

PROVING UPPER AND LOWER BOUNDS IN CRYPTOGRAPHY VIA ORACLES

Vom Fachbereich Informatik der
Technischen Universität Darmstadt genehmigte

Dissertation

zur Erlangung des Grades
Doctor rerum naturalium (Dr. rer. nat.)

von

Felix Rohrbach, M.Sc.



Referenten: Prof. Dr. Marc Fischlin
Prof. Dr. Chris Brzuska

Tag der Einreichung: 11. März 2025
Tag der mündlichen Prüfung: 22. April 2025

Darmstadt, 2025

Dieses Dokument wird bereitgestellt von tuprints, E-Publishing-Service der TU Darmstadt.

<http://tuprints.ulb.tu-darmstadt.de>

tuprints@ulb.tu-darmstadt.de

Bitte zitieren Sie dieses Dokument als:

Rohrbach, Felix: *Proving Upper and Lower Bounds in Cryptography via Oracles*.

Darmstadt, Technische Universität Darmstadt,

Jahr der Veröffentlichung der Dissertation auf TUprints: 2026

DOI: <https://doi.org/10.26083/tuda-7578>

URN: <https://nbn-resolving.de/urn:nbn:de:tuda-tuda-148038>

URL: <https://tuprints.ulb.tu-darmstadt.de/handle/tuda/14803>

Tag der mündlichen Prüfung: 22.04.2025

Die Veröffentlichung steht unter folgender Creative Commons Lizenz:

Attribution 4.0 International (CC BY 4.0)

<http://creativecommons.org/licenses/by/4.0/>



Erklärung

Hiermit versichere ich, die vorliegende Arbeit ohne Hilfe Dritter nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus Quellen entnommen wurden, sind als solche kenntlich gemacht. Die Arbeit wurde im Einklang mit den Grundsätzen zur Sicherung guter wissenschaftlicher Praxis an der Technischen Universität Darmstadt und Leitlinien zum Umgang mit digitalen Forschungsdaten an der TU Darmstadt verfasst. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Darmstadt, den 11. März 2025

Felix Rohrbach

Wissenschaftlicher Werdegang

Oktober 2011 – September 2015

Studium der Mathematik an der Technischen Universität Darmstadt,
Bachelor of Science.

Oktober 2015 – Oktober 2017

Studium der Mathematik an der Technischen Universität Darmstadt,
Master of Science.

Seit Mai 2018

Doktorand der Informatik an der Technischen Universität Darmstadt,
Fachgebiet Kryptographie und Komplexitätstheorie.

List of Publications

Note. As customary in the field of cryptography, most publications list authors in alphabetical order, as it is usually not possible to identify one main author. In particular, the concept of lead author does not apply. Exceptions, i.e., papers that are not ordered alphabetically, are marked with “*”.

Papers in Conferences and Workshops with Proceedings

- [1] Andreas Erwig, Marc Fischlin, Martin Hald, Dominik Helm, Robert Kiel, Florian Kübler, Michael Kümmerlin, Jakob Laenge, and Felix Rohrbach. “Redactable Graph Hashing, Revisited (Extended Abstract)”. In: *ACISP 2017 – Information Security and Privacy* Ed. by Josef Pieprzyk and Suriadi Suriadi. Vol. 10343. LNCS. Springer, Cham, May 2017, pp. 398–405. DOI: [10.1007/978-3-319-59870-3_24](https://doi.org/10.1007/978-3-319-59870-3_24).
- [2]* Dennis Heinze, Jiska Classen, and Felix Rohrbach. “MagicPairing: Apple’s take on securing bluetooth peripherals”. In: *WiSec ’20 – Proceedings of the 13th ACM Conference on Security and Privacy in Wireless and Mobile Networks*. Association for Computing Machinery, New York, NY, USA, July 2020, pp. 111–121. DOI: [10.1145/3395351.3399343](https://doi.org/10.1145/3395351.3399343). Also available at [arXiv.org Report 2005.07255](https://arxiv.org/abs/2005.07255).
- [3]* Jörn Tillmanns, Jiska Classen, Felix Rohrbach, and Matthias Hollick. “Firmware Insider: Bluetooth Randomness is Mostly Random”. In: *14th USENIX Workshop on Offensive Technologies (WOOT 20)*. USENIX Association, August 2020. Also available at [arXiv.org Report 2006.16921](https://arxiv.org/abs/2006.16921)
- [4] Marc Fischlin and Felix Rohrbach. “Single-to-Multi-theorem Transformations for Non-interactive Statistical Zero-Knowledge”. In: *PKC 2021, Part II*. Ed. by Juan Garay. Vol. 12711. LNCS. Springer, Cham, May 2021, pp. 205–234. DOI: [10.1007/978-3-030-75248-4_8](https://doi.org/10.1007/978-3-030-75248-4_8). Full version available at [Cryptology ePrint Archive Report 2020/1204](https://eprint.iacr.org/2020/1204). **Part of this thesis.**
- [5] Chris Brzuska, Geoffroy Couteau, Pihla Karanko, and Felix Rohrbach. “On Derandomizing Yao’s Weak-to-Strong OWF Construction”. In: *TCC 2021, Part II*. Ed. by Kobbi Nissim and Brent Waters. Vol. 13043. LNCS. Springer, Cham, Nov. 2021, pp. 429–456. DOI: [10.1007/978-3-030-90453-1_15](https://doi.org/10.1007/978-3-030-90453-1_15). Full version available at [Cryptology ePrint Archive Report 2023/1091](https://eprint.iacr.org/2023/1091). **Part of this thesis.**

- [6] Marc Fischlin, Felix Rohrbach, and Tobias Schmalz. “A Random Oracle for All of Us”. In: *AFRICACRYPT '22*. Ed. by Lejla Batina and Joan Daemen. Vol. 2022. LNCS. Springer, Cham, July 2022, pp. 469–489. DOI: [10.1007/978-3-031-17433-9_20](https://doi.org/10.1007/978-3-031-17433-9_20). Full version available at [Cryptology ePrint Archive Report 2022/906](https://eprint.iacr.org/2022/906). **Part of this thesis.**
- [7] Marc Fischlin and Felix Rohrbach. “Searching for ELFs in the Cryptographic Forest”. In: *TCC 2023, Part III*. Ed. by Guy N. Rothblum and Hoeteck Wee. Vol. 14371. LNCS. Springer, Cham, Nov. 2023, pp. 207–236. DOI: [10.1007/978-3-031-48621-0_8](https://doi.org/10.1007/978-3-031-48621-0_8). Full version available at [Cryptology ePrint Archive Report 2023/1403](https://eprint.iacr.org/2023/1403). **Part of this thesis.**
- [8]* Jiska Classen, Alexander Heinrich, Fabian Portner, Felix Rohrbach, and Matthias Hollick. “Starshields for iOS: Navigating the Security Cosmos in Satellite Communication”. In: *Network and Distributed System Security (NDSS) Symposium 2025*. February 2025. DOI: [10.14722/ndss.2025.240124](https://doi.org/10.14722/ndss.2025.240124).

Articles in Submission

- [9] Chris Brzuska, Christoph Egger, Pihla Karanko, and Felix Rohrbach. “On the Difference between Distributional and Existential Collision-Resistance”. 2025. **Part of this thesis.**

Acknowledgements

A PhD is an extraordinary journey, and one that would not have been possible without the support and encouragement I received from everyone around me – and who made the journey all the more enjoyable!

First and foremost, I am deeply grateful to Marc Fischlin, who not only let me pursue my PhD, but supported me extensively throughout and even before. Marc introduced me to the world of cryptography, and I was immediately hooked by his puzzle-style of teaching the topic. When the university didn't offer any advanced lectures in this area, he invited me to his reading group, in which I got to know the Cryptoplexity group and already felt part of it when I, the “Little Felix”, finally joined as a PhD student.

During my PhD, Marc gave me a lot of freedom to choose the research projects we worked on. I enjoyed our regular deep technical discussions; his optimism and his good humor, even in situations where we were stuck, got me through the first few rejected papers. Thank you for the trust you put into me, and for being an excellent mentor!

