mass formula can also be generalized, leading to similar claims as in this work for module lattices over CM-fields.

Furthermore, the results of Siegel go further than just the expectation of the number of vectors of some squared norm. In fact, they can also count the expected number of higher rank constellations of multiple vectors with certain norm and inner product within each lattice. More precisely, for forms $G \in \mathcal{S}_n^{>0}(\mathbb{Z})$ and $K \in \mathcal{S}_m^{>0}(\mathbb{Z})$ for $1 \leq m \leq n$, it counts the expected number of solutions $X \in \mathbb{Z}^{n \times m}$ such that $X^\top GX = K$ when varying G over a genus. This might open up the possibility to study more advanced geometric properties of random lattices over a genus.

Acknowledgements. W. van Woerden was supported by the CHARM ANR-NSF grant (ANR-21-CE94-0003).




References

1. Ackermann, L., Roux-Langlois, A., Wallet, A.: Public-key encryption from the lattice isomorphism problem. WCC 2024: The Thirteenth International Workshop on Coding and Cryptography (2024), extended abstract available at https://wcc2024.sites.dmi.unipg.it/WCC_proceedings.pdf. Full version to appear.
2. Aono, Y., Espitau, T., Nguyen, P.Q.: Random lattices: Theory and practice (2019), available at https://espitau.github.io/bin/random_lattice.pdf
3. Bennett, H., Ganju, A., Peetathawatchai, P., Stephens-Davidowitz, N.: Just how hard are rotations of \mathbb{Z}^n ? algorithms and cryptography with the simplest lattice. In: EUROCRYPT 2023: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 252–281. Springer (2023)
4. Biage, G.C., Zambonin, G., Idalino, T.B., Panario, D., Custodio, R.: A concrete LIP-based KEM with simple lattices. IEEE Access (2024)
5. Chenevier, G.: Statistics for kneser p -neighbors. arXiv preprint [arXiv:2104.06846](https://arxiv.org/abs/2104.06846) (2021)
6. Conway, J.H., Sloane, N.J.A.: Sphere Packings, Lattices and Groups. Springer New York (1999)
7. Conway, J.H., Sloane, N.J.: Low-dimensional lattices. IV. The mass formula. Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences **419**(1857), 259–286 (1988)
8. Debris-Alazard, T., Ducas, L., Resch, N., Tillich, J.P.: Smoothing codes and lattices: Systematic study and new bounds. IEEE Transactions on Information Theory **69**(9), 6006–6027 (2023)
9. Ducas, L., Gibbons, S.: Hull attacks on the lattice isomorphism problem. In: PKC 2023: IACR International Conference on Public-Key Cryptography. pp. 177–204. Springer (2023)
10. Ducas, L., Postlethwaite, E.W., Pulles, L.N., van Woerden, W.: Hawk: Module LIP makes lattice signatures fast, compact and simple. In: ASIACRYPT 2022: International Conference on the Theory and Application of Cryptology and Information Security. pp. 65–94. Springer (2022)
11. Ducas, L., van Woerden, W.: On the lattice isomorphism problem, quadratic forms, remarkable lattices, and cryptography. In: EUROCRYPT 2022: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 643–673. Springer (2022)
12. Dutour Sikirić, M., Haensch, A., Voight, J., van Woerden, W.: A canonical form for positive definite matrices. ANTS XIV **4**(1), 179–195 (2020)
13. Feit, W.: Orders of finite linear groups. In: Proceedings of the First Jamaican Conference on Group Theory and its Applications. pp. 9–11. Univ. West Indies, Kingston (1996)
14. Hanke, J.: Local densities and explicit bounds for representability by a quadratic form. Duke Mathematical Journal **124**(2), 351 – 388 (2004)
15. Haviv, I., Regev, O.: On the lattice isomorphism problem. In: Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms. pp. 391–404. SIAM (2014)
16. Hein, J.P.: Orthogonal modular forms: an application to a conjecture of Birch, algorithms and computations. Ph.D. thesis, Dartmouth College Library Press (2016)
17. Hlawka, E.: Zur geometrie der zahlen. Mathematische Zeitschrift **49**(1), 285–312 (1943)

18. Kirschmer, M.: Definite quadratic and hermitian forms with small class number. Habilitation, RWTH Aachen University (2016), available at <https://www.math.rwth-aachen.de/~Markus.Kirschmer/papers/herm.pdf>
19. Milnor, J.W., Husemoller, D., et al.: Symmetric bilinear forms, vol. 5. Springer (1973)
20. Minkowski, H.: Untersuchungen über quadratische Formen: Bestimmung der Anzahl verschiedener Formen, welche ein gegebenes Genus enthält. E. Erlatis (1885)
21. Minkowski, H.: Geometrie der Zahlen. B.G. Teubner (1910)
22. Plesken, W., Pohst, M.: Constructing integral lattices with prescribed minimum. I. Mathematics of computation **45**(171), 209–221 (1985)
23. Plesken, W., Souvignier, B.: Computing isometries of lattices. Journal of Symbolic Computation **24**(3-4), 327–334 (1997)
24. Regev, O., Stephens-Davidowitz, N.: A reverse Minkowski theorem. Annals of Mathematics **199**(1), 1–49 (2024)
25. Serre, J.P.: A Course in Arithmetic. Springer New York (1973)
26. Siegel, C.L.: Über die analytische theorie der quadratischen formen. Annals of Mathematics pp. 527–606 (1935)
27. Siegel, C.L.: A mean value theorem in geometry of numbers. Annals of Mathematics pp. 340–347 (1945)
28. The Sage Developers: SageMath, the Sage Mathematics Software System (Version 10.4) (2024), <https://www.sagemath.org>
29. Watson, G.: Integral Quadratic Forms, By G.L. Watson. Cambridge Tracts in Mathematics and Mathematical Physics, No. 51 (1960)
30. Weil, A.: Sur la formule de siegel dans la théorie des groupes classiques. Acta math **113**(1-87), 2 (1965)
31. Weisfeiler, B.: On the size and structure of finite linear groups (1984), available at <https://arxiv.org/pdf/1203.1960>
32. Wood, D.C.: The computation of polylogarithms. Technical report, University of Kent, Computing Laboratory, University of Kent, Canterbury, UK (1992)



Evasive LWE Assumptions: Definitions, Classes, and Counterexamples

Chris Brzuska¹ , Akin Ünal² , and Ivy K. Y. Woo¹ 

¹ Aalto University, Espoo, Finland
`chris.brzuska@aalto.fi`

² ISTA, Klosterneuburg, Austria

Abstract. The evasive LWE assumption, proposed by Wee [Eurocrypt’22 Wee] for constructing a lattice-based optimal broadcast encryption, has shown to be a powerful assumption, adopted by subsequent works to construct advanced primitives ranging from ABE variants to obfuscation for null circuits. However, a closer look reveals significant differences among the precise assumption statements involved in different works, leading to the fundamental question of how these assumptions compare to each other. In this work, we initiate a more systematic study on evasive LWE assumptions:

- (i) Based on the standard LWE assumption, we construct simple counterexamples against three private-coin evasive LWE variants, used in [Crypto’22 Tsabary, Asiacrypt’22 VWW, Crypto’23 ARYY] respectively, showing that these assumptions are unlikely to hold.
- (ii) Based on existing evasive LWE variants and our counterexamples, we propose and define three classes of plausible evasive LWE assumptions, suitably capturing all existing variants for which we are not aware of non-obfuscation-based counterexamples.
- (iii) We show that under our assumption formulations, the security proofs of [Asiacrypt’22 VWW] and [Crypto’23 ARYY] can be recovered, and we reason why the security proof of [Crypto’22 Tsabary] is also plausibly repairable using an appropriate evasive LWE assumption.

1 Introduction

Resolving a decade-long open problem, Wee [Wee22] constructs a lattice-based ciphertext-policy attribute-based encryption (CP-ABE) for NC1 with parameters independent of the circuit size, implying an optimal broadcast encryption, under a new assumption called the evasive LWE assumption. Roughly, the assumption states that, for any PPT Samp outputting an arbitrary matrix \mathbf{P} and auxiliary information aux containing all coin tosses used by Samp ,

if $(\mathbf{B}, \mathbf{P}, \mathbf{s}^T \mathbf{B} + \mathbf{e}_0^T, \mathbf{s}^T \mathbf{P} + \mathbf{e}_1^T, \text{aux}) \approx_c (\mathbf{B}, \mathbf{P}, \$, \$, \text{aux})$ (1)
 then $(\mathbf{B}, \mathbf{P}, \mathbf{s}^T \mathbf{B} + \mathbf{e}_0^T, \mathbf{B}^{-1}(\mathbf{P}), \text{aux}) \approx_c (\mathbf{B}, \mathbf{P}, \$, \mathbf{B}^{-1}(\mathbf{P}), \text{aux})$,
 for a uniformly random matrix $\mathbf{B} \leftarrow_s \mathbb{Z}_q^{n \times m}$, a uniformly random LWE secret $\mathbf{s} \leftarrow_s \mathbb{Z}_q^n$, Gaussian errors $\mathbf{e}_0, \mathbf{e}_1$ of appropriate dimensions, and where $\mathbf{B}^{-1}(\mathbf{P})$

denotes a short Gaussian preimage of \mathbf{P} with respect to \mathbf{B} , i.e. it holds that $\mathbf{B} \cdot \mathbf{B}^{-1}(\mathbf{P}) = \mathbf{P} \bmod q$ and $\mathbf{B}^{-1}(\mathbf{P})$ is short in Euclidean norm. Intuitively, the assumption postulates that, to distinguish LWE samples $\mathbf{s}^T \mathbf{B} + \mathbf{e}_0^T \bmod q$ given a short preimage $\mathbf{B}^{-1}(\mathbf{P})$ as hint, the only thing one can do with this hint is to right-multiply it to $\mathbf{s}^T \mathbf{B} + \mathbf{e}_0^T \bmod q$ to obtain another LWE sample $\mathbf{s}^T \mathbf{P} + \mathbf{e}_1^T \bmod q$ and try to distinguish the latter. We shall call this the *public-coin* evasive LWE assumption, to highlight that all random coins of \mathbf{Samp} are given to the distinguisher via *aux*. Subsequently, public-coin evasive LWE was adapted by [WWW22, HLL24], to prove security of a multi-authority (MA-)ABE scheme without a random oracle and construct a variant of inner-product functional encryption, respectively.

Concurrently and subsequently, a number of works considered what we shall call *private-coin* variants of evasive LWE, where \mathbf{Samp} does not need to provide all its randomness to the distinguisher. These private-coin variants of evasive LWE (sometimes applied along with other new lattice assumptions) have shown to imply further advanced primitives including multi-input (MI-)ABE [ARYY23], CP-ABE for unbounded depth circuits [AKY24], null-iO [VWW22], and witness encryption [Tsa22, VWW22]. Notably, these primitives tend to be believed to be stronger than what is currently achievable by public-coin evasive LWE.

Underlying the name of evasive LWE, however, are significant differences among the actual assumption statements involved in all aforementioned works, in addition to the distinction between public and private-coin. For example, the public-coin evasive LWE variants in [WWW22, HLL24] involve multiple independent matrices and LWE samples $(\mathbf{B}_i, \mathbf{s}_i^T \mathbf{B}_i + \mathbf{e}_i^T \bmod q)$, where the \mathbf{s}_i 's can be identical or different, and [HLL24] further requires a joint preimage where $\mathbf{B}_i^{-1}(\mathbf{P}) = \mathbf{B}_j^{-1}(\mathbf{P})$ for all i, j . In the private-coin world, [VWW22] formulates an evasive LWE assumption where the matrices \mathbf{B}, \mathbf{P} are not explicitly stated in the *if* and *then* joint distributions, whereas [ARYY23] includes \mathbf{B} into both distributions, but not \mathbf{P} . Unfortunately, up to now, we have little understanding as to how these and other modifications affect the strength and plausibility of the assumptions, with the only cryptanalytic insight being a heuristic obfuscation-based counterexample by [VWW22] against some private-coin variants.

1.1 Our Contributions

We initiate a systematic study on evasive LWE assumptions, summarised by three aspects:

Falsifying Existing Private-Coin Variants. We construct counterexamples against subclasses of existing evasive LWE variants. Specifically, we give three examples to show that, each of the three private-coin evasive LWE assumptions stated in [Tsa22, VWW22, ARYY23], respectively, are unlikely to hold. All our counterexamples are conceptually simple, based on the standard LWE assumption and do not rely on obfuscation. Our counterexamples are summarised in Sect. 2.2 and formalised in Sect. 5. More counterexamples are sketched in the full version of this work.

Classifications and Definitions. Based on existing variants together with the essences of our counterexamples, we propose a classification of plausible evasive LWE assumptions. Our proposed three families are summarised below:

1. Public-coin evasive LWE, i.e. the PPT sampler **Samp** does not hide any of its computation from the distinguisher;
2. Private-coin binding evasive LWE, where (i) **Samp** does not input the matrix **B**, and (ii) the matrices **B**, **P** are given to the distinguisher;
3. Private-coin hiding evasive LWE, where (i) **Samp** does not input the matrix **B**, (ii) **B** is not given to the distinguisher, and (iii) the matrix **P** is provably sufficiently hidden from the distinguisher.

For each family we formulate a general definition, suitably capturing all variants in existing works that are not subject to simple counterexamples (see Remark 3 for a discussion of obfuscation-based counterexamples). We summarise the rationale in Sect. 2.3 and give precise definitions in Sect. 4. We hope that this will serve as a first step to provide the community with a language for communicating about evasive LWE assumptions, leaving its intuition unchanged while simultaneously expressing the qualitative differences between the actual assumptions involved.

Implications on Existing Constructions. We show that the assumption instances in the security proofs of [VWW22, ARYY23] fall under our proposed families of plausible evasive LWE, as such, their related constructions remain secure under our assumptions. More concretely, in Sect. 6 we prove that the evasive LWE instances in the proof of [VWW22] satisfy our condition of **P** being sufficiently hidden, thus falling into the private-coin hiding family. The assumption instances of [ARYY23] fall into the private-coin binding family directly by definition. For the proof of [Tsa22], we discuss in Sect. 2.4 why we believe it may be repairable with an alternative and plausible evasive LWE assumption.

2 Overview

In Sect. 2.1 we review a number of evasive LWE variants. In Sect. 2.2 we sketch our counterexamples against three subclasses, which leads to our proposal of three plausible evasive LWE families summarised in Sect. 2.3. For existing works that rely on assumptions affected by our counterexamples, we discuss in Sect. 2.4 to which extent their security proofs may be repairable.

We adopt simplified notation in this overview: Operations are understood to be over \mathbb{Z}_q and we suppress mod q expressions. To denote noise terms, we use curly underline $\underline{\sim}$, e.g. $\underline{\mathbf{s}}^T \mathbf{B}$ means $\mathbf{s}^T \mathbf{B} + \mathbf{e}^T$ where \mathbf{e} is short relative to q . We use $\$$ to refer to uniformly random values, where multiple $\$$ signs in a joint distribution are understood to be independent uniform samples.

2.1 Existing Evasive LWE Variants and Classifications

To aid understanding the differences among the evasive LWE variants below, we provide two partition grids in Fig. 1 which we will cross-reference with.

Public-Coin Evasive LWE. We recall again Wee’s public-coin evasive LWE assumption: For any PPT Samp outputting an arbitrary matrix \mathbf{P} and auxiliary information aux containing all randomness used by Samp ,

$$\text{if } (\mathbf{B}, \mathbf{P}, \underline{\underline{\mathbf{s}^T \mathbf{B}}}, \underline{\underline{\mathbf{s}^T \mathbf{P}}}, \text{aux}) \approx_c (\mathbf{B}, \mathbf{P}, \$, \$, \text{aux}) \quad (2)$$

$$\text{then } (\mathbf{B}, \mathbf{P}, \underline{\underline{\mathbf{s}^T \mathbf{B}}}, \mathbf{B}^{-1}(\mathbf{P}), \text{aux}) \approx_c (\mathbf{B}, \mathbf{P}, \$, \mathbf{B}^{-1}(\mathbf{P}), \text{aux})$$

for uniformly random \mathbf{B} and uniformly random LWE secret \mathbf{s} .¹ Subsequently, Waters, Wee and Wu [WWW22] define a variant consisting of polynomially many pairs of matrices $(\mathbf{B}_i, \mathbf{P}_i)_i$ and their respective LWE samples $\underline{\underline{\mathbf{s}_i^T \mathbf{B}_i}}, \underline{\underline{\mathbf{s}_i^T \mathbf{P}_i}}$. More recently, Hsieh, Lin, and Luo [HLL24] define a public-coin variant consisting also of pairs of $(\mathbf{B}_i, \mathbf{P}_i)_i$, but with LWE samples of the form $(\underline{\underline{\mathbf{s}^T \mathbf{B}_i}}, \underline{\underline{\mathbf{s}^T \mathbf{P}_i}})_i$ sharing the same secret \mathbf{s} , with a structured error distribution instead of a random Gaussian, and further requiring joint preimages satisfying $\mathbf{B}_i^{-1}(\mathbf{P}) = \mathbf{B}_j^{-1}(\mathbf{P})$ for all i, j .

Private-Coin without \mathbf{B}, \mathbf{P} . Vaikunthanathan, Wee and Wichs [VWW22] define the following private-coin evasive LWE assumption: For any PPT Samp which outputs (arbitrary) LWE secret \mathbf{S} , matrix \mathbf{P} , and auxiliary information aux (not necessarily containing all randomness used by Samp),

$$\text{if } (\underline{\underline{\mathbf{S}\mathbf{B}}}, \underline{\underline{\mathbf{S}\mathbf{P}}}, \text{aux}) \approx_c (\$, \$, \text{aux}) \quad (3)$$

$$\text{then } (\underline{\underline{\mathbf{S}\mathbf{B}}}, \mathbf{B}^{-1}(\mathbf{P}), \text{aux}) \approx_c (\$, \mathbf{B}^{-1}(\mathbf{P}), \text{aux})$$

for uniformly random \mathbf{B} . This variant corresponds to the [blue area](#) in Fig. 1b. We observe that, unlike in Eq. (2), in this variant, \mathbf{B} is not included in the if and then distributions, and \mathbf{P} also not necessarily. However, since both aux, \mathbf{P} are generated by Samp , information about \mathbf{P} may or may not be included in the distributions via aux , e.g., Samp could choose $\text{aux} = \mathbf{P}$. Using the above, [VWW22] prove that the well-known GGH15 encodings [GGH15] are secure, implying the existence of null-iO and witness encryption, and recent works also use the above evasive LWE variant to construct SNARG for UP as well as universal computational extractors [MPV24, CM24].