Being a part of the Cryptoplexity group in Darmstadt was a wonderful time! At the start of a PhD, there is so much to learn, and you were always eager to support me with whatever issue arose. Special thanks to Andrea Püchner, our team assistant, who always made sure all the bureaucratic tasks are taken care of, who organized so many wonderful events, and who had an open ear for every problem. Patrick Harasser shared my first office with me and introduced me to Goldreich's Foundations of Cryptography book and the importance for caring about the little details. Thank you also for surviving two rounds of TAing for AuD together with me, and teaching me so much about L^AT_EX! Furthermore, thank you to all who have been part of the (extended) group in my time in Darmstadt: Sogol Mazaheri, Christian Janson, Felix Günther, Jacqueline Brendel, Giorgia Azzurra Marson, Patrick Struck, Stefanie Kettler, Juliane Krämer, Jean Paul Degabriele, Johannes Braun, Shan Chen, Aishwarya Thiruvengadam, Olga Sanina, Samed Düzlü, Jérôme Govinden, Vukašin Karadžić, Daniela Fleckenstein, Rune Fiedler, Tobias Schmalz, Moritz Hupper, Elif Özbay Gürler, Zoé Vignes, Evangelos Gkoumas, Julius Hardt, Leonhard Ruppel, and Jenit Tomy. Thank you also to the CROSSING project, which supported me in many ways during my PhD, and all the people I got to know there!

I am also very grateful to Chris Brzuska, who I've met early in my PhD and who invited me to visit him at Aalto – a visit that should only be the first one of five in the following years. Working with Chris has been one of the most delightful times during my PhD – his sheer joy and dedication for research makes an hour-long session discussing probabilities and entropy notions, of course with an infinite supply of tea, a truly enjoyable experience. Thank you also to Pihla Karanko, who has been a part of most of these discussions, it was so much fun working with you! And also beside research, the time at Aalto was always wonderful, with cooking, board games, Finnish sauna, hikes

through the snow and even a weekend visit to Chris' mökki. I felt like I was an honorary member of your group as well, and sometimes during the months of lock-down, Aalto really felt closer than Darmstadt!

Thank you also, Chris, for agreeing to be the second referee for my thesis, and thank you to Thomas Schneider and Sebastian Faust for taking the time to serve on my defense committee.

Research has always been a collaborative experience for me, and without my co-authors, this thesis would probably not exist. Thank you to Christoph Egger, with whom I had many deep discussions about research and the life as researcher. I really enjoyed visiting you (and Geoffroy) in Paris, and going to Île d'Oléron together for a summer school! Thank you to Geoffroy Couteau and Iftach Haitner for discussing the nuances of one-way functions with me, and thank you Tobias for joining me in going to Morocco to present our joint paper there. Finally, thank you Jiska Classen for giving me the opportunity to work on something completely different and look at implementations of Bluetooth and Satellite protocols!

Last but not least, I am thankful to all my friends and family for supporting me and giving me the opportunity to tune out of my PhD from time to time. Thank you, Anne and Matthias, for always backing me to the fullest in my choices! Thank you, Eva, for all the love and support, and for bearing with me during the last months of writing! Thank you to my two choirs and all the other musical projects, which were immense amounts of fun and a perfect counter balance to all the math. And of course thanks to all the other friends with whom I had a wonderful time in Darmstadt!

Felix Rohrbach
Saarbrücken, November 2025

Abstract

Provable security is a cornerstone of modern cryptography: Due to ubiquitous and diverse applications of cryptography, a proof of security gives us the necessary confidence to deploy a cryptographic protocol. In most cases, such a security proof comes in the form of a *black-box reduction*, which bases the security of a potentially complex protocol on a small set of simple and abstract assumptions that are much easier to analyse.

However, proving a black-box reduction can be quite complicated, and we do not have proofs for every protocol used in practice. Here, analysing the protocols relative to *oracles*, a technique from computational complexity theory, can provide insights: Oracles provide the ability to compute functionalities in one computational step that otherwise might not be efficiently computable, e.g., provide access to a truly random function or solve any NP-complete problem. These oracles now allow us to replace some parts in the protocol with abstract, idealized primitives that are easier to analyse, e.g., to replace a one-way function with a truly random function. In this thesis, we utilize oracles in two different ways.

In the first part, we use oracles to prove *lower bounds* for cryptographic primitives, i.e., showing that certain assumptions are not sufficient to build this primitive securely. The essential idea here, going back to Impagliazzo and Rudich, is to replace the assumption with an oracle, i.e., replacing a one-way function with a truly random function, and then showing that relative to this oracle, it is impossible to build the primitive. From this impossibility result relative to the oracle, we can now conclude that the primitive cannot be built from the assumption in a black-box way. We use this technique to prove a lower bound on the efficiency of constructing strong from weak one-way functions, to show that we cannot construct collision-resistant hash functions from distributional collision-resistant hash functions in a fully black-box way, and to prove that extremely lossy functions cannot be built from a large class of symmetric primitives in a black-box way.

In the second part of this thesis, we use oracles as idealized models that can be used to provide heuristic security arguments for protocols. These idealized models, starting with the random oracle model (short ROM) introduced and defined by Fiat and Shamir as well as Bellare and Rogaway, were motivated by the existence of very efficient cryptographic protocols used in practice, but for which no proof of security existed. Using idealized models, it was now possible to give at least a heuristic security argument for them. In this thesis, we first focus on the common random string model, an idealized model introduced to circumvent impossibility results for non-interactive zero-knowledge proofs. We show how to reuse a single common random string for polynomially many non-interactive *statistical* zero-knowledge arguments, as well as analyze the relation between different soundness definitions used in literature. In a second result, we introduce an alternative notion for the ROM, the universal random oracle model, which brings this idealized model closer to reality.

Zusammenfassung

Beweisbare Sicherheit ist ein Grundpfeiler moderner Kryptographie: Wegen der allgegenwärtigen und vielseitigen Verwendung von Kryptographie ist ein Sicherheitsbeweis essentiell, um das notwendige Vertrauen in ein kryptografisches Protokoll herzustellen. In den meisten Fällen hat ein solcher Beweis die Form einer *Black-Box-Reduktion*, in der die Sicherheit des potenziell komplexen Verfahrens auf eine kleine Menge von einfacheren Annahmen reduziert wird, welche dann ihrerseits besser zu untersuchen sind.

Eine solche Reduktion zu beweisen kann jedoch kompliziert sein, und nicht für jedes Protokoll, welches in der Praxis eingesetzt wird, kennen wir einen Beweis. Daher kann es aufschlussreich sein, die Protokolle relativ zu einem *Orakel* zu analysieren: Ein Orakel erlaubt es, bestimmte Funktionalitäten in einem logischen Schritt zu berechnen, die ansonsten potenziell nicht effizient berechenbar wären, wie zum Beispiel der Zugriff auf eine echt zufällige Funktion oder die Lösung eines jeden NP-Problems. Diese Orakel ermöglichen es, bestimmte Teile des Protokolls mit abstrakten, idealisierten Primitiven zu ersetzen, die einfacher zu analysieren sind, beispielsweise könnte man eine One-Way-Funktion mit einer echt zufälligen Funktion ersetzen. In dieser Thesis verwenden wir Orakel auf zwei verschiedene Weisen:

In ersten Teil der Thesis verwenden wir Orakel, um *untere Schranken* für die Konstruktion von kryptografischen Primitiven zu beweisen, das heißt, wir zeigen, dass bestimmte Annahmen nicht ausreichend sind, um dieses Primitiv sicher zu konstruieren. Die grundlegende Idee hierfür geht auf Impagliazzo und Rudich zurück, die die Annahme durch ein Orakel ersetzt haben, also beispielsweise eine One-Way-Funktion durch eine echt zufällige Funktion, um dann im nächsten Schritt zu zeigen, dass es unmöglich ist, das kryptografische Primitiv relativ zu dem Orakel zu bauen. Basierend auf dieser unteren Schranke relativ zum Orakel kann man nun schließen, dass es keine Black-Box-Konstruktion des Primitives von den ersetzten Annahmen geben kann. Wir verwenden diese Technik in drei Resultaten: Wir zeigen eine untere Schranke für die Effizienz von Konstruktionen von starken One-Way-Funktionen basierend auf schwachen One-Way-Funktionen; wir beweisen, dass es keine Fully-Black-Box-Konstruktionen von Collision-Resistant Hash Functions basierend auf Distributional Collision-Resistant Hash Functions geben kann; und wir zeigen, dass es keine Black-Box-Konstruktion von Lossy Functions basierend auf einer großen Menge von symmetrischen Primitiven geben kann.