Private-Coin Without \mathbf{P} . Agrawal, Rossi, Yadav and Yamada (ARYY) [ARYY23] defined the following private-coin variant: For any PPT Samp outputting (arbitrary) LWE secret \mathbf{S} , matrix \mathbf{P} , and auxiliary information aux (not necessarily containing all randomness used by Samp),

$$\text{if } (\mathbf{B}, \underline{\underline{\mathbf{S}\mathbf{B}}}, \underline{\underline{\mathbf{S}\mathbf{P}}}, \text{aux}) \approx_c (\mathbf{B}, \$, \$, \text{aux}) \quad (4)$$

$$\text{then } (\mathbf{B}, \underline{\underline{\mathbf{S}\mathbf{B}}}, \mathbf{B}^{-1}(\mathbf{P}), \text{aux}) \approx_c (\mathbf{B}, \$, \mathbf{B}^{-1}(\mathbf{P}), \text{aux})$$

¹ [Wee22] includes an additional matrix \mathbf{A} and LWE samples $\underline{\underline{\mathbf{s}^T \mathbf{A}}}$ in the joint distributions for further expressiveness, which we omit in this overview.

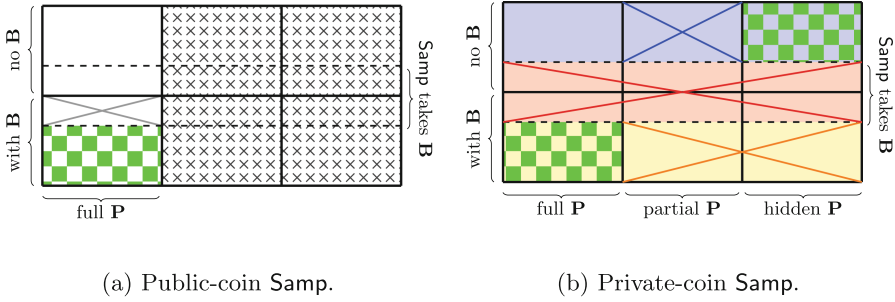


Fig. 1. Partition of evasive LWE assumptions. Background colour {yellow,blue,red} = Assumption {(4),(3),(5)}. Cross in {yellow,blue,red} = Counterexample {1, 2, 3} sketched in Sect. 2.2 and formalised in Sect. 5. Cross in gray = heuristic counterexample. Green checkboxes are the three proposed classes in Sect. 2.3. (Color figure online)

for uniformly random \mathbf{B} . This corresponds to the yellow area in Fig. 1b. We notice that here the matrix \mathbf{B} is included in the distributions, and \mathbf{P} again may or may not be via aux. ARYY show that Eq. (4) implies another private-coin variant, which is almost identical except that aux can be partitioned as $(\text{aux}_1, \text{aux}_2)$, with aux_1 provably pseudorandom, and \mathbf{P} is efficiently computable from aux_2 . The latter is used for proving security of their MI-ABEs², and subsequently also by [AKY24] for proving their CP-ABE for unbounded depth circuits³.

Samp Generates $\mathbf{B}^{-1}(\mathbf{P})$. Concurrent to [VWW22], Tsabary [Tsa22] proposes a similar flavour of assumption for constructing witness encryption. Putting formulation differences aside, Tsabary’s assumption can be summarised as follows: For any PPT Samp which inputs a matrix \mathbf{B} with its trapdoor $\text{td}_{\mathbf{B}}$, and outputs LWE secret \mathbf{S} , target matrix \mathbf{P} , preimages $\mathbf{B}^{-1}(\mathbf{P})$ sampled using $\text{td}_{\mathbf{B}}$, and auxiliary information aux (not necessarily containing all randomness of Samp),

$$\begin{aligned} \text{if } & (\underline{\mathbf{S}}\mathbf{B}, \underline{\mathbf{S}}\mathbf{P}, \text{aux}) && \approx_c & (\$, \$, \text{aux}) & (5) \\ \text{then } & (\underline{\mathbf{S}}\mathbf{B}, \mathbf{B}^{-1}(\mathbf{P}), \text{aux}) && \approx_c & (\$, \mathbf{B}^{-1}(\mathbf{P}), \text{aux}). \end{aligned}$$

This variant covers the red area in Fig. 1b. The if and then distributions are identical to Eq. (3). The crucial difference is, however, that Samp takes the matrix \mathbf{B} as input and samples the preimages $\mathbf{B}^{-1}(\mathbf{P})$ itself, in contrast to other variants where the preimages are provided by the challenger. This formulation necessitates Samp knowing \mathbf{B} and $\text{td}_{\mathbf{B}}$, and as a consequence \mathbf{S}, \mathbf{P} and aux can be arbitrarily correlated with \mathbf{B} .

Summarising from above, we obtain the following factors:

- Public-coin vs. private-coin Samp (Fig. 1a vs. Fig. 1b).
- Whether \mathbf{B} is included in the if and then distributions (left axis of Fig. 1b).

² For their MI-ABE for P, along with a new assumption called extended tensor LWE.

³ Along with a new assumption called circular tensor LWE.

- Whether \mathbf{P} is included in the if and then distributions. We separately consider, informally for now, \mathbf{P} is fully available, partially available, or fully hidden from the distinguisher (bottom axis of Fig. 1b), the last to be expanded later.
- Whether Samp inputs the matrix \mathbf{B} (right axis of Fig. 1b).

The above jointly form a partition of Fig. 1, as marked by the figure labels.⁴

Remark 1 (On Assumption Strength). The classes of samplers Samp in evasive LWE form natural inclusions corresponding to the strength of the assumptions: The largest class contains the most Samp , which can take the most inputs, e.g. \mathbf{B} , and output arbitrary matrices, representing the strongest assumptions. Underlying are various subclasses, e.g. one containing Samp which does not input \mathbf{B} , and one containing Samp which must output \mathbf{P} in plain, each forming weaker assumptions. Similarly, public-coin samplers are a subclass of private-coin ones.

In contrast, including \mathbf{B} in the distributions or not rather makes the assumptions incomparable: Including \mathbf{B} leads to a stronger if condition to be satisfied, but the assumption simultaneously asserts a stronger then statement with \mathbf{B} .

2.2 Counterexamples

In order to invalidate an evasive LWE assumption, our goal is to construct a PPT Samp such that, with respect to Samp , (1) the if statement holds (assuming plausible assumptions), but (2) the then statement does not.

Counterexample 1: Private-Coin with \mathbf{B} and Partial/Hidden \mathbf{P} . We give a counterexample for the case where the distinguisher of if receives the matrix \mathbf{B} , but not \mathbf{P} , corresponding to a subclass of Eq. (4) with a yellow cross in Fig. 1b. The idea is simple: Given \mathbf{B} and $\mathbf{B}^{-1}(\mathbf{P})$, the distinguisher of then can recover \mathbf{P} , so that if \mathbf{P} contains useful information for distinguishing, this can be used by then but not if. Concretely, let Samp return the following:

$$\mathbf{P} = \left(\mathbf{P}_1, \mathbf{P}_2 = \begin{bmatrix} \mathbf{u}^T \\ \mathbf{R} \end{bmatrix} \right), \quad \text{aux} = ()$$

where \mathbf{u} is a short vector such that $\mathbf{P}_1 \mathbf{u} = \mathbf{0}$, i.e. Samp samples \mathbf{P}_1 together with a trapdoor to generate \mathbf{u} , and \mathbf{R} is uniformly random. To see that the the if statement holds, observe that

$$(\mathbf{B}, \underbrace{\mathbf{s}^T \mathbf{B}}_{\text{stat.}}, \underbrace{\mathbf{s}^T \mathbf{P}_1}_{\text{stat.}}, \underbrace{\mathbf{s}^T \mathbf{P}_2}_{\text{stat.}}) \approx (\mathbf{B}, \underbrace{\mathbf{s}^T \mathbf{B}}_{\text{LWE}}, \$, \$) \approx (\mathbf{B}, \$, \$, \$)$$

where the first statistical statement holds since \mathbf{R} and \mathbf{P}_1 are (close to) uniformly random (and unknown), and the second follows by LWE, as \mathbf{B} is uniformly random. For the distinguisher of then, observe that, when given

$$(\mathbf{B}, \mathbf{B}^{-1}(\mathbf{P}_1, \mathbf{P}_2), \underbrace{\mathbf{s}^T \mathbf{P}_1}_{\text{stat.}}, \underbrace{\mathbf{s}^T \mathbf{P}_2}_{\text{stat.}}),$$

⁴ The partition has not taken into account whether the LWE secret \mathbf{S} is generated by Samp . This is to be discussed at the end of Sect. 2.2.

it can compute

$$\mathbf{P}_2 \leftarrow \mathbf{B} \cdot \mathbf{B}^{-1}(\mathbf{P}_2); \quad \begin{bmatrix} \mathbf{u}^\top \\ \mathbf{R} \end{bmatrix} \leftarrow \mathbf{P}_2; \quad \text{Test } \underbrace{\mathbf{s}^\top \mathbf{P}_1}_{\text{LWE}} \cdot \mathbf{u} \approx? \mathbf{0}, \quad (6)$$

where $\underbrace{\mathbf{s}^\top \mathbf{P}_1}_{\text{LWE}} \cdot \mathbf{u}$ would likely not be short when replacing $\underbrace{\mathbf{s}^\top \mathbf{P}_1}_{\text{LWE}}$ with a random vector. Note that the example works also when \mathbf{P}_1 is given in `aux`.

Counterexample 2: Private-Coin Without \mathbf{B} and Partial \mathbf{P} . Next we turn to the case where \mathbf{B} is not available to the distinguisher of if but \mathbf{P} is partially available, corresponding to a subclass of Eq. (3) with a blue cross in Fig. 1b. Here, our `Samp` is almost identical to the previous counterexample, except that now it lets `aux` = \mathbf{P}_1 . Similar to above, for the if condition, we have

$$(\underbrace{\mathbf{s}^\top \mathbf{B}}_{\text{LWE}}, \underbrace{\mathbf{s}^\top \mathbf{P}_1}_{\text{LWE}}, \underbrace{\mathbf{s}^\top \mathbf{P}_2}_{\text{LWE}}, \mathbf{P}_1) \stackrel{\text{stat.}}{\approx} (\$, \underbrace{\mathbf{s}^\top \mathbf{P}_1}_{\text{LWE}}, \$, \mathbf{P}_1) \stackrel{\text{LWE}}{\approx} (\$, \$, \$, \mathbf{P}_1)$$

where the first relation is due to \mathbf{B}, \mathbf{R} being uniform and unknown to the distinguisher. For distinguishing the then distribution, since \mathbf{B} is not contained in the joint distribution any longer, the distinguisher takes an extra step to recover it. Namely, our crucial observation is that when \mathbf{P}_1 has (at least) as many columns as that of \mathbf{B} and \mathbb{Z}_q is a field⁵ then with high probability, \mathbf{B} is fully determined given $(\mathbf{B}^{-1}(\mathbf{P}_1), \mathbf{P}_1)$, and is efficiently recoverable by a system of linear equations from the relation $\mathbf{B} \cdot \mathbf{B}^{-1}(\mathbf{P}_1) = \mathbf{P}_1$. This follows since each entry of $\mathbf{B}^{-1}(\mathbf{P}_1)$ is Gaussian-distributed so that $\mathbf{B}^{-1}(\mathbf{P}_1)$ has full rank over \mathbb{Z}_q if the variance of its entries is large enough. After recovering \mathbf{B} , the distinguisher performs the same steps as Eq. (6).

Remark 2 (When `aux` contains `s`). Counterexample 2 relies on that `aux` does not contain information about the LWE secret `s`, so that $(\underbrace{\mathbf{s}^\top \mathbf{P}_1}_{\text{LWE}}, \mathbf{P}_1) \approx (\$, \mathbf{P}_1)$ holds by LWE. Against settings where `aux` is required to contain `s` (e.g. that in [VWW22]), one can modify the above slightly to yield another counterexample: Write $\mathbf{s}^\top = (\mathbf{s}_1^\top, \mathbf{s}_2^\top)$, $\mathbf{B} = \begin{pmatrix} \mathbf{B}_1 \\ \mathbf{B}_2 \end{pmatrix}$, $\mathbf{P}_1 = \begin{pmatrix} \mathbf{Q}_{11} & \mathbf{Q}_{12} \\ \mathbf{Q}_{21} & \mathbf{Q}_{22} \end{pmatrix}$. Let `aux` = $(\mathbf{s}, \mathbf{Q}_{11}, \mathbf{Q}_{22})$. Now, $(\underbrace{\mathbf{s}^\top \mathbf{P}_1}_{\text{LWE}}, \text{aux}) \approx (\$, \text{aux})$ still holds since $\mathbf{s}_2^\top \mathbf{Q}_{21}$ and $\mathbf{s}_1^\top \mathbf{Q}_{12}$ are uniform without information about $\mathbf{Q}_{21}, \mathbf{Q}_{12}$. To distinguish then, use the relations $\mathbf{B}_1 \cdot \mathbf{B}^{-1} \begin{pmatrix} \mathbf{Q}_{11} \\ \mathbf{Q}_{21} \end{pmatrix} = \mathbf{Q}_{11}$ and $\mathbf{B}_2 \cdot \mathbf{B}^{-1} \begin{pmatrix} \mathbf{Q}_{12} \\ \mathbf{Q}_{22} \end{pmatrix} = \mathbf{Q}_{22}$ to recover \mathbf{B} , the rest is the same.

Counterexample 3: Private-Coin Where `Samp` Inputs \mathbf{B} . Our last counterexample applies whenever a private-coin `Samp` inputs the matrix \mathbf{B} , spanning

⁵ The condition of \mathbb{Z}_q being a field can be naturally extended to the ring setting, where \mathcal{R}_q splits into subfields of super-polynomial size, so that a random element is invertible with high probability. For simplicity we only consider \mathbb{Z}_q in the rest.

the whole area in red in Fig. 1b, marked with a red cross.⁶ The idea is again simple: If **Samp** knows the matrices $\mathbf{B}, \mathbf{P} = (\mathbf{p}_1, \mathbf{p}_2)$ and wishes to make a secret available to the distinguisher of **then** who additionally receives a short preimage $\mathbf{B}^{-1}(\mathbf{p}_1)$, then an immediate strategy is to encrypt, specifically with the dual-Regev encryption. As is usual in the lattice-setting, for one to distinguish LWE, an appropriate short preimage suffices. Therefore, let **Samp** on input \mathbf{B} output

$$\mathbf{P} = (\mathbf{p}_1, \mathbf{p}_2), \quad \text{aux} = \text{ctxt}_{\mathbf{u}_2} = (\underbrace{\mathbf{R}\mathbf{B}}_{\mathbf{R}\mathbf{B}}, \underbrace{\mathbf{R}\mathbf{p}_1}_{\mathbf{R}\mathbf{p}_1} + \lfloor q/2 \rfloor \mathbf{u}_2),$$

where $\mathbf{P} = (\mathbf{p}_1, \mathbf{p}_2) = \mathbf{B} \cdot (\mathbf{u}_1, \mathbf{u}_2)$ are the images under random short (e.g. Gaussian or binary) preimages $(\mathbf{u}_1, \mathbf{u}_2)$ sampled by **Samp** itself, and $\text{ctxt}_{\mathbf{u}_2}$ the dual-Regev encryption of \mathbf{u}_2 under the public key $(\mathbf{B}, \mathbf{p}_1)$. To see that the if statement holds, observe

$$(\mathbf{B}, \mathbf{P}, \underbrace{\mathbf{s}^T\mathbf{B}}_{\mathbf{s}^T\mathbf{B}}, \underbrace{\mathbf{s}^T\mathbf{P}}_{\mathbf{s}^T\mathbf{P}}, \text{ctxt}_{\mathbf{u}_2}) \stackrel{c}{\approx} (\mathbf{B}, \mathbf{P}, \underbrace{\mathbf{s}^T\mathbf{B}}_{\mathbf{s}^T\mathbf{B}}, \underbrace{\mathbf{s}^T\mathbf{P}}_{\mathbf{s}^T\mathbf{P}}, \$) \stackrel{c}{\approx} (\mathbf{B}, \mathbf{P}, \$, \$, \$)$$

where the first \approx_c follows from security of dual-Regev encryption, the second by LWE. Finally, to distinguish the **then** distributions, we can use $\mathbf{B}^{-1}(\mathbf{p}_1)$ to decrypt by observing that $\mathbf{R}\mathbf{B} \cdot \mathbf{B}^{-1}(\mathbf{p}_1) \approx \mathbf{R}\mathbf{p}_1$, so we can obtain \mathbf{u}_2 . Next, we observe that $\mathbf{B}^{-1}(\mathbf{p}_2) - \mathbf{u}_2$ is a short preimage of $\mathbf{0}$ for the image \mathbf{p}_2 , i.e. $\mathbf{B} \cdot (\mathbf{B}^{-1}(\mathbf{p}_2) - \mathbf{u}_2) \approx \mathbf{0}$. We highlight that in the above we can prove the hardest if condition where \mathbf{B}, \mathbf{P} are in the joint distributions, and we can distinguish the weakest **then** distributions without \mathbf{B} and \mathbf{P} , since the distinguisher does not require them. As such, the above counterexample covers the whole red area in Fig. 1b spanning all settings with/without \mathbf{B}, \mathbf{P} .

On LWE Secret s. We are not aware of counterexamples which specifically require **Samp** to know/generate the LWE secret \mathbf{s} . More concretely, for any attack which targets a subclass where **Samp** needs to know/generate \mathbf{s} , we realise a more general attack which works over the corresponding superclass where **Samp** does not know \mathbf{s} . To illustrate, when **Samp** is allowed to generate the LWE secret \mathbf{s} , we can simplify counterexample 1 against Eq. (4) above, by letting **Samp** embed \mathbf{s} in e.g. the first row \mathbf{P} , thus skipping generating \mathbf{P}_1 and \mathbf{u} . However, once such “embed-secret-in- \mathbf{P} ” mechanism is possible, the attack generalises to one where **Samp** does not need to know \mathbf{s} . Indeed, all attacks that we are aware of work by embedding some secret chosen by **Samp**, which can be recovered given $\mathbf{B}^{-1}(\mathbf{P})$. As long as this can be achieved, one can naturally pick a secret that breaks LWE by interacting with (parts of) \mathbf{P} , without interacting with \mathbf{s} .

Remark 3 (Obfuscation-based Counterexample). [VWW22] suggested a heuristic obfuscation-based counterexample which applies to all private-coin variants discussed above. To recall, let **aux** contain an obfuscated circuit Γ which has \mathbf{P} and its trapdoor hardwired and, on input (\mathbf{C}, \mathbf{D}) , outputs 1 if (1) \mathbf{D} is short, and (2) there exists \mathbf{S} such that $\mathbf{C}\mathbf{D} \approx \mathbf{S}\mathbf{P}$. This allows distinguishing **then** since Γ

⁶ Note that the counterexample also applies to Eq. (5) which considers an even larger sampler class that also knows the trapdoor $\text{td}_{\mathbf{B}}$.

outputs 1 if $\mathbf{C} = \underline{\underline{\mathbf{S}\mathbf{B}}}$ and $\mathbf{D} = \mathbf{B}^{-1}(\mathbf{P})$, but 0 w.h.p. if \mathbf{C} is random. Unfortunately, both proving and disproving this example seem tricky. On the one hand, a proof requires to argue that no if distinguisher can find an input such that Γ evaluates to 1, but the characterisation of such set of inputs is unclear (regardless of strong assumptions like VBB). Meanwhile, to disprove this, one may try finding alternative inputs such that Γ outputs 1, e.g. find short \mathbf{R}, \mathbf{T} such that $\underline{\underline{\mathbf{SPRT}}} = \underline{\underline{\mathbf{SP}}}$, and let $\mathbf{C} = \underline{\underline{\mathbf{SPR}}}$, $\mathbf{D} = \mathbf{T}$.⁷ This however seems hard since $\mathbf{RT} - \mathbf{I} \neq \mathbf{0}$ are SIS solutions w.r.t. (the supposedly random-looking) $\underline{\underline{\mathbf{SP}}}$.⁸

2.3 Plausible Classes of Assumptions

Recall again the intuition of evasive LWE from [Wee22]: To distinguish $\underline{\underline{\mathbf{s}^T\mathbf{B}}}$ given $\mathbf{B}^{-1}(\mathbf{P})$, the seemingly only meaningful way to use $\mathbf{B}^{-1}(\mathbf{P})$ is to right-multiply it to $\underline{\underline{\mathbf{s}^T\mathbf{B}}}$, obtaining new samples $\underline{\underline{\mathbf{s}^T\mathbf{P}}}$ and distinguishing with the latter.

Underlying all counterexamples above are simple alternative uses of $\mathbf{B}^{-1}(\mathbf{P})$ crafted according to the corresponding assumption setting, which we summarise:

Mapping Between \mathbf{B}, \mathbf{P} . Both counterexamples 1 and 2 target the setting where in the if distribution only one of the two matrices \mathbf{B} and \mathbf{P} are fully known to the distinguisher, but in the then distribution the additional information of $\mathbf{B}^{-1}(\mathbf{P})$ allows to recover the other hidden matrix via the relation

$$\mathbf{B} \cdot \mathbf{B}^{-1}(\mathbf{P}) = \mathbf{P}, \tag{7}$$

the latter containing sufficient information for distinguishing LWE.

Encrypt w.r.t. Short Vector $\mathbf{B}^{-1}(\mathbf{P})$. For counterexample 3, $\mathbf{B}^{-1}(\mathbf{P})$ acts as the secret key of an encryption scheme. In fact, in this scenario, during decryption $\mathbf{B}^{-1}(\mathbf{P})$ is still being multiplied to some LWE samples $\underline{\underline{\mathbf{r}^T\mathbf{B}}}$ of \mathbf{B} to obtain new samples $\underline{\underline{\mathbf{r}^T\mathbf{P}}}$ w.r.t. \mathbf{P} . The crucial difference, however, is that such pair of LWE samples is prepared by **Samp** but not the challenger, allowing its embedding of secret in the form of $\underline{\underline{\mathbf{r}^T\mathbf{P}}} + \text{secret}$.

With these in mind, we propose three main families of evasive LWE assumptions where these alternative uses cannot apply. These families are marked with green checkboxes in Fig. 1. See Sect. 4 for formal definitions.

Public-coin Evasive LWE. The first family is when **Samp** is public-coin, meaning that it outputs all randomness used. This captures for example the assumptions in [Wee22, WWW22, HLL24, Wee24, CLW24]. We require that \mathbf{B} is contained in the joint distribution, see Remark 4 for a discussion. We also require that **Samp** does not input \mathbf{B} , which is supported by a heuristic counterexample sketched in the full version of this work.⁹ Note that under

⁷ \mathbf{R} serves to make \mathbf{C} of the same width as $\underline{\underline{\mathbf{S}\mathbf{B}}}$. If \mathbf{B}, \mathbf{P} were of the same width, letting

$\mathbf{R} = \mathbf{T} = \mathbf{I}$ completes the disproof; The parameters given by [VWW22] disallow this.

⁸ It holds that $\underline{\underline{\mathbf{SP}}}(\mathbf{RT} - \mathbf{I}) = \mathbf{0} \pmod q$ and that $\mathbf{RT} - \mathbf{I}$ is short.

⁹ We thank a reviewer of AsiaCrypt 2024 for pointing out this counterexample to us!

this family, \mathbf{P} is always available to the distinguisher (c.f. Fig. 1a), since it is efficiently recoverable from the randomness used by **Samp**.

Private-coin Binding Evasive LWE. The second family is when **Samp** is private-coin, and \mathbf{B}, \mathbf{P} are explicitly included in the joint distributions. Assumptions of this family take the following form: For any PPT **Samp** inputting the security parameter λ and outputting $(\mathbf{S}, \mathbf{P}, \text{aux})$, where **aux** need not include all randomness used,

$$\begin{aligned} \text{if } (\mathbf{B}, \mathbf{P}, \underline{\mathbf{S}}\mathbf{B}, \underline{\mathbf{S}}\mathbf{P}, \text{aux}) &\approx_c (\mathbf{B}, \mathbf{P}, \$, \$, \text{aux}) \\ \text{then } (\mathbf{B}, \mathbf{P}, \underline{\mathbf{S}}\mathbf{B}, \mathbf{B}^{-1}(\mathbf{P}), \text{aux}) &\approx_c (\mathbf{B}, \mathbf{P}, \$, \mathbf{B}^{-1}(\mathbf{P}), \text{aux}). \end{aligned}$$

This corresponds to the bottom-left green checkbox area of Fig. 1b and captures the variant Eq. (8) used by [ARYY23, AKY24], to be discussed in Sect. 2.4.

Private-coin Hiding Evasive LWE. The third and most subtle family is when **Samp** is private-coin and \mathbf{B}, \mathbf{P} are hidden from the joint distribution. Our proposed family takes the following form: For any PPT **Samp** inputting the security parameter λ and outputting $(\mathbf{S}, \mathbf{P}, \text{aux})$, where **aux** need not include all randomness used, and it holds $(\mathbf{P}, \text{aux}) \approx_c (\mathbf{P} + \mathbf{R}, \text{aux})$ for a bounded-norm random \mathbf{R} ,

$$\begin{aligned} \text{if } (\underline{\mathbf{S}}\mathbf{B}, \underline{\mathbf{S}}\mathbf{P}, \text{aux}) &\approx_c (\$, \$, \text{aux}) \\ \text{then } (\underline{\mathbf{S}}\mathbf{B}, \mathbf{B}^{-1}(\mathbf{P}), \text{aux}) &\approx_c (\$, \mathbf{B}^{-1}(\mathbf{P}), \text{aux}). \end{aligned}$$

The bounded-norm \mathbf{R} can be interpreted as noise, and $\mathbf{P} + \mathbf{R}$ some approximation of \mathbf{P} . The indistinguishability between \mathbf{P} and $\mathbf{P} + \mathbf{R}$ serves as a proof of “ \mathbf{P} cannot be approximated given **aux**” and resists algebraic attacks which would, e.g., recover a column of \mathbf{P} , or a sum of a two columns, since either allows to distinguish by cross-checking with **aux**. See Sect. 4.3 for a more thorough discussion. This family corresponds to the top-right green checkbox area of Fig. 1b, and seeks to capture existing assumptions taking the form of Eq. (3) while plausibly resisting attacks exploiting the alternative use of $\mathbf{B}^{-1}(\mathbf{P})$ via Eq. (7): Without knowing neither \mathbf{B} nor approximation of \mathbf{P} , it is unclear how Eq. (7) may still be exploited.

Remark 4. Observant readers might have noticed that we have not included the top-left areas of Figs. 1a and 1b, the families where \mathbf{B} is hidden but \mathbf{P} is known. While we have not found counterexamples on these families, we believe they are of relatively low utility, based on the following informal argument: Since \mathbf{P} is fully known, the indistinguishability $\underline{\mathbf{S}}\mathbf{P} \approx_c \$$ implies \mathbf{S} having high entropy given the “if” distribution, in which case an appropriate entropic LWE assumption [BD20] implies $\underline{\mathbf{S}}\mathbf{B} \approx_c \$$ even when \mathbf{B} is known.¹⁰ We therefore expect that, instead of resorting to an assumption in this family, one can utilise the families where \mathbf{B} is known, yielding also a stronger **then** relation where \mathbf{B} is also known.

¹⁰ The argument is informal, because $\underline{\mathbf{S}}\mathbf{P} \approx_c \$$ might not necessarily imply sufficiently high entropy for entropic LWE to apply.

2.4 Assumption Instances in Existing Works

[ARYY23, AKY24]: Counterexample 1 applies to the evasive LWE variant in Eq. (4), which is the one of the two involved in [ARYY23, AKY24]. Fortunately, both works’ proofs are modular, in particular, ARYY show that Eq. (4) implies a second evasive LWE assumption, the latter used by both works to prove security of their constructions. The second assumption is as follows: For any PPT Samp outputting an (arbitrary) LWE secret \mathbf{S} , matrix \mathbf{P} , and auxiliary information $\text{aux}_1, \text{aux}_2$ (not necessarily containing all randomness used by Samp), where \mathbf{P} is efficiently computable given aux_2 ,

$$\text{if } (\mathbf{B}, \underline{\mathbf{SB}}, \underline{\mathbf{SP}}, \text{aux}_1, \text{aux}_2) \approx_c (\mathbf{B}, \$, \$, \$, \text{aux}_2) \quad (8)$$

$$\text{then } (\mathbf{B}, \underline{\mathbf{SB}}, \mathbf{B}^{-1}(\mathbf{P}), \text{aux}_1, \text{aux}_2) \approx_c (\mathbf{B}, \$, \mathbf{B}^{-1}(\mathbf{P}), \$, \text{aux}_2)$$

for uniformly random \mathbf{B} . Since \mathbf{P} is efficiently computable from aux_2 , one can also phrase Eq. (8) as an instance of our proposed private-coin binding evasive LWE, where \mathbf{B} and \mathbf{P} are both fully available in the joint distribution. Thus, assuming the private-coin binding evasive LWE, the MI-ABE of [ARYY23] and CP-ABE of [AKY24] remain secure.

[VWW22]: Counterexample 2 applies to the evasive LWE variant in Eq. (3), which first appears in [VWW22] and is recently involved in [CM24, MPV24]. Nevertheless, we have not found attacks on the assumption instance (i.e. the specific Samp) used by [VWW22]. Indeed, in Sect. 6 we show that, the assumption instance used by [VWW22] falls under our proposed private-coin hiding evasive LWE, where \mathbf{B} is not given in the joint distribution and \mathbf{P} is hidden. The main reason is that each entry of \mathbf{P} contains Gaussian noise that is *independent* of aux , and when the Gaussian parameter is sufficiently large, the noise term, consequently also \mathbf{P} , is statistically irrecoverable. Thus, assuming the private-coin hiding evasive LWE, the constructions of [VWW22] remain secure.

[Tsa22]: Counterexample 3 applies to the evasive-type assumption by [Tsa22], where a private-coin Samp inputs \mathbf{B} and its outputs can be arbitrarily correlated with \mathbf{B} . Nevertheless, we observe that the condition of Samp knowing \mathbf{B} seems to be an artifact of the definitional style used in [Tsa22]. In Tsabary’s evasive-type assumption, Samp also outputs the pre-image $\mathbf{B}^{-1}(\mathbf{P})$ for the *then* distribution, which necessitates Samp to input \mathbf{B} and its trapdoor. In contrast, in other evasive LWE assumptions, such $\mathbf{B}^{-1}(\mathbf{P})$ is generated by the challenger but not Samp . Upon closer inspection, the Samp instance in the security proof of [Tsa22] indeed *only* uses \mathbf{B} and $\text{td}_{\mathbf{B}}$ for sampling the preimages $\mathbf{B}^{-1}(\mathbf{P})$ but nothing else. In particular aux contains no further components correlated to \mathbf{B} and $\text{td}_{\mathbf{B}}$. Therefore, it seems plausible to us that Tsabary’s construction can be proved under a variant of private-coin evasive LWE that is not subject to counterexamples. However, verifying this claim is not straightforward, since Tsabary’s assumption and proof comes with many subtle differences. In more detail, Tsabary considers exponential-size distributions over matrices and LWE samples, and the distinguisher accesses them via querying an oracle with an index for the sample. In turn, evasive LWE provides polynomial-size samples

directly to the distinguisher, so the adaptation of Tsabary’s proof to evasive LWE requires syntactic changes and possibly re-structuring the proof into more game hops. We leave verifying the security of Tsabary’s construction to future works.

3 Preliminaries

Denote by $\mathbb{N} = \{1, 2, \dots\}$ the set of natural numbers, by \mathbb{Z} the set of integers and by \mathbb{R} the set of real numbers. For $n \in \mathbb{N}$, we set $[n] = \{1, \dots, n\}$. For a ring extension $\mathcal{R} \supseteq \mathbb{Z}$, we set $\mathcal{R}_q := \mathcal{R}/(q \cdot \mathcal{R})$.

Denote by λ the security parameter. Write $\text{poly}(\lambda) = \bigcup_{d \in \mathbb{N}} O(\lambda^d)$, $\text{negl}(\lambda) = \bigcap_{d \in \mathbb{N}} o(\lambda^{-d})$. A **PPT** algorithm is a probabilistic algorithm whose time complexity lies in $\text{poly}(\lambda)$. For a PPT algorithm \mathcal{A} , we write $\mathcal{A}(\cdot; \text{rand})$ for running which on randomness rand , where rand is understood to be uniformly random over its randomness space. Denote by $\mathcal{U}(S)$ the uniform distribution over a finite set S . The **statistical distance** between two discrete distributions $\mathcal{S}_1, \mathcal{S}_2$ over a set X is $\Delta(\mathcal{S}_1, \mathcal{S}_2) = \frac{1}{2} \sum_{x \in X} |\mathcal{S}_1(x) - \mathcal{S}_2(x)|$. For a vector $\mathbf{x} = (x_1, \dots, x_m)^T \in \mathbb{R}^m$, its ℓ_2 -norm is $\|\mathbf{x}\| := \|\mathbf{x}\|_2 = \sqrt{\sum_{i \in [m]} x_i^2}$.

3.1 Lattices and Gaussian Distributions

Definition 1 (Lattice and Dual Lattice). A *lattice* is a discrete additive subgroup $\Lambda \subset \mathbb{R}^m$. The *rank* of a lattice Λ is defined to be the vector space dimension by the space spanned by the elements of Λ .

The *dual lattice* of a lattice Λ is given by

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

Definition 2 (q -ary Lattices and Cosets). For a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, define its orthogonal lattice by

$$\Lambda^\perp(\mathbf{A}) := \{\mathbf{e} \in \mathbb{Z}^m \mid \mathbf{A}\mathbf{e} = \mathbf{0} \bmod q\} \subseteq \mathbb{Z}^m$$

and the lattice spanned by its rows by

$$\Lambda(\mathbf{A}) := \{\mathbf{y} \in \mathbb{Z}^m \mid \exists \mathbf{x} \in \mathbb{Z}^n, \mathbf{y} = \mathbf{x}^T \mathbf{A} \bmod q\} \subseteq \mathbb{Z}^m.$$

Additionally, for a syndrome $\mathbf{u} \in \mathbb{Z}_q^n$, define the following coset of $\Lambda^\perp(\mathbf{A})$

$$\Lambda_{\mathbf{u}}^\perp(\mathbf{A}) := \{\mathbf{e} \in \mathbb{Z}^m \mid \mathbf{A}\mathbf{e} = \mathbf{u} \bmod q\} \subseteq \mathbb{Z}^m.$$

For $\mathbf{U} = (\mathbf{u}_1, \dots, \mathbf{u}_k) \in \mathbb{Z}_q^{n \times k}$, the notation extends naturally to $\Lambda_{\mathbf{U}}^\perp(\mathbf{A}) = \Lambda_{\mathbf{u}_1}^\perp(\mathbf{A}) \times \dots \times \Lambda_{\mathbf{u}_k}^\perp(\mathbf{A}) \subseteq \mathbb{Z}^{m \times k}$.

Note that we have $q \cdot \Lambda(\mathbf{A}) = (\Lambda^\perp(\mathbf{A}))^*$ and $\Lambda_{\mathbf{u}}^\perp(\mathbf{A}) = \Lambda^\perp(\mathbf{A}) + \mathbf{t}$, if there exists a $\mathbf{t} \in \mathbb{Z}^m$ s.t. $\mathbf{A}\mathbf{t} = \mathbf{u} \pmod q$.

Definition 3 (Gaussian measure). For $\mathbf{x} \in \mathbb{R}^m$, the *Gaussian measure* with parameter $\Sigma \in \mathbb{R}^{m \times m}$ and center $\mathbf{c} \in \mathbb{R}^m$, where Σ is positive semi-definite, is $\rho_{\Sigma, \mathbf{c}}(\mathbf{x}) = \exp(-\pi \cdot (\mathbf{x} - \mathbf{c})^\top \Sigma^{-1} (\mathbf{x} - \mathbf{c}))$. If $\Sigma = \chi^2 \mathbf{I}$, we write $\rho_{\chi, \mathbf{c}}(\mathbf{x})$. If $\mathbf{c} = \mathbf{0}$, we simply write $\rho_\Sigma(\mathbf{x})$ (resp. $\rho_\chi(\mathbf{x})$). For a discrete set $\Lambda \subset \mathbb{R}^m$, we set $\rho_{\Sigma, \mathbf{c}}(\Lambda) := \sum_{\mathbf{x} \in \Lambda} \rho_{\Sigma, \mathbf{c}}(\mathbf{x})$.

Definition 4 (Discrete Gaussian Distribution). For a non-empty discrete set $\Lambda \subset \mathbb{R}^m$, define the *discrete Gaussian distribution* over Λ with parameter $\Sigma \in \mathbb{R}^{m \times m}$ and center $\mathbf{c} \in \mathbb{R}^m$, where Σ is positive definite, as

$$D_{\Lambda, \Sigma, \mathbf{c}}(\mathbf{x}) = \frac{\rho_{\Sigma, \mathbf{c}}(\mathbf{x})}{\rho_{\Sigma, \mathbf{c}}(\Lambda)} \quad \text{if } \mathbf{x} \in \Lambda,$$

and $D_{\Lambda, \chi, \mathbf{c}}(\mathbf{x}) = 0$ otherwise. If $\Sigma = \chi^2 \mathbf{I}$, we write $D_{\Lambda, \chi, \mathbf{c}}$ for $D_{\Lambda, \Sigma, \mathbf{c}}$. If $\mathbf{c} = \mathbf{0}$, we simply write $D_{\Lambda, \Sigma}$ (resp. $D_{\Lambda, \chi}$).