Im zweiten Teil der Thesis nutzen wir Orakel als idealisierte Modelle, welche verwendet werden, um heuristische Sicherheitsargumente für kryptografische Protokolle zu erstellen. Beginnend mit dem Random Oracle Model, eingeführt von Fiat und Shamir sowie Bellare und Rogaway, wurden diese idealisierten Modelle durch die Existenz von Protokollen motiviert, welche effizient waren und in der Praxis eingesetzt wurden, für die jedoch kein Sicherheitsbeweis existierte. Mit Hilfe der Modelle war es jetzt möglich, zumindest ein heuristisches Sicherheitsargument für diese Protokolle zu erstellen. In dieser Thesis betrachten wir zunächst das Common Random String Model, ein Modell,

welches das Umgehen von Unmöglichkeitsresultaten bei nicht-interaktiven Zero-Knowledge-Beweisen ermöglicht. Weiterhin untersuchen wir in diesem Resultat verschiedene Korrektheitsdefinitionen, die in der Literatur existieren. In einem zweiten Resultat führen wir eine Alternative zum Random Oracle Model ein, das Universal Random Oracle Model, welches im Vergleich die Realität besser abbildet.

Short Contents

List of Publications	v
Acknowledgements	vii
Abstract	ix
Zusammenfassung	xi
Short Contents	xiii
Contents	xv
Part I — Synopsis	1
1 Introduction	3
2 Preliminaries	9
3 Proving Lower Bounds using Oracle Separations	13
4 Using Idealized Models for Constructions	27
5 Conclusion & Outlook	37
Bibliography	39
Part II — Publications	47
1 On Derandomizing Yao’s Weak-to-Strong OWF Construction	49
2 On the Difference between Distributional and Existential Collision-Resistance	77
3 Searching for ELFs in the Cryptographic Forest	129
4 Single-to-Multi-Theorem Transformations for Non-Interactive Statistical Zero-Knowledge	169
5 A Random Oracle for All of Us	199

Contents

List of Publications	v
Acknowledgements	vii
Abstract	ix
Zusammenfassung	xi
Short Contents	xiii
Contents	xv
Part I — Synopsis	1
1 Introduction	3
1.1 Oracles in Complexity Theory and Cryptography	3
1.2 Proving Lower Bounds with Oracles	5
1.3 Idealized Models	5
1.4 Contributions of This Thesis	6
1.5 A Note on Authorship and Versions	8
2 Preliminaries	9
2.1 Languages, Language Classes and Oracle Machines	9
2.2 Black-Box Reductions	10
3 Proving Lower Bounds using Oracle Separations	13
3.1 Different Types of Oracle Separations	13
3.1.1 Impagliazzo–Rudich-style separations	13
3.1.2 Simon-style separations	14
3.2 Limits of Oracle Separations	15
3.3 <i>On Derandomizing Yao’s Weak-to-Strong OWF Construction</i>	16
3.3.1 Related Work	17
3.3.2 Proof Overview	17
3.3.3 Limitations of our result	18
3.3.4 My Contributions	18

3.3.5	Versions	18
3.4	<i>On the Difference between Distributional and Existential Collision-Resistance</i>	19
3.4.1	Oracles	20
3.4.2	Restrictions	20
3.4.3	F is a dCRH	20
3.4.4	No existential CRH	21
3.4.5	My Contributions	21
3.4.6	Versions	21
3.5	How to Replace the Random Oracle Model as an Assumption	22
3.6	<i>Searching for ELF's in the Cryptographic Forest</i>	22
3.6.1	Oracle Separations	23
3.6.2	A Note on Random Oracles and the Random Oracle Model	24
3.6.3	My Contributions	24
3.6.4	Versions	24
3.7	Contributions of This Chapter	25
4	Using Idealized Models for Constructions	27
4.1	Non-Interactive Zero-Knowledge and the Common Reference String Model	27
4.2	<i>Single-to-Multi-Theorem Transformations for Non-Interactive Statistical Zero-Knowledge</i>	29
4.2.1	Multi-Theorem from One-Way Permutations	30
4.2.2	Multi-Theorem from LWE	30
4.2.3	Soundness of Non-Interactive Zero-Knowledge Arguments	31
4.2.4	My Contributions	32
4.2.5	Versions	32
4.3	Random Oracles and Variations	32
4.4	<i>A Random Oracle for All of Us</i>	33
4.4.1	Defining the UROM	33
4.4.2	UROM \Leftrightarrow AI-ROM	34
4.4.3	One-way functions exist in UROM	34
4.4.4	My Contributions	35
4.4.5	Versions	35
4.5	Contributions of This Chapter	35
5	Conclusion & Outlook	37
	Bibliography	39
	Part II — Publications	47
1	On Derandomizing Yao's Weak-to-Strong OWF Construction	49
1	Introduction	49
1.1	On Security-Preserving Amplification of Weak OWFs	51
1.2	Our Contribution	51
1.3	Relation to Correlated-Product and Correlated-Input Security	52
1.4	Related Work	53

1.5	Technical Overview	54
2	Preliminaries	56
3	Main Results	58
3.1	Black-box constructions and reductions	58
3.2	Theorems	59
4	Oracle Distributions	61
5	Proof of Theorem 3.5	61
5.1	R is not a successful weak OWF inverter	62
5.2	A is a successful strong OWF inverter	62
6	Constructions with post-processing	67
	References	68
A	Additional Lemmas and Proofs	72
B	Proof of Theorem 5.1 (F is a weak OWF)	73
2	On the Difference between Distributional and Existential Collision-Resistance	77
1	Introduction	77
1.1	Distributional collision-resistance	78
1.2	Multi-collision resistance	79
1.3	Relations between different notions of collision resistance	79
1.4	Our contribution	80
2	Technical overview	82
2.1	F is a dCRH	83
2.2	No CRH relative to F, ColFind	84
3	Preliminaries	88
4	Main Theorem (dCRH $\not\Rightarrow$ CRH)	89
5	No CRH relative to F, ColFind	91
5.1	Using Chebychev	93
5.2	Bounding Co-Variance (Lemma 5.3)	95
5.3	Proof of Lemma 5.4 (Intersections of F-Collisions)	99
5.4	Proof of Lemma 5.1 (Ground Truth)	102
5.5	Proof of Lemma 5.5	106
6	F is a dCRH	108
6.1	The Compression and Decompression Algorithms	110
6.2	Correctness of Compression	111
6.3	Compressed Encoding	112
6.4	Adversary Finds Many Hard Collisions	114
6.5	Proof of Lemma 4.3	117
	References	118
A	Proof of Theorem 4.2 (Main)	121
B	negl-dCRH $\not\Rightarrow$ OWF	123
B.1	Oracle distribution	124
B.2	Proof of Lemma B.3 (Exists Negl-dCRH)	125
B.3	Proof of Lemma B.2 (No OWF)	127
C	Unsalted and Salted dCRH Definitions	127