For $\Lambda_{\mathbf{U}}^\perp(\mathbf{A}) = \Lambda_{\mathbf{u}_1}^\perp(\mathbf{A}) \times \dots \times \Lambda_{\mathbf{u}_k}^\perp(\mathbf{A})$ where each $\Lambda_{\mathbf{u}_i}^\perp(\mathbf{A})$ is non-empty, we write $D_{\Lambda_{\mathbf{U}}^\perp(\mathbf{A}), \Sigma, \mathbf{c}}$ for the (horizontal) concatenation of the discrete Gaussian distributions over each $\Lambda_{\mathbf{u}_i}^\perp(\mathbf{A})$.

Definition 5 (Smoothing Parameter [MR04]). For a full-rank lattice $\Lambda \subset \mathbb{R}^m$ and $\epsilon > 0$, its *smoothing parameter* is

$$\eta_\epsilon(\Lambda) := \inf\{\chi > 0 \mid \rho_{1/\chi}(\Lambda^* \setminus \{\mathbf{0}\}) \leq \epsilon\}.$$

The following lemma allows to bound the smoothing parameter for orthogonal lattices of random matrices:

Lemma 1 ([Pei07, GPV08]). Let q be prime, $m \geq 2n \cdot \log q$. We have

$$\Pr_{\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}} \left[\eta_{2^{-n}}(\Lambda^\perp(\mathbf{A})) \leq \frac{4}{\sqrt{\pi}} \cdot \sqrt{\log(2m) + \log(1 + 2^n)} \right] \geq 1 - q^{-n}.$$

Lemma 1 is an implication of results of [Pei07, GPV08]. For details, we refer the reader to the full version of this work.

Lemma 2 ([MR04, PR06]). Let $\Lambda \subset \mathbb{R}^m$ be an m -dimensional full-rank lattice. For any $\mathbf{c} \in \mathbb{R}^m$, $\epsilon > 0$, $\chi \geq 2\eta_\epsilon(\Lambda)$, and $\mathbf{y} \in \Lambda$, it holds that

$$D_{\Lambda, \chi, \mathbf{c}}(\mathbf{y}) \leq 2^{-m} \cdot \frac{1 + \epsilon}{1 - \epsilon}.$$

Lemma 3 (Leftover Hash Lemma [HILL99, BDK+11]). Let $m, n, q \in \mathbb{N}$. If we draw $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}$, $\mathbf{y} \leftarrow \mathbb{Z}_q^n$ and $\mathbf{x} \leftarrow \{0, 1\}^m$, we have

$$\Delta((\mathbf{A}, \mathbf{A}\mathbf{x} \pmod q), (\mathbf{A}, \mathbf{y})) \leq \frac{1}{2} \cdot \frac{q^{n/2}}{2^{m/2}}.$$

Lemma 4 (Noise Flooding). *Let $\chi = D_{\mathbb{Z}, \sigma}$. For all $t \in \mathbb{Z}$, we have $\Delta(\chi, \chi + t) \leq \sqrt{\frac{\pi}{2}} \cdot \frac{\|t\|}{\sigma}$. In particular, if $\chi \in \lambda^{\omega(1)} \|t\|$, then $\Delta(\chi, \chi + t) \in \text{negl}(\lambda)$.*

For a proof of Lemma 4, see [BDE+18, Appendix A.2].

3.2 Lattice Assumptions and Lattice Trapdoors

Definition 6 (Learning with Errors). *Let q, n, m, χ be parametrised by λ , where $n, m, q \in \mathbb{N}$ with $n, m \in \text{poly}(\lambda)$. The (decisional) **Learning with Errors** $\text{LWE}_{q, n, m, \chi}$ assumption states that for every PPT distinguisher \mathcal{D} , it holds that*

$$\left| \Pr \left[b = 0 \mid \begin{array}{l} \mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m} \\ \mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow D_{\mathbb{Z}^m, \chi} \\ \mathbf{b}^\top := \mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top \pmod{q} \\ b \leftarrow \mathcal{D}(\mathbf{A}, \mathbf{b}) \end{array} \right] - \Pr \left[b = 0 \mid \begin{array}{l} \mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m} \\ \mathbf{b} \leftarrow \mathbb{Z}_q^m \\ b \leftarrow \mathcal{D}(\mathbf{A}, \mathbf{b}) \end{array} \right] \right| \in \text{negl}(\lambda).$$

In their seminal work, [MP12] gave algorithms for sampling (almost) uniformly random matrices together with trapdoors that allow for LWE inversion and preimage lattice sampling. Currently, their algorithms are the status quo for these tasks, and we will usually assume that the challengers for the evasive LWE assumptions will resort to them. Hence, we will give here an overview of the guarantees given in [MP12] for these algorithms:

Theorem 1 ([MP12]). *Let $m \geq 3n \cdot \log(q)$ and set $w = n \cdot \lceil \log q \rceil$. Let $\chi \in \Omega(n \cdot \sqrt{\log q})$. There exist algorithms (TrapGen, SampPre) with the following properties:*

1. *TrapGen($1^n, 1^m$) outputs two matrices $\mathbf{B} \in \mathbb{Z}_q^{n \times m}$, $\text{td} \in \{-1, 0, 1\}^{w \times (m-w)}$ s.t. the statistical distance between \mathbf{B} and $\mathcal{U}(\mathbb{Z}_q^{n \times m})$ is upper-bounded by 2^{-n} .*
2. *Draw $(\mathbf{B}, \text{td}) \leftarrow \text{TrapGen}(1^n, 1^m)$, let $\mathbf{u} \in \mathbb{Z}_q^n$. For $\mathbf{e}' \leftarrow \text{SampPre}(\mathbf{B}, \text{td}, \mathbf{u}, \chi)$ and $\mathbf{e} \leftarrow D_{\Lambda_{\mathbf{u}}^\perp(\mathbf{B}, \chi)}$, we have $\Delta((\mathbf{B}, \mathbf{e}), (\mathbf{B}, \mathbf{e}')) \in O(2^{-n})$.*

4 Evasive LWE: Definitions, Classes

We formally define the three classes of evasive LWEs outlined in Sect. 2.3.

4.1 Public-Coin Evasive LWE

Definition 7 (Public-coin Evasive LWE). *Let the parameters*

$$\text{param} = (\mathcal{R}, q, n, n_A, m, m_P, m_A, t, t_A, \mathcal{D}, \mathcal{S}, \chi_B, \chi_P, \chi_A, f, f_A, \Sigma)$$

be parametrised by λ , where \mathcal{R} is a ring admitting an embedding as a lattice in \mathbb{R}^φ for some $\varphi \in \mathbb{N}$, $\mathcal{D} \sim \mathcal{R}_q^{n \times m}$, $\mathcal{S} \sim \mathcal{R}_q^{t \times n} \times \mathcal{R}_q^{t_A \times n_A}$, $\chi_B \sim \mathcal{R}^{t \times m}$, $\chi_P \sim \mathcal{R}^{t \times m_P}$, and $\chi_A \sim \mathcal{R}^{t_A \times m_A}$ are distributions, $\Sigma \in \mathbb{R}^{\varphi m \times \varphi m}$ is positive definite, and f, f_A are PPT algorithms. Let Samp be a PPT algorithm which, on input 1^λ , outputs

$$(\mathbf{A} \in \mathcal{R}_q^{n_A \times m_A}, \mathbf{P} \in \mathcal{R}_q^{n \times m_P}, \text{aux} \in \{0, 1\}^*).$$

$\text{Pre}_A^b(1^\lambda)$	$\text{Post}_B^b(1^\lambda)$
$\mathbf{B} \leftarrow \mathcal{D}$	$\mathbf{B} \leftarrow \mathcal{D}$
$(\mathbf{A}, \mathbf{P}, \text{aux}) \leftarrow \text{Samp}(1^\lambda; \text{rand})$	$(\mathbf{A}, \mathbf{P}, \text{aux}) \leftarrow \text{Samp}(1^\lambda; \text{rand})$
$\text{assert } \mathbf{P} \in \mathbf{BR}^{m \times m_P}$	$\text{assert } \mathbf{P} \in \mathbf{BR}^{m \times m_P}$
$(\mathbf{S}, \mathbf{S}_A) \leftarrow \mathcal{S}$	$(\mathbf{S}, \mathbf{S}_A) \leftarrow \mathcal{S}$
$\text{hint} := (f(\mathbf{S}; \text{rand}_f), f_A(\mathbf{S}_A; \text{rand}_{f_A}))$	$\text{hint} := (f(\mathbf{S}; \text{rand}_f), f_A(\mathbf{S}_A; \text{rand}_{f_A}))$
if $b = 0$ then	if $b = 0$ then
$\mathbf{E}_A \leftarrow \chi_A, \mathbf{E}_B \leftarrow \chi_B, \mathbf{E}_P \leftarrow \chi_P$	$\mathbf{E}_A \leftarrow \chi_A, \mathbf{E}_B \leftarrow \chi_B$
$\mathbf{C}_A := \mathbf{S}_A \mathbf{A} + \mathbf{E}_A \text{ mod } q$	$\mathbf{C}_A := \mathbf{S}_A \mathbf{A} + \mathbf{E}_A \text{ mod } q$
$\mathbf{C}_B := \mathbf{S} \mathbf{B} + \mathbf{E}_B \text{ mod } q$	$\mathbf{C}_B := \mathbf{S} \mathbf{B} + \mathbf{E}_B \text{ mod } q$
$\mathbf{C}_P := \mathbf{S} \mathbf{P} + \mathbf{E}_P \text{ mod } q$	$\mathbf{U} \leftarrow D_{A_{\mathbb{F}}^b(\mathbf{B}), \Sigma}$
if $b = 1$ then	if $b = 1$ then
$\mathbf{C}_A \leftarrow \mathcal{R}_q^{t_A \times m_A}$	$\mathbf{C}_A \leftarrow \mathcal{R}_q^{t_A \times m_A}$
$\mathbf{C}_B \leftarrow \mathcal{R}_q^{t_B \times m}$	$\mathbf{C}_B \leftarrow \mathcal{R}_q^{t_B \times m}$
$\mathbf{C}_P \leftarrow \mathcal{R}_q^{t_P \times m_P}$	$\mathbf{U} \leftarrow D_{A_{\mathbb{F}}^b(\mathbf{B}), \Sigma}$
return $\mathcal{A} \left(\begin{array}{c} \mathbf{A}, \mathbf{B}, \mathbf{P}, \text{hint, aux,} \\ \mathbf{C}_A, \mathbf{C}_B, \mathbf{C}_P, \\ \text{rand, rand}_{f_A}, \text{rand}_f \end{array} \right)$	return $\mathcal{B} \left(\begin{array}{c} \mathbf{A}, \mathbf{B}, \mathbf{P}, \text{hint, aux,} \\ \mathbf{C}_A, \mathbf{C}_B, \mathbf{U}, \\ \text{rand, rand}_{f_A}, \text{rand}_f \end{array} \right)$

Fig. 2. Experiments Pre and Post for public-coin evasive LWE.

Denote

$$\begin{aligned} \text{Adv}_A^{\text{Pre}}(\lambda) &:= \left| \Pr [\text{Pre}_A^0(1^\lambda) = 1] - \Pr [\text{Pre}_A^1(1^\lambda) = 1] \right|, \\ \text{Adv}_B^{\text{Post}}(\lambda) &:= \left| \Pr [\text{Post}_B^0(1^\lambda) = 1] - \Pr [\text{Post}_B^1(1^\lambda) = 1] \right|, \end{aligned}$$

where the experiments Pre_A^b and Post_B^b are defined in Fig. 2. The $\text{PublicEvLWE}_{\text{param}}$ assumption states that for any PPT Samp and \mathcal{B} there exists a PPT \mathcal{A} such that $\text{Adv}_A^{\text{Pre}}(\lambda) \geq \text{Adv}_B^{\text{Post}}(\lambda) / \text{poly}(\lambda) - \text{negl}(\lambda)$.

Remark 5. We parametrise the assumption by the modulus, matrix dimensions, noise parameters, etc. This is analogous to how the LWE assumption is defined when done precisely, which formally is parametrised by $(\mathcal{R}, q, n, m, \chi)$. We emphasise that we believe the plausibility of an evasive LWE assumption should depend on these parameters, analogous to that the plausibility of LWE depends on e.g. the dimension n and the ratio q/χ . As we will see, existing public-coin evasive LWE all fall under Definition 7 with specially chosen parameters.

Definition 7 is a versatile definition designed to capture both the public-coin flavour and many different (sometimes implicit) features involved in existing definitions. We elaborate on some key aspects.

Randomness of Samp. The randomness rand used by Samp is made explicit syntactically, to highlight the public-coin nature. The same convention was used in [HLL24]. Given rand , the inputs $(\mathbf{A}, \mathbf{P}, \text{aux})$ to \mathcal{A} and \mathcal{B} may be omitted as they can be derived from rand ; we include them for clarity.

Matrix \mathbf{A} . The matrix \mathbf{A} serves to represent the LWE matrix not involved in the preimage-image relation which the evasive LWE assumption concerns, i.e. in the Post experiment, the acquired preimages do not involve \mathbf{A} . This serves to increase expressiveness of the assumption while leaving the intuition of evasive LWE unchanged, and is required in a number of works [Wee22, WWW22, HLL24, CLW24]. By setting \mathbf{A} the empty matrix, we recover the simpler cases outlined in Sect. 2.

Check $\mathbf{P} = \mathbf{B}\mathcal{R}^{m_P}$. This check makes the evasive LWE assumption formally well-defined: Without this check, in case $\mathbf{P} = (\mathbf{p}_i)_i$ is not in the \mathcal{R} -span of \mathbf{B} , the distribution $D_{\Lambda_{\mathbf{P}}^\perp(\mathbf{B}), \Sigma}$, and consequently Post^b , would be ill-defined, since some $\Lambda_{\mathbf{p}_i}^\perp(\mathbf{B})$ would be empty. In existing works, this check is implicit since in those settings w.h.p. \mathbf{B} is primitive and all $\Lambda_{\mathbf{p}_i}^\perp(\mathbf{B}) \neq \emptyset$.

Distribution of \mathbf{B} . We let \mathbf{B} be sampled from a distribution \mathcal{D} , the latter being itself a parameter of the assumption. In the simple setting of [Wee22], \mathcal{D} is the uniform distribution over $\mathbb{Z}_q^{n \times m}$. As discussed below, by setting \mathcal{D} appropriately, other existing variants, e.g. where an evasive LWE is defined over multiple \mathbf{B}_i 's, can also be naturally captured. To understand and gain confidence in this generalisation, we discuss some special cases of \mathcal{D} . (1) Low-entropy \mathcal{D} (in the extreme case \mathbf{B} is deterministic) and $n < m$: In this case LWE w.r.t. \mathbf{B} is likely easy, so that both Pre^b and Post^b can be efficiently distinguished, and the assumption is vacuously true. (2) \mathcal{D} is such that \mathbf{B} is (likely) not primitive: If \mathbf{P} is in the column span of \mathbf{B} then the rationale of the assumption stays unchanged; otherwise, the check $\mathbf{P} = \mathbf{B}\mathcal{R}^{m_P}$ fails, so that the winning probability in both $\text{Pre}^b, \text{Post}^b$ are zero and the assumption is again true. (3) \mathcal{D} is such that the lattice $\Lambda^\perp(\mathbf{B})$ (likely) contains no short vector: The assumption, in particular the distribution $D_{\Lambda_{\mathbf{P}}^\perp(\mathbf{B}), \Sigma}$, is still well-defined, although with the unusual scenario that \mathbf{U} is (likely) not short, which intuitively only makes distinguishing Post^b harder.

LWE Secret Distributions. The LWE secret distribution \mathcal{S} , determining (correlation between) the LWE secret for samples w.r.t. to (\mathbf{B}, \mathbf{P}) and \mathbf{A} respectively, is parametrised by the assumption (instead of e.g. fixed to be uniform). This is in line with the intuition of evasive LWE (and already implicit in existing private-coin variants), which says that secret distributions should not matter to the if-then relation that the assumption postulates, since intuitively $\mathbf{B}^{-1}(\mathbf{P})$ should not be able to interact with \mathbf{S} and \mathbf{S}_A meaningfully. Note that a poor secret distribution would allow to distinguish Pre^b in the first place so that the assumption is vacuously true. As to be discussed below, this treatment further allows us to interpret some existing variants in an arguably more intuitive way.

Public-coin Hint. The hint on the LWE secrets, which are outputs of the functions f, f_A parametrised by the assumption, manifests the intuition of “secret

distributions should not matter to the if-then relation”, in that leakages on the secret may be given to the distinguishers. We view this as an evasive analogue of entropic LWE [BD20]. (In private-coin variants, such hint is implicit in `aux` since `Samp` can generate it itself.) To be consistent with the spirit of “public-coin”, we require the randomness $\text{rand}_{f_A}, \text{rand}_f$ used in generating hint (if any) to be provided to the distinguisher. We note again that a poor leakage would allow a distinguisher to distinguish Pre^b so that the assumption is vacuously true.

B not known to Samp. Similar to the private-coin setting to be discussed up next, in Definition 7 we forbid the sampler `Samp` to input `B`. Although we are unable to provide a provable counterexample against this case, in the full version of this work we provide a heuristic counterexample to support this restriction.

Relating to Existing Definitions. All existing public-coin evasive LWE can be viewed as special cases of Definition 7, which we briefly summarise. Most works define an assumption over the integers $\mathcal{R} = \mathbb{Z}$.¹¹ The original evasive LWE of [Wee22], later also used in [Wee24], has secret distribution \mathcal{S} such that $\mathbf{S}_A = \mathbf{S} = \mathbf{s}^T$ where \mathbf{s} is uniform, and f_A, f the empty function, i.e. no hint involved. [WWW22] defined a public-coin evasive LWE with multiple independent and uniform $\mathbf{B}_i \in \mathbb{Z}_q^{n \times m}$, with LWE samples $\mathbf{s}_i^T \mathbf{B}_i + \mathbf{e}_i^T \bmod q$ under independent uniform secrets \mathbf{s}_i , and additionally with samples w.r.t. \mathbf{A} under the concatenated secret $(\mathbf{s}_1^T, \dots, \mathbf{s}_k^T)$. This can be expressed by our definition as hav-

ing $\mathbf{B} = \begin{pmatrix} \mathbf{B}_1 & & \\ & \ddots & \\ & & \mathbf{B}_k \end{pmatrix}$ and secret distribution \mathcal{S} such that $\mathbf{S} = \mathbf{S}_A = (\mathbf{s}_1^T, \dots, \mathbf{s}_k^T)$.