D	Counterexample: F-queries first	128
3	Searching for ELFs in the Cryptographic Forest	129
1	Introduction	129
1.1	Our Contributions	130
1.2	Our Techniques	132
1.3	Related Work	132
2	Technical Overview	133
2.1	No (E)LFs in Oraclecrypt	133
2.2	No Key Agreement from (E)LFs	135
3	Preliminaries	136
3.1	Lossy Functions	137
3.2	Notions of Black-box Constructions and Oracle Separations	138
3.3	Oraclecrypt	139
4	On the Impossibility of Building (E)LFs in Oraclecrypt	139
4.1	Introducing the Oracles	140
4.2	Approximating the Set of Heavy Queries	141
4.3	Distinguishing Lossiness from Injectivity	143
4.4	Fixing an Oracle	145
5	On the Impossibility of Building Key Agreement Protocols from ELFs	147
5.1	Lossy Function Oracle	147
5.2	Key Exchange	152
5.3	ELFs	155
6	Relationship of Lossy Functions to Statistical Zero-Knowledge	159
	References	162
A	Lazy-Sampling a Random Permutation	165
B	Lossy Functions Imply Hard Statistical Zero-Knowledge Problems	167
C	Deferred Proofs	167
C.1	Proof for Lemma 3.7	167
4	Single-to-Multi-Theorem Transformations for Non-Interactive Statistical Zero-Knowledge	169
1	Introduction	169
1.1	Flavors of Non-Interactive Zero-Knowledge	170
1.2	From Single-Theorem to Multi-Theorem Proofs	170
1.3	Known NISZK Constructions	171
1.4	Our Results	173
1.5	Squeezing in into Possibility and Impossibility Results	174
1.6	Concurrent Work	174
2	Preliminaries	175
2.1	Non-Interactive Arguments	175
2.2	Zero-Knowledge	176
2.3	From Single-Theorem Zero-Knowledge to Multi-Theorem Witness Indistinguishability	178
3	Soundness of Non-Interactive Arguments	179

3.1	Soundness Definitions	179
3.2	Equivalence of the Non-Adaptive Soundness Notions	182
3.3	Exclusive Soundness Implies Culpable Soundness	183
4	Constructions based on General Assumptions	183
4.1	Multi-theorem NISZK based on One-way Permutations	183
4.2	Adaptive Perfect Zero-Knowledge under Expected Polynomial Time	187
5	A Lattice-Based Construction	188
5.1	Dual-Mode Commitment Schemes based on Lattices	188
5.2	SZK-FLS-Transformation based on Lattices	191
6	Conclusion	192
	References	193
A	Multi-theorem NIPZK in the Common Reference String Model	196
5	A Random Oracle for All of Us	199
1	Introduction	199
1.1	The Universal Random Oracle Model	200
1.2	Defining Universal Random Oracles	200
1.3	Relationship to Auxiliary-Input Random Oracles	201
1.4	Relationship to Global Random Oracles	201
1.5	Proving Security in the UROM	202
2	Preliminaries	202
2.1	Negligible Functions	202
2.2	Security Games	203
2.3	The Random Oracle Model	204
2.4	One-Wayness in the Random Oracle Model	204
3	The Universal Random Oracle Model UROM	204
4	UROM vs. AI-ROM	205
4.1	AI-ROM	205
4.2	AI-ROM implies UROM	206
4.3	UROM implies AI-ROM	207
4.4	Advantages of UROM	208
5	Universal Random Oracles are One-way Functions	208
6	Conclusion	213
	References	213
A	Defining Universal Random Oracles	215

PART I

Synopsis

1 Introduction

Cryptography is the science of defining, constructing and understanding computing systems, which feature security properties against potentially powerful adversaries — such systems, for example, include secure communication between two parties over an untrusted channel, validating the origin of a text, or dividing a secret between multiple parties. Implementations of these systems are omnipresent in today’s electronic devices, such as smartphones, printers, Internet-of-Things devices and many more.

Due to the ubiquitous and diverse applications of cryptography, it is essential to get the security right. Indeed, as the exact nature in which the cryptographic primitive will be used as well as the attack strategy of the adversary is unknown at the time of design, a thorough security analysis of the primitive is necessary, and more ad-hoc implementations have often lead to attacks in the real world (e.g. [Ble98], [MDK14]). The best way to do this is to *prove* the primitive to be secure, usually by constructing the primitive based on simpler, easier-to-analyse primitives, and prove the security of the new primitive assuming the security of the underlying primitive. For such a construction, we normally do not care how exactly the underlying primitives are implemented — in these cases we call the construction a *black-box* construction, and the proof is called a *black-box reduction* (see also Section 2.2).

Reductions are essential to cryptography, as for most cryptographic primitives, it is impossible today to provide an unconditional proof: Should the P vs. NP question, one of the biggest unsolved questions in mathematics, be answered with $P = NP$, most of cryptography would be insecure. While the general consensus between mathematicians is that this is unlikely [Gas19], cryptography depends on even more assumptions: In what is known as his five worlds, Impagliazzo [Imp95] describes five different “worlds” where different assumptions hold regarding the worst-case and average-case hardness: The first three don’t contain (complexity-based) cryptography, the fourth world is one that allows for one-way functions, commonly seen as the minimal cryptographic primitive, to exist, and the fifth one allows for public-key cryptography to exist. As we don’t know which of these worlds we live in, cryptographic primitives usually have reduction proofs all the way down to either some mathematical assumption, e.g. the Diffie-Hellman assumption, or some concrete implementation of an easy-to-understand primitive that has been analyzed extensively by the cryptographic community, e.g. SHA3 as hash function.

1.1 Oracles in Complexity Theory and Cryptography

Proofs about what an efficient algorithm can’t compute are notoriously elusive. Therefore, it can be helpful to assume the existence of idealized versions of some functionality the algorithm (either

for an implementation or an adversary) might want to implement — for example, we could have an idealized functionality computing a truly random function, which can be seen as an idealized, unconditionally secure version of a one-way function. We call these idealized functionalities *oracles*, and the algorithm would then get access to the oracle, which computes the abstract functionality in one computational step. Contrary to the original algorithm, oracles are not restricted to be efficient algorithms, they can run in exponential time or use exponential space, or even solve non-computable problems like the halting problem. If we introduce an oracle, we therefore say that we are in an *oracle world* or an *idealized model*, while the world without these oracles is called the *standard model*.

The origin of oracles lies in the area of computational complexity. Here, a common task is to prove that two language classes C_1 and C_2 are either equal or unequal. Proving such a result in an oracle world, i.e., whether C_1^O equals C_2^O or not, might be a task that is much easier and already provides a hint for which direction to prove in the standard model. Indeed, nearly all known proof techniques in computational complexity *relativize*, meaning that they still hold if we move from the standard model to some oracle world. Therefore, if one shows that $C_1^O = C_2^O$, we know that we won't find a relativizing proof for $C_1 \neq C_2$.

Baker, Gill and Solovay [BGS75] were the first to use oracle worlds to show that a result cannot be proven by relativizing proof techniques: They show that two oracles A and B exist such that $P^A = NP^A$ and $P^B \neq NP^B$. Therefore, no relativizing proof can be found for either $P = NP$ or $P \neq NP$, and new proof techniques would be needed to prove such a result.

The work of Baker, Gill and Solovay has inspired a lot of other results which show that an unsolved problem cannot be proven using a relativizing technique: for example, Watrous [Wat00] shows that relative to an oracle, BQP is not contained in NP, and Bennett and Gill [BG81] show that, while there exists an oracle relative to which BPP equals P, relative to a random oracle, BPP does not equal P with probability 1. Indeed, showing that no relativizing proof can solve a problem is the main way of showing that a problem in computational complexity is hard and probably out of reach with today's techniques. However, it should not be seen as a marker of an impossible problem, as the story of IP (the class of all languages decidable by an interactive proof) vs. PSPACE (the class of all languages decidable by a Turing machine using a polynomial amount of space) shows: It was known that there exists oracles A and B , such that $IP^A = PSPACE^A$, but $IP^B \neq PSPACE^B$ [FS88], i.e., relativizing techniques are not enough to solve this problem. In 1992, however, Shamir [Sha92] showed $IP = PSPACE$ using a new, non-relativizing technique which is called arithmetization (which, however, is still not powerful enough to decide P vs. NP [AW08]).

In cryptography, two major applications of oracles exist: The first one are *oracle separations*, in which we show that relative to some oracles, one cryptographic primitive P exists, but another primitive Q does not. This result in an oracle world gives us a *lower bound* on the assumptions needed to build Q in the standard model: There exists no *black-box* construction of Q from P , i.e., P is not a strong enough assumption to build Q .

The second application are heuristic security arguments for protocols by using idealized models: Here, we idealize some primitive in a protocol by an oracle (e.g., a hash function by a random oracle), prove the protocol secure in this *idealized model*, and then assume that security for the protocol still holds if we replace the idealized model with a concrete implementation of the primitive. While this heuristic does not yield a security proof, it allows to explain the security of protocols for which no proofs exist.

1.2 Proving Lower Bounds with Oracles

In their seminal work, Impagliazzo and Rudich [IR89] used oracles to show that it is hard to build key agreement from one-way functions: They start with assuming that $P = NP$, but also allow for a random oracle, an oracle representing a truly random function, to exist. Indeed, a random oracle can be seen as an idealized one-way function (which even stays a one-way function if $P = NP$). Next, they show using $P = NP$ that no key agreement can exist, even when utilizing the random oracle. Now, they claim, this means that any generic construction of key agreement from a one-way function must contain an implicit proof that $P \neq NP$.

Another way to look at this result is as an *oracle separation*: In an oracle world with a PSPACE-oracle (i.e., an oracle that can decide any language in PSPACE, including every language in NP) and a random oracle, one-way functions exist, but key agreement does not. Therefore, in the standard model, no *relativizing* proof can exist for a (generic) construction of key agreement from one-way functions. The main technique to construct a cryptographic primitive from an assumption are black-box constructions, which indeed are relativizing — so an oracle separation can be used to rule out the existence of black-box constructions.

As nearly all constructions in cryptography are black-box constructions, oracle separations developed into the main tool to show cryptographic lower bounds, a few important examples being Rudich [Rud88] separating one-way permutations from one-way functions, Simon [Sim98] separating collision-resistant hash functions from one-way function, Gertner et al. [GKMRV00] showed that public-key encryption and oblivious transfer are incomparable (i.e., none can be built from the other), and Gennaro and Trevisan [GT00] show limits to the efficiency of building PRGs from one-way functions.

1.3 Idealized Models

The random oracle model (ROM) first appeared in a work by Fiat and Shamir [FS87] and was later formally introduced by Bellare and Rogaway [BR93] as a paradigm to give heuristic security arguments for highly efficient protocols. They argue that proving a protocol in the standard model often impacts the efficiency of it, while using the random oracle paradigm allows for much more efficient protocols, while still retaining a solid security argument. Further, their paradigm explains previous cases of “unjustified” uses of hash functions, i.e., cases in which the hash-function is used as random-looking function, which however can’t be explained with the security notion of collision resistance. They provide proofs in the random oracle model for constructions of encryption, signatures and zero-knowledge protocols.

While Bellare and Rogaway already note that this security heuristic fails for contrived protocols if the hash function is not chosen “independently” from the protocol, Canetti, Goldreich and Halevi [CGH98] showed that there exists a protocol for which the heuristic fails for any instantiation of the hash function. However, also this example is quite contrived – the protocol accepts any signature if the internal hash function is efficiently implemented, which is not the case in the random oracle model, but true for any implementation in the standard model. Similar counterexamples have been developed for other use cases of the random oracle, e.g., Barak [Bar01] and Goldwasser and Kalai [GK03] gave an example of an interactive protocol, which, when the Fiat-Shamir transform is applied to it, turns into an insecure non-interactive protocol. Very recently, Khovratovich, Rothblum

and Soukhanov [KRS25] showed arguably the first non-contrived counter example for the random oracle model, which is however quite specific to a proof protocol using the Fiat-Shamir heuristic, and depends on being able to control the statement in the proof.

Another important idealized model is the generic group model by Shoup [Sho97]¹, which gives access to an abstract random group, allowing to analyse discrete logarithm-based cryptography (e.g., Diffie-Hellman key exchange) without having to take into account the concrete underlying group. Other idealized models allow to circumvent impossibility results: The Common Reference String model [BFM90] gives all parties access to a shared string sampled honestly from some specific distribution (in the case of [BFM90], this is the uniform distribution), allowing for the existence of non-interactive zero-knowledge protocols, which cannot exist in the standard model. The Common Reference String model is quite different to the previous idealized models, as it can be instantiated efficiently, either by a trusted (third) party, a common source of randomness or a (pre-computed) interactive protocol. Nevertheless, the model is still useful as it allows to analyse the non-interactive protocol in isolation.

As proofs in idealized models like the random oracle model only provide a heuristic security argument, different approaches have been developed to bring the proofs closer to the standard model. One line of work replaces the random oracle in proofs with new assumptions that might hold in the standard model: For random oracles, these include the notions of correlation intractability [CGH98], universal computational extractors (UCEs) [BHK13] and extremely lossy functions (ELFs) [Zha16]. Another line of work tries to restrict the use of the random oracle to better match how we would use a hash function in the standard model, e.g., by making the random oracle non-programmable [Nie02; FLRSST10] or non-observable [AB13].

1.4 Contributions of This Thesis

This thesis is divided into two parts: In the first one, I will show multiple lower bounds using oracle separations, and in the second one, I will build constructions based on an idealized model, as well as present a new variant of the random oracle model.

For our first oracle separation, we look at one-way functions: Next to the standard definitions of (strong) one-way functions (OWFs), there are also *weak* one-way functions, which can be inverted with at most polynomial probability (instead of negligible probability for strong OWFs). Yao [Yao82] showed that one can build strong one-way functions from weak ones, using a direct product approach where the input is split and the one-way function is called on each part. Unfortunately, this construction has a polynomial security loss. In our paper **On Derandomizing Yao’s Weak-to-Strong OWF Construction** (with Chris Brzuska, Geoffroy Couteau and Pihla Karanko, published at TCC 2021), we show that this security loss is essentially inevitable for non-adaptive constructions: We construct an oracle separation, showing that no fully black-box non-adaptive construction of strong one-way functions from weak ones exist as long as the output of the weak one-way functions is not compressed too much.

The second oracle separation in my thesis is in regard to two different notions of collision resistance: For (existential) collision resistance, an adversary has to just find some collision to be successful, while for distributional collision resistance, it has to output a *random* collision. Contrary to similar

¹similar models were proposed by Nechaev [Nec94] and Maurer [Mau05]

definitions for one-way functions, there exists no construction of existential collision resistance from distributional collision resistance. In our paper **On the Difference between Distributional and Existential Collision Resistance** (with Chris Brzuska, Christoph Egger and Pihla Karanko, in submission), we show that indeed such a construction is unlikely to exist, as we show that no fully black-box construction can exist, for a large class of reductions.

The third contribution of my thesis combines both aspects of idealized models and oracle separations. In **Searching for ELF's in the Cryptographic Forest** (with Marc Fischlin, published at TCC 2023), we analyse extremely lossy functions, or short ELF's, and the minimal assumptions to build them. ELF's, introduced by Mark Zhandry [Zha16], are a primitive used to replace random oracle assumptions, i.e., moving results from the random oracle model to the standard model. There exists a construction based on the exponential security of elliptic curves, but no constructions from other assumptions. Indeed, one open question was whether a construction from a symmetric primitive could exist. In our paper, we rule out any black-box construction of ELF's from a large number of symmetric primitives, including exponentially secure one-way functions or collision-resistant hash functions. However, we also affirm the intuition that ELF's are not inherently a public-key primitive by showing an oracle separation between ELF's and key agreement.