[CLW24] used a public-coin evasive LWE that is similar to that of [WWW22], but let \mathcal{S} be such that $\mathbf{S} = (\mathbf{s}_1^T, \mathbf{s}^T, \dots, \mathbf{s}_k^T, \mathbf{s}^T)$, $\mathbf{S}_A = (\mathbf{s}_1^T, \dots, \mathbf{s}_k^T, \mathbf{s}^T)$, where \mathbf{s}_i, \mathbf{s} are all uniform. [HLL24] defined a public-coin evasive LWE with matrices and LWE samples of the (arguably ad-hoc) forms

$$\mathbf{B} = \begin{pmatrix} \mathbf{B}_0 \\ \vdots \\ \mathbf{B}_k \end{pmatrix}, \quad \mathbf{s}_i^T (\mathbf{B}_0, \dots, \mathbf{B}_k) + (\mathbf{0}^T, \mathbf{g}^T \otimes \mathbf{e}_{i,1}^T) + \mathbf{e}_{i,2}^T \bmod q \quad \text{for all } i \in [I]$$

(with preimages $\mathbf{U} = \mathbf{B}^{-1}(\mathbf{P})$ w.r.t. \mathbf{B}), where $\mathbf{g}^T = (2^0, \dots, 2^{k-1})$, and with further samples of the form $(\mathbf{s}_i^T, \mathbf{s}_0^T) \mathbf{A} + \mathbf{e}_{i,0}^T \bmod q$ for all $i \in [I]$; This can be summarised by our definition as letting $\mathbf{B} \leftarrow_{\mathcal{S}} \mathbb{Z}_q^{nk \times m}$ be simply uniformly random, and with structured secret and error distributions $\mathcal{S}, \chi_B, \chi_A$ such that

¹¹ On the other hand, suppose one is to optimise any involved constructions for efficiency, then it is easy to see that an analogous evasive LWE instance over other rings \mathcal{R} , e.g. the ring of integers of some cyclotomic field, is to be involved.

$$\mathbf{S} = \begin{pmatrix} \hat{\mathbf{S}} & & \\ & \ddots & \\ & & \hat{\mathbf{S}} \end{pmatrix} \text{ where } \hat{\mathbf{S}} = \begin{pmatrix} \mathbf{s}_1^\top \\ \vdots \\ \mathbf{s}_I^\top \end{pmatrix}, \quad \mathbf{S}_A = \begin{pmatrix} \mathbf{s}_1^\top, \mathbf{s}_0^\top \\ \vdots \\ \mathbf{s}_I^\top, \mathbf{s}_0^\top \end{pmatrix},$$

$$\mathbf{E}_B = \begin{pmatrix} (\mathbf{0}^\top, \mathbf{g}^\top \otimes \mathbf{e}_{1,1}^\top) + \mathbf{e}_{1,2}^\top \\ \vdots \\ (\mathbf{0}^\top, \mathbf{g}^\top \otimes \mathbf{e}_{I,1}^\top) + \mathbf{e}_{I,2}^\top \end{pmatrix}, \quad \mathbf{E}_A = \begin{pmatrix} \mathbf{e}_{1,0}^\top \\ \vdots \\ \mathbf{e}_{I,0}^\top \end{pmatrix}.$$

Remark 6 (Relation to evasive circular LWE of [HLL23]). In [HLL23] an “evasive circular LWE assumption” is proposed, which can be viewed as involving a non-trivial hint function. However, despite being regarded as a public-coin assumption by [HLL23], this does not fall into the public-coin family under our characterisation (also when factoring out its circular nature), due to its hint function being not public-coin. Further discussion on this in the full version of this work.

4.2 Private-Coin Binding Evasive LWE

Definition 8 (Private-coin Binding Evasive LWE). *Let the parameters*

$$\text{param} = (\mathcal{R}, q, n, m, m_P, t, \mathcal{D}, \chi_B, \chi_P, \Sigma)$$

be parametrised by λ , where \mathcal{R} is a ring admitting an embedding as a lattice in \mathbb{R}^φ for some $\varphi \in \mathbb{N}$, $\mathcal{D} \sim \mathcal{R}_q^{n \times m}$, $\chi_B \sim \mathcal{R}^{t \times m}$, and $\chi_P \sim \mathcal{R}^{t \times m_P}$ are distributions, and $\Sigma \in \mathbb{R}^{\varphi m \times \varphi m}$ is positive definite. Let Samp be a PPT algorithm which, on input 1^λ , outputs

$$(\mathbf{S} \in \mathcal{R}_q^{t \times n}, \mathbf{P} \in \mathcal{R}_q^{n \times m_P}, \text{aux} \in \{0, 1\}^*).$$

Let Pre_A^b and Post_B^b be the experiments defined in Fig. 3 and denote

$$\text{Adv}_A^{\text{Pre}}(\lambda) := |\Pr[\text{Pre}_A^0(1^\lambda) = 1] - \Pr[\text{Pre}_A^1(1^\lambda) = 1]|,$$

$$\text{Adv}_B^{\text{Post}}(\lambda) := |\Pr[\text{Post}_B^0(1^\lambda) = 1] - \Pr[\text{Post}_B^1(1^\lambda) = 1]|.$$

The $\text{PrivateBindEvLWE}_{\text{param}}$ assumption states that for any PPT Samp and \mathcal{B} there exists a PPT \mathcal{A} such that $\text{Adv}_A^{\text{Pre}}(\lambda) \geq \text{Adv}_B^{\text{Post}}(\lambda)/\text{poly}(\lambda) - \text{negl}(\lambda)$.

We explain some key aspects of Definition 8.

Randomness of Samp . As its name suggests, in Definition 8 the randomness of Samp need not be given to the distinguishers. This makes the assumption more susceptible to future attacks, as Samp can embed secret information in aux . For example, as discussed in [VWW22], one may potentially include in aux a carefully crafted obfuscation containing a trapdoor of \mathbf{P} (cf. Remark 3).

B not known to Samp . The assumption is restricted to the class of Samp which does not input the matrix \mathbf{B} . This avoids counterexamples such as ours, sketched in Sect. 2.2 and detailed in Sect. 5.3.

$\text{Pre}_{\mathcal{A}}^b(1^\lambda)$	$\text{Post}_{\mathcal{B}}^b(1^\lambda)$
$\mathbf{B} \leftarrow_{\$} \mathcal{D}$	$\mathbf{B} \leftarrow_{\$} \mathcal{D}$
$(\mathbf{S}, \mathbf{P}, \text{aux}) \leftarrow \text{Samp}(1^\lambda)$	$(\mathbf{S}, \mathbf{P}, \text{aux}) \leftarrow \text{Samp}(1^\lambda)$
assert $\mathbf{P} \in \mathcal{BR}^{m \times m_P}$	assert $\mathbf{P} \in \mathcal{BR}^{m \times m_P}$
if $b = 0$ then	if $b = 0$ then
$\mathbf{E}_B \leftarrow_{\$} \chi_B, \mathbf{E}_P \leftarrow_{\$} \chi_P$	$\mathbf{E}_B \leftarrow_{\$} \chi_B$
$\mathbf{C}_B := \mathbf{S}\mathbf{B} + \mathbf{E}_B \text{ mod } q$	$\mathbf{C}_B := \mathbf{S}\mathbf{B} + \mathbf{E}_B \text{ mod } q$
$\mathbf{C}_P := \mathbf{S}\mathbf{P} + \mathbf{E}_P \text{ mod } q$	$\mathbf{U} \leftarrow_{\$} D_{A_{\mathbf{P}}^\perp(\mathbf{B}), \Sigma}$
if $b = 1$ then	if $b = 1$ then
$\mathbf{C}_B \leftarrow_{\$} \mathcal{R}_q^{t \times m}$	$\mathbf{C}_B \leftarrow_{\$} \mathcal{R}_q^{t \times m}$
$\mathbf{C}_P \leftarrow_{\$} \mathcal{R}_q^{t \times m_P}$	$\mathbf{U} \leftarrow_{\$} D_{A_{\mathbf{P}}^\perp(\mathbf{B}), \Sigma}$
return $\mathcal{A} \left(\begin{array}{l} \mathbf{B}, \mathbf{P}, \text{aux} \\ \mathbf{C}_B, \mathbf{C}_P, \text{aux} \end{array} \right)$	return $\mathcal{B} \left(\begin{array}{l} \mathbf{B}, \mathbf{P}, \text{aux} \\ \mathbf{C}_B, \mathbf{U}, \text{aux} \end{array} \right)$

Fig. 3. Experiments Pre and Post for private-coin binding evasive LWE.

Correlations among \mathbf{S}, \mathbf{P} and aux . Samp outputs also the LWE secret \mathbf{S} , implying that \mathbf{S} can be correlated to \mathbf{P} and aux secretly and arbitrarily. This leads to another potential attack angle of exploiting such correlations.

Other outputs of Samp. Relative to Definition 7, other components such as \mathbf{A}, \mathbf{S}_A and hint functions are omitted, since these can now be generated by Samp and contained in aux .

Relating to Existing Definitions. The private-coin variant of [ARYY23, Lemma 3.4] falls under Definition 8, where they formulated aux as $(\text{aux}_1, \text{aux}_2)$, where aux_1 can be proven pseudorandom and \mathbf{P} is efficiently computable from aux_2 . As reference, the security proofs of [ARYY23] involve calling private-coin evasive LWE iteratively, some instances due to that aux contains preimages w.r.t. components of \mathbf{P} sampled using a trapdoor which cannot be leaked to the distinguisher, and some others due to that aux involves secret correlation with \mathbf{S} ; all instances do not involve correlation between \mathbf{S} and \mathbf{P} . The same variant is later used by [AKY24]. Moreover, the evasive circular LWE assumption of [HLL23] is also a member of Definition 8, for which we discuss in the full version.

4.3 Private-Coin Hiding Evasive LWE

Definition 9 (Private-coin Hiding Evasive LWE). *Let the parameters*

$$\text{param} = (\mathcal{R}, q, n, m, m_P, t, \mathcal{D}, \chi_B, \chi_P, \ell, \Sigma)$$

$\text{Pre1}_{\mathcal{A}}^b(1^\lambda)$ <hr style="border: 0.5px solid black;"/> $\mathbf{B} \leftarrow_{\$} \mathcal{D}$ $(\mathbf{S}, \mathbf{P}, \text{aux}) \leftarrow \text{Samp}(1^\lambda)$ $\text{assert } \mathbf{P} \in \mathcal{BR}^{m \times m_P}$ $\text{if } b = 0 \text{ then}$ $\quad \mathbf{E}_B \leftarrow_{\$} \chi_B, \mathbf{E}_P \leftarrow_{\$} \chi_P$ $\quad \mathbf{C}_B := \mathbf{S}\mathbf{B} + \mathbf{E}_B \text{ mod } q$ $\quad \mathbf{C}_P := \mathbf{S}\mathbf{P} + \mathbf{E}_P \text{ mod } q$ $\text{if } b = 1 \text{ then}$ $\quad \mathbf{C}_B \leftarrow_{\$} \mathcal{R}_q^{t \times m}$ $\quad \mathbf{C}_P \leftarrow_{\$} \mathcal{R}_q^{t \times m_P}$ $\text{return } \mathcal{A}(\mathbf{C}_B, \mathbf{C}_P, \text{aux})$	$\text{Post}_{\mathcal{B}}^b(1^\lambda)$ <hr style="border: 0.5px solid black;"/> $\mathbf{B} \leftarrow_{\$} \mathcal{D}$ $(\mathbf{S}, \mathbf{P}, \text{aux}) \leftarrow \text{Samp}(1^\lambda)$ $\text{assert } \mathbf{P} \in \mathcal{BR}^{m \times m_P}$ $\text{if } b = 0 \text{ then}$ $\quad \mathbf{E}_B \leftarrow_{\$} \chi_B$ $\quad \mathbf{C}_B := \mathbf{S}\mathbf{B} + \mathbf{E}_B \text{ mod } q$ $\quad \mathbf{U} \leftarrow_{\$} D_{\Lambda_{\mathbf{P}}^\perp(\mathbf{B}), \Sigma}$ $\text{if } b = 1 \text{ then}$ $\quad \mathbf{C}_B \leftarrow_{\$} \mathcal{R}_q^{t \times m}$ $\quad \mathbf{U} \leftarrow_{\$} D_{\Lambda_{\mathbf{P}}^\perp(\mathbf{B}), \Sigma}$ $\text{return } \mathcal{B}(\mathbf{C}_B, \mathbf{U}, \text{aux})$
$\text{Pre2}_{\mathcal{A}}^b(1^\lambda)$ <hr style="border: 0.5px solid black;"/> $(\mathbf{S}, \mathbf{P}, \text{aux}) \leftarrow \text{Samp}(1^\lambda)$ $\text{if } b = 0 \text{ then}$ $\quad \text{return } \mathcal{A}(\mathbf{P}, \text{aux})$ $\text{if } b = 1 \text{ then}$ $\quad \mathbf{R} \leftarrow_{\$} \mathcal{U}(\{0, 1, \dots, \ell\}^{n \times m_P})$ $\quad \text{return } \mathcal{A}(\mathbf{P} + \mathbf{R} \text{ mod } q, \text{aux})$	

Fig. 4. Experiments Pre1, Pre2 and Post for private-coin hiding evasive LWE.

be parametrised by λ , where \mathcal{R} is a ring admitting an embedding as a lattice in \mathbb{R}^φ for some $\varphi \in \mathbb{N}$, $\mathcal{D} \sim \mathcal{R}_q^{n \times m}$, $\chi_B \sim \mathcal{R}^{t \times m}$, and $\chi_P \sim \mathcal{R}^{t \times m_P}$ are distributions, $\Sigma \in \mathbb{R}^{\varphi m \times \varphi m}$ is positive definite, and $\ell \in \{1, 2, \dots, q\}$. Let Samp be a PPT algorithm which, on input 1^λ , outputs

$$(\mathbf{S} \in \mathcal{R}_q^{t \times n}, \mathbf{P} \in \mathcal{R}_q^{n \times m_P}, \text{aux} \in \{0, 1\}^*).$$

Let $\text{Pre1}_{\mathcal{A}}^b$, $\text{Pre2}_{\mathcal{A}}^b$ and $\text{Post}_{\mathcal{B}}^b$ be the experiments defined in Fig. 4 and denote

$$\text{Adv}_{\mathcal{A}}^{\text{Pre1}}(\lambda) := |\Pr[\text{Pre1}_{\mathcal{A}}^0(1^\lambda) = 1] - \Pr[\text{Pre1}_{\mathcal{A}}^1(1^\lambda) = 1]|,$$

$$\text{Adv}_{\mathcal{A}}^{\text{Pre2}}(\lambda) := |\Pr[\text{Pre2}_{\mathcal{A}}^0(1^\lambda) = 1] - \Pr[\text{Pre2}_{\mathcal{A}}^1(1^\lambda) = 1]|,$$

$$\text{Adv}_{\mathcal{B}}^{\text{Post}}(\lambda) := |\Pr[\text{Post}_{\mathcal{B}}^0(1^\lambda) = 1] - \Pr[\text{Post}_{\mathcal{B}}^1(1^\lambda) = 1]|.$$

The PrivateHideEvLWE_{param} assumption states that for any PPT Samp and \mathcal{B} there exists a PPT \mathcal{A} such that $\text{Adv}_{\mathcal{A}}^{\text{Pre1}}(\lambda) + \text{Adv}_{\mathcal{A}}^{\text{Pre2}}(\lambda) \geq \text{Adv}_{\mathcal{B}}^{\text{Post}}(\lambda)/\text{poly}(\lambda) - \text{negl}(\lambda)$.

The experiments Pre1, Post in Definition 9 are almost identical to the experiments Pre, Post in Definition 8, except that the matrix \mathbf{B} is not given to the

distinguishers \mathcal{A} and \mathcal{B} , and \mathbf{P} also not necessarily. The obvious distinction is the additional experiment Pre2 , which we define below.

Experiment Pre2. This experiment seeks to ensure that “both \mathbf{B}, \mathbf{P} are sufficiently hidden from the distinguishers”. First, observe that the indistinguishability of Pre1^0 and Pre1^1 , i.e., $(\underline{\mathbf{SB}}, \underline{\mathbf{SP}}, \text{aux}) \approx_c (\$, \$, \text{aux})$, implies that $(\underline{\mathbf{SB}}, \underline{\mathbf{SP}}, \text{aux})$ does not leak information about \mathbf{B} (computationally), since aux is independent of \mathbf{B} . Moreover, the only possible way for a PPT adversary \mathcal{A} to obtain information about \mathbf{P} is via aux . Experiment Pre2^b then ensures that the latter is also impossible, in that (\mathbf{P}, aux) and $(\mathbf{P} + \text{noise}, \text{aux})$ are indistinguishable. In words, given aux , no PPT \mathcal{A} can learn sufficiently about (an approximation of) \mathbf{P} , in that \mathbf{P} and $\mathbf{P} + \text{noise}$ look the same to \mathcal{A} .

On ℓ and $\mathcal{U}(\{0, 1, \dots, \ell\})$. The parameter ℓ parametrises the strength of Pre2 and the overall evasive LWE assumption. For example, if $\ell = q$, then Pre2 can be seen as requiring \mathbf{P} to be pseudorandom, and since this is the hardest case to achieve, the resulting Hiding Evasive LWE assumption is the weakest. As ℓ decreases, i.e. less noise is added to \mathbf{P} , Pre2 becomes easier to be satisfied, and the evasive LWE assumption grows stronger. The noise distribution $\mathcal{U}(\{0, 1, \dots, \ell\})$ may alternatively be replaced by other natural distributions, e.g. discrete Gaussian over \mathcal{R} , with the Gaussian parameter being a suitable ℓ parametrising the hardness of Pre2 .

Relating to Existing Definitions. Suppose $\ell = 0$ (which is disallowed in Definition 9), then the experiment Pre2 is trivial and Definition 9 collapses to the private-coin variant in [VWW22]. Letting $m_P = 2m$, our counterexamples prove that the assumption for this setting is false (conditioned on other appropriate parameters). On the other hand, even just by setting $\ell = 1$, we are unaware of counterexamples (except the obfuscation-based one by [VWW22], c.f. Remark 3). Moreover, in Sect. 6 we will see that Definition 9 with a large ℓ can be applied to the security proof of [VWW22], without altering their parameters.

Remark 7 (What if Pre2 restricts some but not all entries of \mathbf{P}). In the full version of this work, we sketch an alternative counterexample against the assumption in [VWW22], which demonstrates that it is necessary for most entries of \mathbf{P} to be irrecoverable, as even leaking only m entries of \mathbf{P} (for m the number of columns of \mathbf{B}) would lead to a successful distinguisher for Post^b .

Remark 8 (Alternative Pre2 candidate). Another way to potentially capture the intuition of “cannot approximate \mathbf{P} given aux ” might be to ask for indistinguishability of $\mathbf{P} \bmod \ell$ and $\mathbf{R} \bmod \ell$, for a uniform \mathbf{R} over \mathbb{Z}_q .¹² Other than the complication in formalisation and potential issues due to number-theoretic relations between q and ℓ , this alternative is also intuitively a weaker pre-condition

¹² These mean, fix the representatives of $\mathbb{Z}_q = \{0, 1, \dots, q - 1\}$ and map each entry in \mathbb{Z}_q to their representative over \mathbb{Z} , then further apply mod ℓ operation. For an intuition, if $\ell = 2^k$ is a power of 2, then $\mathbf{P} \bmod \ell$ may be interpreted as the last k bits of (each entry of) \mathbf{P} .