As first contribution in my second chapter, I present the paper **Single-to-Multi-Theorem Transformation for Non-Interactive Statistical Zero-Knowledge** (with Marc Fischlin, published at PKC 2021), which is a construction relative to an idealized model. Non-interactive zero-knowledge proofs are protocols in which a prover convinces a verifier of some statement, without revealing anything more than the truthfulness of the statement itself. As the protocol is supposed to be non-interactive, it consists of just one message from the prover to the verifier, and no more interaction. Zero-knowledge proofs require interaction in the standard model, however, therefore, results about non-interactive zero-knowledge are proven in the common random string model or the common reference string model [BFM90] — an honestly generated string, either from the uniform distribution (in the “random” case) or some other distribution (in the “reference” case).

Unfortunately, in general, a fresh common random/reference is required for each proven statement. Feige, Lapidot and Shamir [FLS90] showed a generic transformation to prove polynomially many statements with just one common random string for non-interactive *computational* zero-knowledge proofs. A folklore version for *statistical* zero-knowledge exists, but only works in the common reference string model. We present two new transformations for non-interactive statistical zero-knowledge in the common random string model, one construction based on one-way permutations and one construction based on LWE. As a second contribution, we identify different types of soundness in literature, define five notions and analyse their relations.

In my last contribution, the paper **A Random Oracle for All of Us** (with Marc Fischlin and Tobias Schmalz, published at Africacrypt 2022), we construct an alternative to the random oracle model, which we call the universal random oracle model (short UROM). The motivation for the UROM is that the classic random oracle model is defined in such a way that a new random oracle is drawn for every adversary. However, a hash function, which the ROM idealizes, is identical for every adversary. We therefore define the UROM in such a way that it is randomly chosen, but identical for every adversary. We show that the UROM is asymptotically equivalent to the auxiliary-input random oracle model (AI-ROM) [Unr07] – however, while security in the UROM implies security in the AIROM tightly, the other direction introduces a security loss. We also show that in the UROM, one-way function exist, validating our model.

1.5 A Note on Authorship and Versions

Researching has been a very collaborative experience for me. I have written all papers presented in this thesis with co-authors, and for all papers, the research contained in them originated from countless hours of meetings, in which we discussed ideas, definitions, proof ideas, found counterexamples to our definitions and proof ideas, only to refine them afterwards. This way of conducting research has been very productive (and, indeed, very enjoyable) for me, and it leads to publications where every part is owed to all co-authors collaboratively. However, this thesis is of course about my contribution, and I will therefore highlight for every paper where I see my main contributions (e.g., where I came up with the right definition, or where I took the lead in formulating the proof etc.). I want to emphasize this should not diminish the contributions of my co-authors in any way.

Publishing a paper is unfortunately subject to strict page limits, which in practice means many published versions of papers are incomplete, as even things that have been peer-reviewed in the publishing process have to be cut out in the end. Further, a published paper can normally not easily be updated, in case there is a small mistake in the paper or the presentation of a result could be improved. This is why most researchers in the cryptographic community also publish a full version of their paper on the ePrint server of the IACR². As I view the full version of my papers on ePrint as the most comprehensive and up-to-date version, I will use these full versions of the papers in this thesis. If there are changes between the peer-reviewed version and the full version, I will describe these for each paper.

²<https://eprint.iacr.org>

2 Preliminaries

2.1 Languages, Language Classes and Oracle Machines

Languages are a way to encode problems in complexity theory. A language L is a subset of all possible strings: $L \subset \{0, 1\}^*$. A language L is decided by a Turing machine A , if for all strings x , $A(x)$ outputs 1 if and only if $x \in L$. We say that a language L is efficiently decidable if there exists a Turing machine A that runs in polynomial time in the length of its input x and which decides L . We define a few language classes that will be used in this thesis:

Definition 2.1.1 (P). The class P contains all the languages that can be decided efficiently: $L \in P$ if there exists a polynomial-time Turing machine A such that for every string $x \in \{0, 1\}^*$:

$$x \in L \Leftrightarrow A(x) = 1$$

Definition 2.1.2 (NP). The class NP contains all the languages that can be verified efficiently if given a witness w : $L \in NP$ iff there exists a polynomial-time Turing machine A and a polynomial $p(n)$ such that for all $x \in \{0, 1\}^*$:

$$x \in L \Leftrightarrow \exists w \in \{0, 1\}^{\leq p(|x|)} : A(x, w) = 1$$

Definition 2.1.3 (BPP). The class BPP contains all the languages that can be decided efficiently by a probabilistic Turing machine with high probability: $L \in BPP$ iff there exists a probabilistic polynomial-time Turing machine A such that

- (a) $\forall x \in L : \Pr[A(x) = 1] \geq \frac{2}{3}$
- (b) $\forall x \notin L : \Pr[A(x) = 1] \leq \frac{1}{3}$

Note that the bounds can be generically improved by running A multiple times and taking a majority vote, therefore, the bounds $\frac{2}{3}$ and $\frac{1}{3}$ are somewhat arbitrary.

Definition 2.1.4 (PSPACE). The class PSPACE contains all languages L that can be decided by a (not necessarily efficient) Turing machine that uses at most polynomial space.

Uniform and non-uniform algorithms. Usually, we assume that our algorithms are uniform, i.e., they can be described by a Turing machine for all input lengths. If an algorithm is *non-uniform*, we model this by giving the Turing machine a non-uniform advice $z = z(|x|)$, that can be arbitrary, but may only depend on the input length, not on the concrete instance.

Oracle machines. Sometimes, we give an algorithm A access to an oracle \mathcal{O} , we denote this by writing $A^{\mathcal{O}}$. In this case, A has the option to call the oracle on some input x and receive the answer

y in a single computational step. If our algorithm A is implemented by a Turing machine, it will have an additional tape to communicate with the oracle, and a new instruction which calls the oracle. To query x to the oracle \mathcal{O} , A would write x onto the communication tape. Next, A invokes the instruction which calls the oracle, in which x is replaced by y on the communication tape.

The oracle \mathcal{O} might be a concrete function, i.e., a function chosen at random³, in which case the function is evaluated on the input and the output is written to the tape, or even a whole language class like NP, in which case x both describes the NP-language L as well as the instance, and either 1 (instance is in L) or 0 (instance is not in L) is written on the tape.

2.2 Black-Box Reductions

As we have already seen in the introduction, reductions are the primary way in cryptography to prove the security of constructions, where the security of the construction is reduced to one or multiple underlying primitives or assumptions. In case that the construction is black-box, i.e., the underlying primitives are generic and not a concrete implementation of a primitive, then usually also the reduction is black-box, i.e., it doesn't depend on the implementation of the underlying primitive. As many constructions don't depend on their underlying primitives, black-box reductions play an important role in cryptography.

Assume we have a black-box construction of a pseudo-random generator (PRG) from a one-way function (OWF). A reduction now works as follows: We assume that we have some adversary against the PRG (i.e., an adversary that can distinguish between true randomness and the output of the PRG) and then show that we can use this adversary to build a new adversary against the one-wayness of the OWF (i.e., it can invert the image of the one-way function on a random input). This new adversary may be black-box, if it does not depend on how the one-way function is implemented and/or if it does not depend on how the adversary against the pseudo-random generator works.

The above example already shows that there can be different definitions on what a black-box reduction means. Reingold, Trevisan and Vadhan [RTV04] classify different types of black-box reductions, and Baecher, Brzuska and Fischlin [BBF13] extended this classification and introduced the CAP-notion, where for the construction (C), the adversary (A) and the underlying primitive (P), we note whether the reduction uses this part as a black-box (B) or not (N).

Such an analysis of different types of black-box reductions is especially interesting if we look at lower bounds, which usually rule out some type of black-box reductions. I will define two types of black-box reductions, semi-black-box reductions and fully-black-box reductions. Semi-black-box constructions is the weaker definition here (i.e., every fully-black-box is also a semi-black-box reduction), and therefore, ruling them out with a lower bound is a *stronger* result. The result aforementioned by Impagliazzo and Rudich [IR89] can for example be cast as ruling out semi-black-box reductions.

Definition 2.2.1 (Semi-Black-Box/BNN Reduction). Let P be a primitive constructed from a second primitive Q . We say that there exists a semi-black-box reduction if the following holds: There exists an efficient algorithm G such that for every instance f of Q and every instance G^f of P , if there exists an efficient adversary A^f that breaks G^f , then there exists an efficient reduction

³Note that this function is chosen at random when the oracle is defined, and is deterministic afterwards: If it is evaluated twice on the same input, it will produce the same output.

S^f against Q . Note that S here can depend on the adversary A and the construction G , therefore, in the CAP notation, it is a BNN-reduction.

Definition 2.2.2 (Fully-Black-Box/BBB Reduction). Let P be a primitive constructed from a second primitive Q . We say that there exists a fully-black-box reduction if the following holds: There exist efficient algorithms G and S such that for every instance f of Q and every adversary A breaking an instance G^f of P , $S^{A,f}$ breaks the instance f of Q . Note that the reduction treats everything as a black-box, and is therefore a BBB-reduction in the CAP-notion.

Note that in the definitions of semi-black-box and fully-black-box reductions above, f does not need to be efficient. Indeed, this is essential for oracle separations, as Baecher, Brzuska and Fischlin [BBF13] show: Relativizing reductions imply fully or semi-black-box reductions, but not their counterpart where f is restricted to be efficient. Therefore, showing that no relativizing reductions exist only rules out black-box reductions for (possibly) inefficient primitives, but not necessarily black-box reductions restricted to efficient primitives.

Concrete and tight security. Sometimes, we are not only interested in whether there exists a reduction, but also how efficient a reduction is, as this informs concrete instantiations of cryptographic primitives: If we had a reduction from PRGs to OWFs which runs in time squared of the adversary against PRGs, then, to get a 128-bit secure PRG, we would need a 256-bit secure OWF. We call this relation between the adversaries against the two notions the *security loss* of a reduction. If the security loss is linear, i.e., they differ by only a constant, we call a reduction *tight*.

3 Proving Lower Bounds using Oracle Separations

Oracle separations are an application of idealized models to show lower bounds for black-box constructions, i.e., to show that certain assumptions are not sufficient to construct a cryptographic primitive in a black-box way. In this chapter, I will present three works that use oracle separations to prove lower bounds: The first paper compares two definitions of one-way functions, *weak* and *strong* one-way functions, and shows that the security loss in Yao’s construction [Yao82] of strong from weak one-way functions is inevitable. The second paper compares two notions of collision resistance, *existential* and *distributional*, and shows that one cannot build existential collision resistance from distributional collision resistance in a black-box way, with some restrictions on the black-box reduction. Finally, the last paper takes a closer look at extremely lossy functions, or short ELFs, and analyses the minimal assumptions needed to build them.

3.1 Different Types of Oracle Separations

The general structure of an oracle separation often follows the following pattern: To show that primitive P cannot be built from primitive Q in a black-box way, we construct an oracle world in which primitive Q exists, but we can build a successful adversary against any candidate construction for primitive P . As discussed in Section 1.2, every black-box construction relativizes, and therefore, we now know that even in the standard model, no black-box construction of primitive P from primitive Q can exist.

The way we construct this oracle world can be quite different depending on the primitives P and Q . There are, however, two classes of oracle separations most results fall into, which I call *Impagliazzo–Rudich-style* separations and *Simon-style* separations.

3.1.1 Impagliazzo–Rudich-style separations

Impagliazzo and Rudich [IR89] were the first to introduce the concept of oracle separations to prove lower bounds to cryptography. Their result, showing that one cannot build key agreement from one-way functions, works as follows: The oracle world they construct consists of a PSPACE oracle and a random oracle. The PSPACE oracle has the task of creating a level playing field: As a PSPACE oracle can even execute exponential-time algorithms, no key agreement or one-way function can exist relative to it alone. The random oracle now re-introduces one-way functions: A random oracle is a one-way function with overwhelming probability, even in the presence of the PSPACE oracle.

Now, the question remains whether the introduction of a random oracle also allows for the existence of key agreement. Impagliazzo and Rudich answer this question negatively by constructing an adversary against every key agreement protocol in which two parties compute a shared key: Impagliazzo and Rudich show that an eavesdropping adversary can learn the key from the transcript (i.e., the messages sent between the two parties) by running an iterative process, where in each round, the adversary either learns a new query from the intersection of random oracle queries by both parties, or calculates the correct key. As there are only polynomially many queries to the random oracle in the protocol, the adversary will learn the key if the process is iterated sufficiently often. Note that this adversary makes heavy use of the PSPACE oracle, and therefore, the attack strategy only works in this oracle world.

An *Impagliazzo–Rudich-style* separation is a separation that has this two-fold type of oracle world, containing one oracle that gives a level playing field (in most cases, this is a PSPACE oracle), and an oracle that allows for the weaker primitive to exist again. This style of separation has been used many times, e.g., Gertner et al. [GKMRV00] showed that public-key encryption and oblivious transfer are incomparable and Rudich [Rud88] separated one-way permutations from one-way functions.

Another separation I want to highlight is the work by Gennaro and Trevisan, using oracle separations to show lower bounds on the complexity of building pseudo-random generators from one-way permutations [GT00], as they introduce the compression (or reconstruction) technique, an important technique to show security relative to a random permutation oracle Π : Assuming the existence of a successful adversary against the one-wayness of the permutation, they construct an encoding algorithm for Π , as well as a corresponding decoding algorithm, where the two algorithms use the adversary to give an encoding of Π that is shorter than information-theoretically possible for a random permutation – this contradicts the existence of the adversary against one-wayness. This technique has since been used in many separation results, e.g., by Pietrzak [Pie08] to show that combiners of collision-resistant hash functions have long outputs and by Gennaro, Gertner and Katz [GGK03] who show lower bounds on the efficiency of encryption schemes and digital signature schemes.

3.1.2 Simon-style separations

Sometimes, however, a Impagliazzo–Rudich-style separation is not fine-grained enough: For example, if we want to show that we cannot build something from a one-way function, we use a random oracle as an idealized version of it. Now, however, a random oracle is much stronger than a one-way function, so it might be the case that the random oracle already implies the primitive we want to separate one-way functions from. A famous example here is the case of collision-resistant hash functions, which exist relative to a random oracle, but are not known to be constructable from one-way functions. To show a separation between one-way functions and collision-resistant hash functions, Simon [Sim98] added another oracle, next to a PSPACE and a random oracle F : An oracle which, for a given construction h^F , returns a (random) collision⁴. I often call this latter oracle a *breaker* oracle, as it is specifically designed to break a certain primitive.

In these *Simon-style* separation results, it is much more complicated to show that the weaker primitive (one-way functions, in our example) does exist, as we have to prove that the breaker-oracle

⁴Note that the collision is not chosen uniformly from the set of all collisions — instead, an element from the domain of h is chosen at random, and then a second element from the domain, conditioned on having the same image as the first element.

cannot be successfully used by an adversary against the weaker primitive. One major obstacle here is that the construction can now also make use of the breaker oracle — so an adversary might call the breaker oracle on a construction which uses the breaker oracle internally, which can lead to recursive oracle queries. In our example, now showing that the stronger primitive, collision-resistant hash functions, does not exist is comparatively easy — however, this is not the case in general, in Section 3.4, we will see an example of a Simon-style separation where showing that the stronger primitive does not exist is the main technical challenge.

To circumvent having to deal with recursive oracle queries, most Simon-style separations after the original paper use the two-oracle technique introduced by Hsiao and Reyzin [HR04] who showed that if we only want to exclude *fully* black-box constructions (instead of all relativizing constructions), we can give the construction a smaller set of oracles compared to the adversaries. Therefore, we can define our oracles in a way that we give the breaker oracle only to adversaries, not to constructions — this way, we can assume that any construction given to the breaker oracle does not itself contain calls to the breaker oracle, and therefore, no recursive calls to the breaker oracle are necessary.

Examples of Simon-style oracle separations together with the two-oracle technique include Hsiao and Reyzin [HR04] who separate private-coin collision-resistant hash functions from public-coin collision-resistant hash functions, Haitner et al. [HHR07] who give a lower bound on the number of communication rounds to build a statistically hiding commitment scheme from trapdoor permutations in a fully black-box way, and Asharov and Segev [AS15] who show that collision-resistant hash functions cannot be built from indistinguishability obfuscation in a fully black-box way.