$\text{Pre}_{\mathcal{A}}^{b,\beta,\gamma}(1^\lambda)$	$\text{Post}_{\mathcal{B}}^{b,\beta,\gamma}(1^\lambda)$
$\mathbf{B} \leftarrow_{\$} \mathbb{Z}_q^{n \times m}$	$\mathbf{B} \leftarrow_{\$} \mathbb{Z}_q^{n \times m}$
if $\beta = 0$ then	if $\beta = 0$ then
$(\mathbf{P}, \text{aux}) \leftarrow \text{Samp}(1^\lambda)$	$(\mathbf{P}, \text{aux}) \leftarrow \text{Samp}(1^\lambda)$
if $\beta = 1$ then	if $\beta = 1$ then
$(\mathbf{P}, \text{aux}) \leftarrow \text{Samp}(1^\lambda, \mathbf{B})$	$(\mathbf{P}, \text{aux}) \leftarrow \text{Samp}(1^\lambda, \mathbf{B})$
if $b = 0$ then	if $b = 0$ then
$\mathbf{e}_0 \leftarrow_{\$} D_{\mathbb{Z}^m, \chi}, \mathbf{e}_0 \leftarrow_{\$} D_{\mathbb{Z}, \chi}, \mathbf{s} \leftarrow_{\$} \mathbb{Z}_q^n$	$\mathbf{e}_0 \leftarrow_{\$} D_{\mathbb{Z}, \chi}, \mathbf{s} \leftarrow_{\$} \mathbb{Z}_q^n$
$\mathbf{c} := \mathbf{s}^\top \mathbf{B} + \mathbf{e}_B^\top \bmod q$	$\mathbf{c} := \mathbf{s}^\top \mathbf{B} + \mathbf{e}_B^\top \bmod q$
$\mathbf{c}_P := \mathbf{s}^\top \mathbf{P} + \mathbf{e}_P^\top \bmod q$	$\mathbf{D} \leftarrow_{\$} D_{\Lambda_{\mathbf{P}}(\mathbf{B}), \chi}$
if $b = 1$ then	if $b = 1$ then
$\mathbf{c} \leftarrow_{\$} \mathbb{Z}_q^m$	$\mathbf{c} \leftarrow_{\$} \mathbb{Z}_q^m$
$\mathbf{c}_P \leftarrow_{\$} \mathbb{Z}_q^{m_P}$	$\mathbf{D} \leftarrow_{\$} D_{\Lambda_{\mathbf{P}}(\mathbf{B}), \chi}$
if $\gamma = 0$ then	if $\gamma = 0$ then
return $\mathcal{A}(\mathbf{c}, \mathbf{c}_P, \text{aux})$	return $\mathcal{B}(\mathbf{c}, \mathbf{D}, \text{aux})$
if $\gamma = 1$ then	if $\gamma \in \{1, 2\}$ then
return $\mathcal{A}(\mathbf{B}, \mathbf{c}, \mathbf{c}_P, \text{aux})$	return $\mathcal{B}(\mathbf{B}, \mathbf{c}, \mathbf{D}, \text{aux})$
if $\gamma = 2$ then	
return $\mathcal{A}(\mathbf{B}, \mathbf{c}, \mathbf{P}, \mathbf{c}_P, \text{aux})$	

Fig. 5. Experiments Pre and Post for private-coin evasive LWE variants, to which our counterexamples apply.

(hence leading to stronger evasive assumption). For more context, in the full version of this work we provide a heuristic counterexample against such alternative Pre2 for $\ell = 2$ and $m_P = O(m^2)$, which may be further generalised to be against constant ℓ and $m_P = O(m^\ell)$ [AG11, NMSÜ24].

5 Counterexamples to Existing Variants

We present our counterexamples against a number of existing evasive LWE variants sketched in Sect. 2.2. These variants are formally defined in Fig. 5 and parametrised by β, γ , each controlling if Samp receives \mathbf{B} as input or not, and if the distinguishers receive $\mathbf{B}, (\mathbf{B}, \mathbf{P})$, or none. We remark that this definition is a special case of the private-coin variants in existing works, where we consider the restricted setting of the LWE sample \mathbf{s} being sampled honestly by the experiments, not available to Samp.

Denote the advantages of distinguishers \mathcal{A}, \mathcal{B} and $\beta \in \{0, 1\}, \gamma \in \{0, 1, 2\}$ by

$$\begin{aligned} \text{Adv}_{\mathcal{A}}^{\text{Pre}, \beta, \gamma}(\lambda) &= \left| \Pr[\text{Pre}_{\mathcal{A}}^{0, \beta, \gamma}(1^\lambda) = 0] - \Pr[\text{Pre}_{\mathcal{A}}^{1, \beta, \gamma}(1^\lambda) = 0] \right| \\ \text{Adv}_{\mathcal{A}}^{\text{Post}, \beta, \gamma}(\lambda) &= \left| \Pr[\text{Post}_{\mathcal{A}}^{0, \beta, \gamma}(1^\lambda) = 0] - \Pr[\text{Post}_{\mathcal{A}}^{1, \beta, \gamma}(1^\lambda) = 0] \right| \end{aligned}$$

In all counterexamples, we consider the usual case of the ring of integers $\mathcal{R} = \mathbb{Z}$ and assume that error noise is distributed according to a discrete Gaussian distribution $D_{\mathbb{Z}^m, \chi}$ for a parameter $\chi > 0$ to be specified. We require q to be prime, and abide to the following parameter restrictions for n, m, χ, q

$$m \geq 3n \log(q), \quad \chi \geq \lambda m, \quad q \geq \lambda m^2 \chi^2.$$

Remark 9 (On Parameters). The above parameters are polynomial (in particular the modulus q can be chosen to lie in $O(n^4 \lambda^4)$, for example) and the attacks we propose in the following have a high advantage of at least $1 - O(1/\lambda)$ to win the then challenge. It is possible to increase the advantage to be overwhelming in λ , by choosing $q \gg \chi \gg m$ such that q is super-polynomially larger than χ , and χ is super-polynomially larger than m .

The following lemma upper-bounds the probability of a square Gaussian matrix sampled over random cosets being not invertible over \mathbb{Z}_q , which be useful for our counterexamples. Its proof in more generality is given in the full version of this work.

Lemma 5. *Let $\mathbf{B}, \mathbf{P}_1 \leftarrow_{\mathcal{S}} \mathbb{Z}_q^{n \times m}$ be uniformly random. Let $\chi \in \omega(\sqrt{n})$, $m \geq 2n \log q$ and let $q > n \cdot \chi$ be prime. We have*

$$\Pr_{\mathbf{D}_1 \leftarrow_{\mathcal{S}} D_{\Lambda_{\mathbf{P}_1}^\perp(\mathbf{B}), \chi}} [\det(\mathbf{D}_1) = 0 \pmod q] \leq O(m/\chi).$$

Remark 10 (Alternative to Lemma 5). As an alternative to Lemma 5, one can use a similar result by Regev [Reg05], which states that m^2 many Gaussian vectors $\mathbf{d}_1, \dots, \mathbf{d}_{m^2} \leftarrow_{\mathcal{S}} D_{\mathbb{Z}^m, \chi}$ will contain a basis of \mathbb{Z}^m with probability at least $1 - 2^{-\Omega(n)}$. Correspondingly, the numbers of columns of the matrices \mathbf{P}_1 and \mathbf{P}_3 in Sects. 5.1 and 5.2 is required to increase from m to m^2 .

5.1 Counterexample 1

We give a counterexample for the case $\beta = 0, \gamma = 1$, i.e., \mathbf{B} is given to the distinguishers, but $\mathbf{P} = (\mathbf{P}_1, \mathbf{P}_2)$ is not and \mathbf{aux} is empty. Our idea is to embed a trapdoor in \mathbf{P}_2 , which we use to distinguish $\mathbf{s}^T \mathbf{B} + \mathbf{e}_0^T$ from uniform randomness in the Post^b challenge. Concretely, let $\text{Samp}_1(1^\lambda)$ output the following:

$$(\mathbf{P} = (\mathbf{P}_1, \mathbf{P}_2), \mathbf{aux} = \perp)$$

where

$$\begin{aligned} \mathbf{u}' &\leftarrow_{\mathcal{S}} \{0, 1\}^{m-1}, & \mathbf{P}'_1 &\leftarrow_{\mathcal{S}} \mathbb{Z}_q^{n \times (m-1)}, & \mathbf{P}'_2 &\leftarrow_{\mathcal{S}} \mathbb{Z}_q^{(n-1) \times m}, \\ \mathbf{u}^T &= ((\mathbf{u}')^T, 1) \in \{0, 1\}^{1 \times m}, & \mathbf{P}_1 &= (\mathbf{P}'_1 \mid -\mathbf{P}'_1 \mathbf{u}' \pmod q), & \mathbf{P}_2 &= \begin{pmatrix} \mathbf{P}'_2 \\ \mathbf{u}^T \end{pmatrix}. \end{aligned}$$

Proposition 1. *Let \mathcal{A} be a PPT adversary. Under the $\text{LWE}_{\mathbb{Z},q,n,2m,\chi}$ assumption, we have for the experiment $\text{Pre}_{\mathcal{A}}^{\beta=0,\gamma=1}(1^\lambda)$ in Fig. 5 instantiated with Samp_1*

$$\text{Adv}_{\mathcal{A}}^{\text{Pre},0,1}(\lambda) \in \text{negl}(\lambda).$$

Proof. The proof proceeds via four hybrid experiments:

\mathcal{D}_0 : The joint distribution of the ifstatement, i.e.

$$(\mathbf{B}, \quad \mathbf{s}^\top \mathbf{B} + \mathbf{e}_0^\top \bmod q, \quad \mathbf{s}^\top \mathbf{P}_1 + \mathbf{e}_1^\top \bmod q, \quad \mathbf{s}^\top \mathbf{P}_2 + \mathbf{e}_2^\top \bmod q)$$

where $\mathbf{B} \leftarrow_{\mathfrak{s}} \mathcal{R}_q^{n \times m}$, $\mathbf{s} \leftarrow_{\mathfrak{s}} \mathbb{Z}_q^n$, $\mathbf{e}_0 \leftarrow_{\mathfrak{s}} \chi^m$, $\mathbf{e}_1 \leftarrow_{\mathfrak{s}} \chi^m$, $\mathbf{e}_2 \leftarrow_{\mathfrak{s}} \chi^m$.

\mathcal{D}_1 : The last term $\mathbf{s}^\top \mathbf{P}_2 + \mathbf{e}_2^\top \bmod q$ is replaced by a random vector \mathbf{y}_2 , i.e.

$$(\mathbf{B}, \quad \mathbf{s}^\top \mathbf{B} + \mathbf{e}_0^\top \bmod q, \quad \mathbf{s}^\top \mathbf{P}_1 + \mathbf{e}_1^\top \bmod q, \quad \mathbf{y}_2)$$

for $\mathbf{y}_2 \leftarrow_{\mathfrak{s}} \mathbb{Z}_q^m$.

\mathcal{D}_2 : As \mathcal{D}_1 , but \mathbf{P}_1 is swapped to random.

\mathcal{D}_3 : $\mathbf{s}^\top \mathbf{B} + \mathbf{e}_0^\top \bmod q$ and $\mathbf{s}^\top \mathbf{P}_1 + \mathbf{e}_1^\top \bmod q$ are swapped to random, i.e.

$$(\mathbf{B}, \quad \mathbf{y}_0, \quad \mathbf{y}_1, \quad \mathbf{y}_2)$$

for $\mathbf{y}_0, \mathbf{y}_1, \mathbf{y}_2 \leftarrow_{\mathfrak{s}} \mathbb{Z}_q^m$.

The statistical distance between \mathcal{D}_0 and \mathcal{D}_1 is bounded by q^{-n+1} . This is because for $\mathbf{s} = (\mathbf{s}', s_n) \leftarrow_{\mathfrak{s}} \mathbb{Z}_q^n$, $\mathbf{s}^\top \mathbf{P}_2 = \mathbf{s}'^\top \mathbf{P}'_2 + s_n \cdot \mathbf{u}^\top \bmod q$ is uniformly random if \mathbf{s}' is non-zero (even if we know \mathbf{B} , \mathbf{P}_1 , \mathbf{s} and \mathbf{u}).

Since $m \geq 2n \log q$, the Leftover-Hash Lemma 3 implies that the statistical distance between $\mathbf{P}_1 = (\mathbf{P}'_1 | -\mathbf{P}'_1 \mathbf{u}' \bmod q)$ and a uniformly random matrix is bounded by 2^{-n} . Hence, the statistical distance between \mathcal{D}_1 and \mathcal{D}_2 is also bounded by 2^{-n} . Finally, $\text{LWE}_{\mathbb{Z},q,n,2m,\chi}$ states that \mathcal{D}_2 and \mathcal{D}_3 are computationally indistinguishable.

Therefore, \mathcal{A} 's distinguishing advantage between $\text{Pre}_{\mathcal{A}}^{0,0,1}(1^\lambda) = \mathcal{D}_0$ and $\text{Pre}_{\mathcal{A}}^{1,0,1}(1^\lambda) = \mathcal{D}_3$ is bounded by 2^{-n+1} plus a negligible term stemming from the LWE assumption. \square

We introduce two more lemmas, their proofs are given in the full version.

Lemma 6. *Let $(\mathbf{P}_1 \in \mathbb{Z}_q^{n \times m}, \mathbf{u} \in \{0, 1\}^m) \leftarrow \text{Samp}_1(1^\lambda)$. For $\mathbf{B} \leftarrow_{\mathfrak{s}} \mathbb{Z}_q^{n \times m}$ and $\mathbf{D}_1 \leftarrow D_{\Lambda_{\mathbf{P}_1}^\perp(\mathbf{B}), \chi}$, we have $\Pr[\mathbf{D}_1 \cdot \mathbf{u} = 0 \bmod q] \leq O(m/\chi)$.*

Lemma 7. *Let $(\mathbf{P}_1 \in \mathbb{Z}_q^{n \times m}, \mathbf{u} \in \{0, 1\}^m) \leftarrow \text{Samp}_1(1^\lambda)$. Sample $\mathbf{B} \leftarrow_{\mathfrak{s}} \mathbb{Z}_q^{n \times m}$, $\mathbf{e}_0 \leftarrow_{\mathfrak{s}} D_{\mathbb{Z}^m, \chi}$ and $\mathbf{D}_1 \leftarrow_{\mathfrak{s}} D_{\Lambda_{\mathbf{P}_1}^\perp(\mathbf{B}), \chi}$. We have $\Pr[\|\mathbf{e}_0^\top \cdot \mathbf{D}_1 \cdot \mathbf{u}\| \geq m^2 \chi^2] \leq 2^{-\lambda}$.*

Proposition 2. *There is a PPT adversary \mathcal{B} s.t. we have for the experiment $\text{Post}_{\mathcal{B}}^{\beta=0,\gamma=1}(1^\lambda)$ in Fig. 5 instantiated with Samp_1*

$$\text{Adv}_{\mathcal{B}}^{\text{Pre},0,1}(\lambda) \geq 1 - O(1/\lambda).$$

Proof. Let $\mathbf{B} \leftarrow_{\S} \mathbb{Z}_q^{n \times m}$ and $(\mathbf{S}, \mathbf{P}, \mathbf{aux} = \perp) \leftarrow \text{Samp}_1(1^\lambda)$. Sample $\mathbf{D} \leftarrow_{\S} D_{\Lambda_{\mathbf{P}}^\perp(\mathbf{B}), \chi}$ and note that $\mathbf{D} \in \mathbb{Z}^{n \times 2m}$ can be split into two equally large parts $\mathbf{D} = (\mathbf{D}_1 | \mathbf{D}_2)$, which are distributed as

$$\mathbf{D}_1 \leftarrow_{\S} D_{\Lambda_{\mathbf{P}_1}^\perp(\mathbf{B}), \chi} \quad \text{and} \quad \mathbf{D}_2 \leftarrow_{\S} D_{\Lambda_{\mathbf{P}_2}^\perp(\mathbf{B}), \chi}.$$

Recall that in the experiment $\text{Post}_{\mathcal{B}}^{\beta=0, \gamma=1}(1^\lambda)$, \mathcal{B} has to decide if \mathbf{c}^\top in

$$(\mathbf{B}, \mathbf{D}_1, \mathbf{D}_2, \mathbf{c}^\top)$$

equals $\mathbf{s}^\top \mathbf{B} + \mathbf{e}_0^\top \bmod q$, for $\mathbf{e}_0 \leftarrow_{\S} D_{\mathbb{Z}^m, \chi}$ and $\mathbf{s} \leftarrow_{\S} \mathbb{Z}_q^n$, or has been sampled uniformly at random from $\mathbb{Z}_q^{1 \times m}$.

Our adversary \mathcal{B} proceeds as follows:

1. It recovers $\mathbf{P}_2 := \mathbf{B} \cdot \mathbf{D}_2 \bmod q$.
2. Since Samp_1 samples \mathbf{P}_2 as $\begin{pmatrix} \mathbf{P}'_2 \\ \mathbf{u}^\top \end{pmatrix}$, \mathcal{B} can extract the binary vector $\mathbf{u} \in \{0, 1\}^m$ from the last row of \mathbf{P}_2 .
3. \mathcal{B} computes $r := \mathbf{c}^\top \cdot \mathbf{D}_1 \cdot \mathbf{u} \bmod q$.
4. If $r \in \{-\chi^2 m^2, \dots, \chi^2 m^2\} \subset \mathbb{Z}_q$, then \mathcal{B} outputs 0. Otherwise, it outputs 1.

If \mathbf{c} is drawn uniformly at random from \mathbb{Z}_q^m , then the probability that r lies in $\{-\chi^2 m^2, \dots, \chi^2 m^2\}$ is bounded by $O(\chi^2 m^2 / q) \subseteq O(1/\lambda)$. This is because r is distributed uniformly at random in \mathbb{Z}_q if $\mathbf{D}_1 \cdot \mathbf{u} \bmod q$ is non-zero, which happens with probability at least $1 - O(m/\chi) = 1 - O(1/\lambda)$ by Lemma 6. Hence, if \mathbf{c} is uniformly random, \mathcal{B} outputs 1 with probability $1 - O(1/\lambda)$.

Else, if $\mathbf{c} = \mathbf{B}^\top \mathbf{s} + \mathbf{e}_0 \bmod q$ for $\mathbf{e}_0 \leftarrow_{\S} D_{\mathbb{Z}^m, \chi}$ and $\mathbf{s} \leftarrow_{\S} \mathbb{Z}_q^n$, then

$$\begin{aligned} r &= \mathbf{c}^\top \cdot \mathbf{D}_1 \mathbf{u} = (\mathbf{s}^\top \mathbf{B} + \mathbf{e}_0^\top) \cdot \mathbf{D}_1 \mathbf{u} = \mathbf{s}^\top \mathbf{B} \cdot \mathbf{D}_1 \mathbf{u} + \mathbf{e}_0^\top \cdot \mathbf{D}_1 \mathbf{u} \\ &= \mathbf{s}^\top \mathbf{P}_1 \mathbf{u} + \mathbf{e}_0^\top \mathbf{D}_1 \mathbf{u} = \mathbf{e}_0^\top \mathbf{D}_1 \mathbf{u} \bmod q. \end{aligned}$$

By Lemma 7, $\|\mathbf{e}_0^\top \mathbf{D}_1 \mathbf{u}\| \leq m^2 \chi^2$ with probability at least $1 - 2^{-\lambda}$. Hence, \mathcal{B} will output 0 if $\mathbf{c} = \mathbf{B}^\top \mathbf{s} + \mathbf{e}_0 \bmod q$ with overwhelming probability. It follows that the advantage of \mathcal{B} lies in $1 - O(1/\lambda)$. \square

5.2 Counterexample 2

We give a counterexample for the case $\beta = \gamma = 0$, i.e. \mathbf{B} is not given to the distinguisher. The sampler is similar to that in Sect. 5.1, except that we add a third part $\mathbf{P}_3 \leftarrow_{\S} \mathbb{Z}_q^{n \times m}$ to \mathbf{P} , which is also included in \mathbf{aux} . While \mathbf{P}_3 is harmless on its own, the distinguisher for Post^b can use it to recover \mathbf{B} and continue as in the first counterexample. Concretely, let $\text{Samp}_2(1^\lambda)$ output the following:

$$(\mathbf{P} = (\mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3), \quad \mathbf{aux} = \mathbf{P}_3)$$

where $\mathbf{P}_3 \leftarrow_{\S} \mathbb{Z}_q^{n \times m}$ and

$$\begin{aligned} \mathbf{u}' &\leftarrow_{\S} \{0, 1\}^{m-1}, & \mathbf{P}'_1 &\leftarrow_{\S} \mathbb{Z}_q^{n \times (m-1)}, & \mathbf{P}'_2 &\leftarrow_{\S} \mathbb{Z}_q^{(n-1) \times m}, \\ \mathbf{u}^\top &= ((\mathbf{u}')^\top, 1) \in \{0, 1\}^{1 \times m}, & \mathbf{P}_1 &= (\mathbf{P}'_1 | -\mathbf{P}'_1 \mathbf{u}' \bmod q), & \mathbf{P}_2 &= \begin{pmatrix} \mathbf{P}'_2 \\ \mathbf{u}^\top \end{pmatrix}. \end{aligned}$$

Proposition 3. *Let \mathcal{A} be a PPT adversary. Under the $\text{LWE}_{\mathbb{Z},q,n,3m,\chi}$ assumption, we have for the experiment $\text{Pre}_{\mathcal{A}}^{\beta=0,\gamma=0}(1^\lambda)$ in Fig. 5 instantiated with Samp_2*

$$\text{Adv}_{\mathcal{A}}^{\text{Pre},0,0}(\lambda) \in \text{negl}(\lambda).$$

Proof. Let $\mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3$ be the output of $\text{Samp}_2(1^\lambda)$. Note that in this case, \mathcal{A} has to distinguish between the distribution

$$(\mathbf{s}^\top \mathbf{B} + \mathbf{e}_0^\top, \mathbf{s}^\top \mathbf{P}_1 + \mathbf{e}_1^\top, \mathbf{s}^\top \mathbf{P}_2 + \mathbf{e}_2^\top, \mathbf{s}^\top \mathbf{P}_3 + \mathbf{e}_3^\top, \mathbf{P}_3) \bmod q \quad (9)$$

for $\mathbf{B} \leftarrow_{\mathcal{S}} \mathcal{R}_q^{n \times m}$, $\mathbf{s} \leftarrow_{\mathcal{S}} \mathbb{Z}_q^n$, $\mathbf{e}_0 \leftarrow_{\mathcal{S}} \chi^m$, $\mathbf{e}_1 \leftarrow_{\mathcal{S}} \chi^m$, $\mathbf{e}_2 \leftarrow_{\mathcal{S}} \chi^m$, and the distribution

$$(\mathbf{y}_0, \mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_3, \mathbf{P}_3),$$

for $\mathbf{y}_0, \mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_3 \leftarrow_{\mathcal{S}} \mathbb{Z}_q^m$. By the same argument as in the proof of Proposition 1, the statistical distance between the distribution in Eq. (9) and

$$(\mathbf{s}^\top \mathbf{B} + \mathbf{e}_0^\top \bmod q, \mathbf{s}^\top \mathbf{P}'_1 + \mathbf{e}_1^\top \bmod q, \mathbf{y}_2, \mathbf{s}^\top \mathbf{P}_3 + \mathbf{e}_3^\top \bmod q, \mathbf{P}_3), \quad (10)$$

for $\mathbf{P}'_1 \leftarrow_{\mathcal{S}} \mathbb{Z}_q^{n \times m}$ and $\mathbf{y}_2 \leftarrow_{\mathcal{S}} \mathbb{Z}_q^m$ is bounded by 2^{-n+1} . Now, the claim follows by invoking $\text{LWE}_{\mathbb{Z},q,n,3m,\chi}$. \square

Proposition 4. *There is a PPT adversary \mathcal{B} s.t. we have for the experiment $\text{Post}_{\mathcal{B}}^{\beta=0,\gamma=0}(1^\lambda)$ in Fig. 5 instantiated with Samp_2*

$$\text{Adv}_{\mathcal{B}}^{\text{Pre},0,0}(\lambda) \geq 1 - O(1/\lambda).$$

Proof. For $i = 1, 2, 3$, let $\mathbf{D}_i \leftarrow_{\mathcal{S}} D_{\Lambda_{\mathbf{P}_i}^\perp(\mathbf{B}),\chi}$ be the short preimages that \mathcal{B} is given in Post^b . Denote the adversary of Proposition 2 from counterexample 1 by \mathcal{B}' . We want to invoke \mathcal{B}' on Post^b , however, note that \mathbf{B} is missing.

Since \mathbf{B} and \mathbf{P}_3 have been sampled uniformly at random, Lemma 5 implies that $\mathbf{D}_3 \bmod q$ is invertible with probability $1 - O(1/\lambda)$. Hence, \mathcal{B} on input

$$(\mathbf{D}_1, \mathbf{D}_2, \mathbf{D}_3, \mathbf{c}^\top, \text{aux} = \mathbf{P}_3)$$

proceeds as follows:

1. If $\mathbf{D}_3 \bmod q$ is not invertible, then \mathcal{B} outputs a random bit and stops.
2. If $\mathbf{D}_3 \bmod q$ is invertible, \mathcal{B} computes $\mathbf{B} = \mathbf{D}_3^{-1} \cdot \mathbf{P}_3 \bmod q$.
3. \mathcal{B} runs $\mathcal{B}'(\mathbf{B}, \mathbf{D}_1, \mathbf{D}_2, \mathbf{c}^\top)$ and defers its output to the post challenger.

By Lemma 5 and Proposition 1, the advantage of \mathcal{B} is at least $1 - O(1/\lambda)$. \square

5.3 Counterexample 3

We give a counterexample for the cases $\beta = 1, \gamma \in \{0, 1, 2\}$. Note that $\beta = 1$ implies that Samp gets \mathbf{B} as input. We prove indistinguishability of Pre^b in the strongest case, where the joint distribution contains \mathbf{B} and \mathbf{P} , i.e. $\gamma = 2$. We

also show that, there exists PPT distinguisher for the weakest Post^b challenge, where the joint distribution will contain neither \mathbf{P} nor \mathbf{B} , i.e. $\gamma = 0$.

Let $\text{Samp}_3(1^\lambda, \mathbf{B})$, on input \mathbf{B} , output a matrix \mathbf{P} constructed as follows:

$$\mathbf{U} = (\mathbf{u}_1 | \mathbf{u}_2) \leftarrow_{\mathcal{S}} \{0, 1\}^{m \times 2}, \quad \mathbf{P} = (\mathbf{p}_1 | \mathbf{p}_2) := \mathbf{B}\mathbf{U} \bmod q \in \mathbb{Z}_q^{n \times 2}.$$

Additionally, Samp_3 outputs aux , which consists of the Dual Regev encryptions (under $(\mathbf{B} | \mathbf{p}_1) \in \mathbb{Z}_q^{n \times (m+1)}$) of the bits $u_{2,1}, \dots, u_{2,m}$ of \mathbf{u} , that is,

$$\text{aux} = \left(\mathbf{r}_i^T \mathbf{B} + \mathbf{f}'_i \bmod q, \quad \mathbf{r}_i^T \mathbf{p}_1 + f_{i,m+1} + \left\lfloor \frac{q}{2} \right\rfloor \cdot u_{2,i} \bmod q \right)_{i \in [m]} \in \mathbb{Z}_q^{m \times (m+1)}$$

where Samp_3 samples $\mathbf{r}_1, \dots, \mathbf{r}_m \leftarrow_{\mathcal{S}} \mathbb{Z}_q^n$ and $\mathbf{f}_i = (\mathbf{f}'_i, f_{i,m+1}) \leftarrow_{\mathcal{S}} D_{\mathbb{Z}^{m+1}, \chi}$.

Proposition 5. *Let \mathcal{A} be a PPT adversary. Under the $\text{LWE}_{\mathbb{Z}, q, n, m+2, \chi}$ assumption, we have for the experiment $\text{Pre}_{\mathcal{A}}^{\beta=1, \gamma=2}(1^\lambda)$ in Fig. 5 instantiated with Samp_3*

$$\text{Adv}_{\mathcal{A}}^{\text{Pre}, 1, 2}(\lambda) \in \text{negl}(\lambda).$$

Proof. We proceed via the following hybrid experiments:

\mathcal{D}_0 : This distribution corresponds to the view of \mathcal{A} in $\text{Pre}^{0,1,2}(1^\lambda)$, i.e.

$$\left(\mathbf{B}, \quad \mathbf{s}^T \mathbf{B} + \mathbf{e}_0^T \bmod q, \quad \mathbf{p}_1, \quad \mathbf{p}_2, \quad \mathbf{s}^T \mathbf{p}_1 + e_1^T \bmod q, \quad \mathbf{s}^T \mathbf{p}_2 + e_2^T \bmod q, \right. \\ \left. \text{aux} = \left(\mathbf{r}_i^T \mathbf{B} + \mathbf{f}'_i \bmod q, \quad \mathbf{r}_i^T \mathbf{p}_1 + f_{i,m+1} + \left\lfloor \frac{q}{2} \right\rfloor \cdot u_{2,i} \bmod q \right)_{i=1, \dots, m} \right)$$

where $\mathbf{B} \leftarrow_{\mathcal{S}} \mathbb{Z}_q^{n \times m}$, $(\mathbf{p}_1, \mathbf{p}_2, \text{aux}) \leftarrow \text{Samp}_3(\mathbf{B}, 1^\lambda)$, $\mathbf{s} \leftarrow_{\mathcal{S}} \mathbb{Z}_q^n$, $\mathbf{e}_0, \leftarrow_{\mathcal{S}} D_{\mathbb{Z}^m, \chi}$, $e_1, e_2 \leftarrow_{\mathcal{S}} D_{\mathbb{Z}, \chi}$.

\mathcal{D}_1 : \mathcal{D}_1 resembles \mathcal{D}_0 , but we replace \mathbf{p}_1 in the joint distribution and in the auxiliary information aux of the sampler with a uniformly random vector $\mathbf{p}'_1 \leftarrow_{\mathcal{S}} \mathbb{Z}_q^m$.

\mathcal{D}_2 : We replace $\text{aux} = \left(\mathbf{r}_i^T \mathbf{B} + \mathbf{f}'_i \bmod q, \mathbf{r}_i^T \mathbf{p}'_1 + f_{i,m+1} + \left\lfloor \frac{q}{2} \right\rfloor \cdot u_{2,i} \bmod q \right)_{i \in [m]}$

by a uniformly random matrix $\text{aux}' \leftarrow_{\mathcal{S}} \mathbb{Z}_q^{m \times m+1}$ in \mathcal{D}_1 .

\mathcal{D}_3 : We replace \mathbf{p}_2 by a uniformly random vector in \mathcal{D}_2 .

\mathcal{D}_4 : We replace $\mathbf{s}^T \mathbf{B} + \mathbf{e}_0^T$, $\mathbf{s}^T \mathbf{p}'_1 + e_1^T$, and $\mathbf{s}^T \mathbf{p}'_2 + e_2^T$ by uniformly random vectors and numbers over \mathbb{Z}_q^n .

\mathcal{D}_5 : We swap \mathbf{p}'_2 back to \mathbf{p}_2 in \mathcal{D}_4 .

\mathcal{D}_6 : We revert the changes on aux' and put again

$$\text{aux} = \left(\mathbf{r}_i^T \mathbf{B} + \mathbf{f}'_i \bmod q, \mathbf{r}_i^T \mathbf{p}'_1 + f_{i,m+1} + \left\lfloor \frac{q}{2} \right\rfloor \cdot u_{2,i} \bmod q \right)_{i=1, \dots, m}.$$

\mathcal{D}_7 : Finally, we replace the uniformly random vector \mathbf{p}'_1 in \mathcal{D}_6 by the vector $\mathbf{p}_1 = \mathbf{B}\mathbf{u}_1$ outputted by the sampler. Note that \mathcal{D}_7 equals now

$$\left(\mathbf{B}, \quad \mathbf{c}_0, \quad \mathbf{p}_1, \quad \mathbf{p}_2, \quad c'_1, \quad c'_2, \right. \\ \left. \text{aux} = \left(\mathbf{r}_i^T \mathbf{B} + \mathbf{f}'_i \bmod q, \quad \mathbf{r}_i^T \mathbf{p}_1 + f_{i,m+1} + \left\lfloor \frac{q}{2} \right\rfloor \cdot u_{2,i} \bmod q \right)_{i=1, \dots, m} \right)$$

for $\mathbf{c}_0 \leftarrow_{\mathcal{S}} \mathbb{Z}_q^n$, $c'_1, c'_2 \leftarrow_{\mathcal{S}} \mathbb{Z}_q$. Hence, \mathcal{D}_7 is the view of \mathcal{A} in $\text{Pre}^{1,1,2}(1^\lambda)$.

We claim that the advantage of \mathcal{A} to distinguish between the hybrids is negligible: By the Leftover-Hash Lemma 3, the statistical distance between \mathcal{D}_0 and \mathcal{D}_1 , between \mathcal{D}_2 and \mathcal{D}_3 , between \mathcal{D}_4 and \mathcal{D}_5 and between \mathcal{D}_6 and \mathcal{D}_7 is bounded by 2^{-n} . Going from \mathcal{D}_1 to \mathcal{D}_2 , we replace the m -fold dual-Regev encryption \mathbf{aux} by a uniformly random matrix $\mathbf{aux}' \leftarrow_{\$} \mathbb{Z}_q^{m \times (m+1)}$, which we revert again when we go from \mathcal{D}_5 to \mathcal{D}_6 . In total, we invoke the $\text{LWE}_{\mathbb{Z}, q, n, m+1, \chi}$ assumption in both directions m times. Finally, the $\text{LWE}_{\mathbb{Z}, q, n, m+2, \chi}$ assumption stipulates that it is hard for \mathcal{A} to distinguish between \mathcal{D}_3 and \mathcal{D}_4 .

Concluding, the advantage of \mathcal{A} to win $\text{Pre}^{\beta=1, \gamma=2}(1^\lambda)$ is bounded by $O(2^{-n})$ plus a negligible term stemming from $(2m+1)$ -times of invoking LWE. \square

We require two more lemmas, their proofs are given in the full version.

Lemma 8. For $\mathbf{B} \leftarrow_{\$} \mathbb{Z}_q^{n \times m}$, $\mathbf{u}_2 \leftarrow_{\$} \{0, 1\}^m$, $\mathbf{p}_2 = \mathbf{B} \cdot \mathbf{u}_2$ and $\mathbf{d}_2 \leftarrow_{\$} D_{\Lambda_{\mathbf{p}_2}^\perp}(\mathbf{B}, \chi)$, we have $\Pr[\mathbf{d}_2 = \mathbf{u}_2] \leq 2^{-m+1} + q^{-n}$.

Lemma 9. Draw $\mathbf{B} \leftarrow_{\$} \mathbb{Z}_q^{n \times m}$, $\mathbf{u}_2 \leftarrow_{\$} \{0, 1\}^m$ and set $\mathbf{p}_2 = \mathbf{B}\mathbf{u}_2 \bmod q$. Draw $\mathbf{d}_2 \leftarrow_{\$} D_{\Lambda_{\mathbf{p}_2}^\perp}(\mathbf{B}, \chi)$ and $\mathbf{f}'_1, \dots, \mathbf{f}'_m \leftarrow_{\$} D_{\mathbb{Z}^m, \chi}$.

We have $\Pr[\exists i \in [m] : \|\mathbf{f}'_i \cdot \mathbf{d}_2\| \geq m\chi^2] \leq 2^{-\lambda}$. Additionally, we have for $\mathbf{e}_0 \leftarrow_{\$} D_{\mathbb{Z}_q^m, \chi}$, $\Pr[\|\mathbf{e}_0^\top \cdot (\mathbf{d}_2 - \mathbf{u}_2)\| \geq \chi(\chi+1) \cdot m] \leq 2^{-\lambda}$.

Proposition 6. There is a PPT adversary \mathcal{B} s.t. we have for the experiment $\text{Post}_{\mathcal{B}}^{\beta=1, \gamma=0}(1^\lambda)$ in Fig. 5 instantiated with Samp_3

$$\text{Adv}_{\mathcal{B}}^{\text{Pre}, 1, 0}(\lambda) \geq 1 - O(1/(\lambda m)).$$

Proof. Recall that the post challenge consists of

$$\left(\begin{array}{c} \mathbf{c}, \quad \mathbf{d}_1^\top, \quad \mathbf{d}_2^\top, \\ \mathbf{aux} = \left(\mathbf{r}_i^\top \mathbf{B} + \mathbf{f}'_i^\top \bmod q, \quad \mathbf{r}_i^\top \mathbf{p}_1 + f_{i, m+1} + \left\lfloor \frac{q}{2} \right\rfloor \cdot u_{2, i} \bmod q \right)_{i=1, \dots, m} \end{array} \right)$$

for $\mathbf{B} \leftarrow_{\$} \mathbb{Z}_q^{n \times m}$, $(\mathbf{p}_1, \mathbf{p}_2, \mathbf{aux}) \leftarrow \text{Samp}_3(\mathbf{B}, 1^\lambda)$, $\mathbf{d}_1 \leftarrow_{\$} D_{\Lambda_{\mathbf{p}_1}^\perp}(\mathbf{B}, \chi)$, $\mathbf{d}_2 \leftarrow_{\$} D_{\Lambda_{\mathbf{p}_2}^\perp}(\mathbf{B}, \chi)$. \mathcal{B} has to decide if \mathbf{c} has been sampled uniformly at random from \mathbb{Z}_q^m or is of shape $\mathbf{s}^\top \mathbf{B} + \mathbf{e}_0^\top \bmod q$ for $\mathbf{s} \leftarrow_{\$} \mathbb{Z}_q^n$, $\mathbf{e}_0 \leftarrow_{\$} D_{\mathbb{Z}^m, \chi}$.

The distinguisher \mathcal{B} , we propose, proceeds as follows:

1. For $i \in [m]$, it computes

$$\begin{aligned} g_i &:= \mathbf{r}_i^\top \mathbf{p}_1 + f_{i, m+1} + \left\lfloor \frac{q}{2} \right\rfloor \cdot u_{2, i} - (\mathbf{r}_i^\top \mathbf{B} + \mathbf{f}'_i^\top) \cdot \mathbf{d}_1 && \bmod q \\ &= \mathbf{r}_i^\top \mathbf{p}_1 + f_{i, m+1} + \left\lfloor \frac{q}{2} \right\rfloor \cdot u_{2, i} - \mathbf{r}_i^\top \cdot \mathbf{p}_1 - \mathbf{f}'_i^\top \cdot \mathbf{d}_1 && \bmod q \\ &= f_{i, m+1} - \mathbf{f}'_i^\top \cdot \mathbf{d}_1 + \left\lfloor \frac{q}{2} \right\rfloor \cdot u_{2, i} && \bmod q. \end{aligned}$$

Further, it sets $u'_i = 1$ if $|g_i| \geq \frac{q}{4}$ and else 0, and $\mathbf{u}' = (u'_1, \dots, u'_m) \in \{0, 1\}^m$.

2. It computes $r := \mathbf{c}^\top \cdot (\mathbf{d}_2 - \mathbf{u}')$ mod q .

3. If $r \in \{-\chi(\chi + 1) \cdot m, \dots, \chi(\chi + 1) \cdot m\} \subset \mathbb{Z}_q$, output 0. Else, output 1.

We claim that the dual-Regev decryption of $u_{2,1}, \dots, u_{2,m}$ in step 1 succeeds with overwhelming probability. Indeed, we have $u'_i = u_{2,i}$ whenever $\|f_{i,m+1} - \mathbf{f}'_i \cdot \mathbf{d}_1\|$ is bounded by $q/4$. Lemma 9 guarantees that we have

$$\|f_{i,m+1} - \mathbf{f}'_i \cdot \mathbf{d}_1\| < m\chi^2 < \frac{q}{4}$$

with overwhelming probability $\geq 1 - 2^{-\lambda}$.

Hence, assume that decryption succeeds, which happens with overwhelming probability, and that \mathcal{B} can recover $\mathbf{u}' = \mathbf{u}_2$. Assume that $\mathbf{d}_2 - \mathbf{u}' = \mathbf{d}_2 - \mathbf{u}_2$ is not zero. We now distinguish two cases:

If $\mathbf{c} = \mathbf{B}^T \mathbf{s} + \mathbf{e}_0$, then we have

$$\begin{aligned} r &= \mathbf{c}^T \cdot (\mathbf{d}_2 - \mathbf{u}') = (\mathbf{s}^T \mathbf{B} + \mathbf{e}_0^T) \cdot (\mathbf{d}_2 - \mathbf{u}') \bmod q \\ &= \mathbf{s}^T \mathbf{B} \cdot (\mathbf{d}_2 - \mathbf{u}') + \mathbf{e}_0^T \cdot (\mathbf{d}_2 - \mathbf{u}') \bmod q \\ &= \mathbf{s}^T \cdot (\mathbf{p}_2 - \mathbf{p}_2) + \mathbf{e}_0^T \cdot (\mathbf{d}_2 - \mathbf{u}') = \mathbf{e}_0^T \cdot (\mathbf{d}_2 - \mathbf{u}') \bmod q. \end{aligned}$$

According to Lemma 9, the quantity $\mathbf{e}_0^T \cdot (\mathbf{d}_2 - \mathbf{u}_2)$ is bounded $\chi(\chi + 1) \cdot m$ with overwhelming probability $\geq 1 - 2^{-\lambda}$. Hence, in this case, \mathcal{B} will output 0 with overwhelming probability.

Else, if $\mathbf{c} \leftarrow \mathbb{Z}_q^n$, the value $r = \mathbf{c}^T \cdot (\mathbf{d}_2 - \mathbf{u}_2) \bmod q$ is uniform whenever $\mathbf{d}_2 - \mathbf{u}_2 \neq \mathbf{0}$. By Lemma 8 this is the case with overwhelming probability $\geq 1 - q^{-n} - 2^{-m+2}$. In this case, the probability that r lies in $\{-\chi(\chi + 1) \cdot m, \dots, \chi(\chi + 1) \cdot m\}$ is bounded by $2^{\frac{\chi(\chi+1)m}{q}} \leq \frac{2^{\chi(\chi+1)}}{\lambda \cdot \chi \cdot m} \in O(1/(\lambda m))$. Hence, \mathcal{B} will output 1 in this case with probability $1 - O(1/(\lambda m))$. It follows that the advantage of \mathcal{B} lies in $1 - O(1/(\lambda m))$. \square

6 Evasive LWE Instance in [VWW22]

We show that we can apply the PrivateHideEvLWE assumption (Definition 9) to the security proof of [VWW22]. Consequently, assuming PrivateHideEvLWE, the GGM15-encoding in [VWW22], as well as its witness encryption and null-iO constructions, remains secure.

Let $n, w, m, h \in \text{poly}(\lambda)$, let $\hat{n} = wn$, $t = 2^{j-1} \hat{n}$, and fix some $j \in [h]$. To recall, the PPT sampler used in the proof of [VWW22, Lemma 5.2], which we denote by Samp_{VWW} , outputs the following:

$$\begin{aligned} \mathbf{S} &:= \{\hat{\mathbf{S}}_{i,b}\}_{i \in [h], b \in \{0,1\}} \in \mathbb{Z}_q^{t \times \hat{n}}, \\ \mathbf{P} &:= \left(\hat{\mathbf{S}}_{j,0} \mathbf{A}_j + \mathbf{E}_{j,0}, \hat{\mathbf{S}}_{j,1} \mathbf{A}_j + \mathbf{E}_{j,1} \right) \bmod q \in \mathbb{Z}_q^{\hat{n} \times 2m}, \\ \text{aux} &:= \{\mathbf{A}_{i-1}^{-1} (\hat{\mathbf{S}}_{i,b} \mathbf{A}_i + \mathbf{E}_{i,0} \bmod q)\}_{i \geq j+1, b \in \{0,1\}}, \quad \{\hat{\mathbf{S}}_{i,b}\}_{i \in [h], b \in \{0,1\}}, \end{aligned}$$

where

- $\hat{\mathbf{S}}_{i,b} \in \mathbb{Z}_q^{\hat{n} \times \hat{n}}$ for $i \in [h], b \in \{0,1\}$ are arbitrary matrices (in the context of [VWW22] representing a branching program), and $\{\hat{\mathbf{S}}_{i,b}\}_{i \in [h], b \in \{0,1\}}$ denotes stacking the 2^h matrices vertically,
- $\mathbf{E}_{j,0}, \mathbf{E}_{j,1} \leftarrow_s (D_{\mathbb{Z}, \chi'})^{\hat{n} \times m}$ are Gaussian with parameter $\chi' \geq \lambda^{\omega(1)} \lambda^h O(n)$,
- $\mathbf{A}_i \in \mathbb{Z}_q^{\hat{n} \times m}$ for $i \geq j+1$ are uniformly random matrices,
- each $\mathbf{A}_{i-1}^{-1}(\hat{\mathbf{S}}_{i,b} \mathbf{A}_i + \mathbf{E}_{i,0} \bmod q)$ for $i \geq j+1, b \in \{0,1\}$ denotes a Gaussian preimage w.r.t. \mathbf{A}_{i-1} for the image $\hat{\mathbf{S}}_{i,b} \mathbf{A}_i + \mathbf{E}_{i,0} \bmod q$, with parameter $\chi'' = O(2\sqrt{nw \log q})$.

The proof of [VWW22, Lemma 5.2] showed that, assuming the condition of [VWW22, Equation 6], there exists no PPT \mathcal{A} distinguishing the Pre1^b experiments in Definition 9 with respect to Samp_{VWW} with non-negligible probability. In Proposition 7 below, we show that, for a large ℓ , there exists no PPT \mathcal{A} that can win Pre2 with respect to Samp_{VWW} with non-negligible probability. Invoking the $\text{PrivateHideEvLWE}_{\text{param}}$ assumption with $\text{param} = (\mathbb{Z}, q, n, m, 2m, t, \mathcal{U}(\mathbb{Z}_q^{n \times m}), (D_{\mathbb{Z}, \chi})^{t \times m}, (D_{\mathbb{Z}, \chi'})^{t \times 2m}, \ell, (\chi'')^2 \mathbf{I})$ for $\chi = \lambda^{\omega(1)} \chi'$ completes the proof of [VWW22, Lemma 5.2]. The existence of a secure witness encryption and null-iO, under the (sub-exponential) LWE and private-coin hiding evasive LWE assumptions, then follows from [VWW22, Theorem 5.1, Sections 6.7].

Proposition 7. *Let $\ell = \lambda^h$. With respect to Samp_{VWW} , there exists no PPT \mathcal{A} distinguishing Pre2^b in Definition 9 with non-negligible probability in λ .*

Proof. Note that the Gaussian parameter of $\mathbf{E}_{j,0}, \mathbf{E}_{j,1}$ is $\chi' \geq \lambda^{\omega(1)} \lambda^h O(n)$, implying $\chi' \geq \lambda^{\omega(1)} \ell$. Also, aux contains no information on $\mathbf{E}_{j,0}, \mathbf{E}_{j,1}$. Therefore, conditioned on aux , for $\mathbf{R} \leftarrow_s \mathcal{U}(\{0, 1, \dots, \ell\}^{n \times 2m})$, we have

$$\begin{aligned} \mathbf{P} + \mathbf{R} &= (\hat{\mathbf{S}}_{j,0} \mathbf{A}_j + \mathbf{E}_{j,0}, \hat{\mathbf{S}}_{j,1} \mathbf{A}_j + \mathbf{E}_{j,1}) + \mathbf{R} \bmod q \\ &\approx_s (\hat{\mathbf{S}}_{j,0} \mathbf{A}_j + \mathbf{E}_{j,0}, \hat{\mathbf{S}}_{j,1} \mathbf{A}_j + \mathbf{E}_{j,1}) \bmod q, \end{aligned}$$

where the second line is due to $(\mathbf{E}_{j,0}, \mathbf{E}_{j,1}) + \mathbf{R} \approx_s (\mathbf{E}_{j,0}, \mathbf{E}_{j,1})$ by noise flooding (Lemma 4). Therefore, $(\mathbf{P}, \text{aux}) \approx_s (\mathbf{P} + \mathbf{R} \bmod q, \text{aux})$. The claim follows. \square

Acknowledgements. The authors thank the anonymous reviewers for insightful comments which very much improved this work, in particular, sharing with us the counterexamples against a prior version of Hiding Evasive LWE, and against public-coin Evasive LWE when the sampler inputs \mathbf{B} . Chris Brzuska and Ivy K. Y. Woo are supported by Research Council of Finland grant 358950. We thank Russell W. F. Lai and Hoeteck Wee for helpful discussions.

References

- AG11. Sanjeev Arora and Rong Ge. New algorithms for learning in presence of errors. In Luca Aceto, Monika Henzinger, and Jiri Sgall, editors, *ICALP 2011, Part I*, volume 6755 of *LNCS*, pages 403–415. Springer, Berlin, Heidelberg, July 2011.

- AKY24. Shweta Agrawal, Simran Kumari, and Shota Yamada. Attribute based encryption for turing machines from lattices. In Leonid Reyzin and Douglas Stebila, editors, *CRYPTO 2024, Part III*, volume 14922 of *LNCS*, pages 352–386. Springer, Cham, August 2024.
- ARYY23. Shweta Agrawal, Mélissa Rossi, Anshu Yadav, and Shota Yamada. Constant input attribute based (and predicate) encryption from evasive and tensor LWE. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part IV*, volume 14084 of *LNCS*, pages 532–564. Springer, Cham, August 2023.
- BD20. Zvika Brakerski and Nico Döttling. Hardness of LWE on general entropic distributions. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part II*, volume 12106 of *LNCS*, pages 551–575. Springer, Cham, May 2020.
- BDE+18. Jonathan Bootle, Claire Delaplace, Thomas Espitau, Pierre-Alain Fouque, and Mehdi Tibouchi. LWE without modular reduction and improved side-channel attacks against BLISS. Cryptology ePrint Archive, Report 2018/822, 2018.
- BDK+11. Boaz Barak, Yevgeniy Dodis, Hugo Krawczyk, Olivier Pereira, Krzysztof Pietrzak, François-Xavier Standaert, and Yu Yu. Leftover hash lemma, revisited. In Phillip Rogaway, editor, *CRYPTO 2011*, volume 6841 of *LNCS*, pages 1–20. Springer, Berlin, Heidelberg, August 2011.
- CLW24. Valerio Cini, Russell W. F. Lai, and Ivy K. Y. Woo. Lattice-based multi-authority/client attribute-based encryption for circuits. *CiC*, 3, 2024. To appear.
- CM24. Yilei Chen and Xinyu Mao. Universal computational extractors from lattice assumptions. *Cryptology ePrint Archive*, 2024. <https://ia.cr/2024/225>.
- GGH15. Craig Gentry, Sergey Gorbunov, and Shai Halevi. Graph-induced multilinear maps from lattices. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015, Part II*, volume 9015 of *LNCS*, pages 498–527. Springer, Berlin, Heidelberg, March 2015.
- GPV08. Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 197–206. ACM Press, May 2008.
- HILL99. Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999.
- HLL23. Yao-Ching Hsieh, Huijia Lin, and Ji Luo. Attribute-based encryption for circuits of unbounded depth from lattices. In *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 415–434. IEEE, 2023.
- HLL24. Yao-Ching Hsieh, Huijia Lin, and Ji Luo. A general framework for lattice-based abe using evasive inner-product functional encryption. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 433–464. Springer, 2024.
- MP12. Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 700–718. Springer, Berlin, Heidelberg, April 2012.

- MPV24. Surya Mathialagan, Spencer Peters, and Vinod Vaikuntanathan. Adaptively sound zero-knowledge snarks for up. *Cryptology ePrint Archive*, 2024. <https://ia.cr/2024/227>.
- MR04. Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on Gaussian measures. In *45th FOCS*, pages 372–381. IEEE Computer Society Press, October 2004.
- NMSÜ24. Miguel Cueto Noval, Simon-Philipp Merz, Patrick Stählin, and Akin Ünal. On the soundness of algebraic attacks against code-based assumptions. 2024.
- Pei07. Chris Peikert. Limits on the hardness of lattice problems in lp norms. In *Twenty-Second Annual IEEE Conference on Computational Complexity (CCC07)*, pages 333–346, 2007.
- PR06. Chris Peikert and Alon Rosen. Efficient collision-resistant hashing from worst-case assumptions on cyclic lattices. In Shai Halevi and Tal Rabin, editors, *TCC 2006*, volume 3876 of *LNCS*, pages 145–166. Springer, Berlin, Heidelberg, March 2006.
- Reg05. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th ACM STOC*, pages 84–93. ACM Press, May 2005.
- Tsa22. Rotem Tsabary. Candidate witness encryption from lattice techniques. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part I*, volume 13507 of *LNCS*, pages 535–559. Springer, Cham, August 2022.
- VWW22. Vinod Vaikuntanathan, Hoeteck Wee, and Daniel Wichs. Witness encryption and null-IO from evasive LWE. In Shweta Agrawal and Dongdai Lin, editors, *ASIACRYPT 2022, Part I*, volume 13791 of *LNCS*, pages 195–221. Springer, Cham, December 2022.
- Wee22. Hoeteck Wee. Optimal broadcast encryption and CP-ABE from evasive lattice assumptions. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part II*, volume 13276 of *LNCS*, pages 217–241. Springer, Cham, May / June 2022.
- Wee24. Hoeteck Wee. Circuit ABE with $\text{poly}(\text{depth}, \lambda)$ -sized ciphertexts and keys from lattices. In Leonid Reyzin and Douglas Stebila, editors, *CRYPTO 2024, Part III*, volume 14922 of *LNCS*, pages 178–209. Springer, Cham, August 2024.
- WWW22. Brent Waters, Hoeteck Wee, and David J. Wu. Multi-authority ABE from lattices without random oracles. In Eike Kiltz and Vinod Vaikuntanathan, editors, *TCC 2022, Part I*, volume 13747 of *LNCS*, pages 651–679. Springer, Cham, November 2022.

Author Index

A

Agarwal, Archita 169
Albrecht, Martin R. 205
Aragon, Nicolas 68

B

Barbosa, Manuel 35
Brzuska, Chris 418

C

Chairattana-Apirom, Rutchathon 268
Chen, Jinrong 101
Chen, Rongmao 101
Cini, Valerio 238
Couvreur, Alain 68

D

Dupressoir, François 35
Dyseryn, Victor 68

F

Fenzi, Giacomo 303
Filić, Mia 137

G

Gaborit, Philippe 68
Ge, Jiangxia 3
Gur, Kamil Doruk 205

H

Huang, Xinyi 101
Hülsing, Andreas 35

J

Jiang, Kaijie 359

K

Kamara, Seny 169
Knabenhans, Christian 303
Kocher, Keran 137
Kummer, Ella 137

L

Liao, Heming 3
Ling, Cong 329
Liu, Jingbo 329
Luo, Hengyi 359

M

Meijers, Matthias 35
Mendelsohn, Andrew 329
Moataz, Tarik 169

N

Nguyen, Ngoc Khanh 303

P

Pan, Yanbin 359
Peng, Wei 101
Pham, Duc Tu 303

S

Strub, Pierre-Yves 35

T

Tessaro, Stefano 268

U

Ünal, Akin 418
Unnikrishnan, Anupama 137

V

van Woerden, Wessel 386
Vinçotte, Adrien 68

W

Wang, Anyu 359
Wang, Yi 101
Wee, Hoeteck 238
Woo, Ivy K. Y. 418

X

Xue, Rui 3

Z

Zhu, Chenzhi 268