3.2 Limits of Oracle Separations

While oracle separations are a very useful tool to prove lower bounds on the construction of primitives, there are always restrictions to the bounds, as oracle separations only rule out black-box constructions. However, some sort of restriction is fundamentally necessary: First note that from a purely logical view, to rule out $\exists\text{OWF} \Rightarrow \exists\text{KA}$ would require us to rule out key agreement protocols in general, as any secure key agreement protocol means that the implication is true. Therefore, as long as we think that key agreement exists even from other assumptions than one-way functions, we cannot rule out such a general implication. From a more practical perspective, it could be that every secure one-way function has an additional property that makes key agreement possible (e.g., some group structure as used in the Diffie-Hellman key agreement). As we don't definitely know how secure one-way functions are constructed, it is not possible to rule out such a case. Therefore, such a lower bound will usually be for some kind of black-box construction.

However, lower bounds for black-box constructions also differ in the restrictions. For example, Impagliazzo and Rudich [IR89] rule out any semi-black-box reduction (Definition 2.2.1), while Simon-style separations using the two-oracle technique only rule out fully-black-box reductions (Definition 2.2.2). Other separations also restrict how often and in which order a reduction may call the oracles [FGHMY24] or rule out a limited class of constructions [BMM24].

It turns out that meaningfully restricted lower bounds can be very informative: They still show evidence that a construction is unlikely to exist, they might lay the foundations for less restrictive lower bounds in the future, or they might even guide constructions that overcome the lower bounds.

3.3 On Derandomizing Yao's Weak-to-Strong OWF Construction

Chris Brzuska

Published at **TCC 2021**

Geoffroy Couteau

Publication **1**

Pihla Karanko

Felix Rohrbach

In this paper, we give a lower bound on the efficiency of constructing strong one-way functions from weak one-way functions: We show for a class of non-adaptive constructions that Yao's construction [Yao82] is probably optimal. We will be using a Simon-style oracle separation to show this result.

Strong one-way functions are the usual definition for a one-way function, requiring that every adversary has at most negligible probability of inverting the image of the function on a random input:

Definition 3.3.1 (Strong one-way functions). A function f is called a (strong) one-way function, if for all adversaries \mathcal{A} there exists a negligible function ε such that for all $n \in \mathbb{N}$:

$$\Pr_{\mathcal{A}, x \leftarrow \{0,1\}^n} [\mathcal{A}(1^n, f(x)) \in f^{-1}(f(x))] \leq \varepsilon(n)$$

A *weak* one-way function, however, only requires the function to be hard-to-invert on a polynomial fraction of its inputs:

Definition 3.3.2 (Weak one-way functions). A function f is called a weak one-way function, if there exists a polynomial $p(n)$ such that, for all adversaries \mathcal{A} and all sufficiently large $n \in \mathbb{N}$:

$$\Pr_{\mathcal{A}, x \leftarrow \{0,1\}^n} [\mathcal{A}(1^n, f(x)) \in f^{-1}(f(x))] \leq 1 - \frac{1}{p(n)}$$

In this case, we sometimes call f p -weak.

Clearly, a weak one-way function is easier to construct, but less useful for applications, and so it is fruitful to construct a strong one-way function from a weak one-way function. Yao [Yao82] provided such a transformation using a *direct-product* construction: The input of the strong one-way function is split into many chunks of length n , then the weak one-way function is called on each chunk and all the outputs concatenated together are the output of the strong one-way function.

Unfortunately, Yao's construction is not security preserving: for a p -weak one-way function with input length n , we need $n \cdot p(n)$ many chunks, i.e., the input length of the strong one-way function must be of length $n^2 \cdot p(n)$ to be secure.

There exist some security-preserving constructions as well: Goldreich, Impagliazzo, Levin, Venkatesan and Zuckerman [GILVZ90] as well as Haitner, Harnik and Reingold [HHR06] provide constructions which only require an input length of $n \log(n)$ for the strong one-way function. However, these constructions call the weak one-way function in an adaptive form (i.e., the output of the weak one-way function is later fed again into the weak one-way function) and they only work for regular one-way functions (i.e., one-way functions where the number of preimages is the same for all images).

Our main research question is now: Could there be a security-preserving non-adaptive construction that works for all one-way functions? For this, we model non-adaptive constructions by consisting of a *pre-processing* phase, which takes the input of the strong one-way function and produces inputs for the weak one-way function, then in parallel calls the weak one-way function with all these inputs, and finally the outputs of the weak one-way function are given to the *post-processing*, which computes the output of the strong one-way function (see Figure 3.1).

We answer this question mostly negative: For non-adaptive constructions where the post-processing is *injective-ish*, i.e., every image of the post-processing has at most polynomially many images, no security-preserving black-box construction can exist.

Theorem 3.3.3 (Main Theorem (informal)). *Let g be a non-adaptive, fully black-box construction of a strong one-way function from a p -weak one-way function f , where the post-processing is injective-ish. Then, the input length of g is larger than $c \cdot p(n)$ for a constant c .*

3.3.1 Related Work

There are a few previous results which look into the limits of creating strong one-way functions from weak ones. Lin, Trevisan and Wee [LTW05] show that every fully black-box construction of an ε -secure strong one-way function from a p -weak one-way function must make at least $q = \Omega(p(n) \log(1/\varepsilon))$ many queries to f . They further show that a fully black-box construction cannot be perfectly security preserving: If f has input size n , the input size of the strong one-way function g must be at least $n + \Omega(\log(1/\varepsilon)) - O(\log q)$. Lu [Lu09] extends this result to the weakly black-box setting with bounded non-uniformity.

We note that there is a large gap between these two results and Yao’s construction, however: Even for the extreme case of a $2^{-\mu n}$ -secure strong one-way function, the input size is only bound to be at least $(1 + \mu)n - \log p$, while Yao’s construction requires an input size of $n^2 p(n)$. Note that these results, contrary to ours, also consider *adaptive* constructions.

Lu [Lu06] shows that one cannot *non-adaptively* amplify the security of a extremely weak one-way function, i.e. a one-way function which is only hard on a *negligible* fraction of its inputs, to a weak one-way function with a fully black-box construction with constant depth: Such a construction would need a circuit size of larger than $2^{\text{poly}(n)}$. However, as we assume the existence of a weak one-way function which is hard on a polynomial fraction, this lower bound does not affect us.

3.3.2 Proof Overview

Our result uses a Simon-style oracle separation, so we show that relative to a set of oracles, weak one-way functions exist, but strong ones with our restrictions do not. We first introduce two oracles, a PSPACE oracle and a random permutation oracle F . Now, relative to these two oracles, both weak and strong one-way functions would exist, as F is hard to invert for nearly all inputs (with high probability over the choice of F). Therefore, we introduce another oracle Inv , which weakens the one-wayness of the random oracle in the following way: We define roughly a $(1 - \frac{1}{\text{poly}})$ fraction of

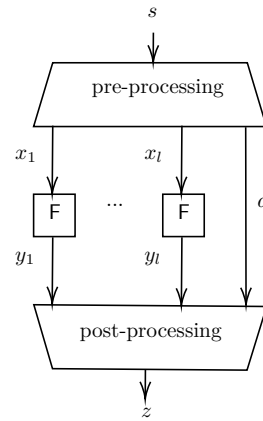


FIGURE 3.1 — A non-adaptive construction of a strong one-way function h^F from F

inputs as *easy* and the rest as *hard*. Now, when we give Inv an image y of F , it will return $x = F^{-1}(y)$ if x is easy – otherwise, it will not invert y and return \perp .

We now prove two statements relative to these two oracles: That F is a weak one-way function, and that no non-adaptive strong one-way function with our restrictions exists. Showing that F is a weak one-way function can be proven using techniques that are similar to showing that a random oracle (without the Inv oracle) is a strong one-way function.

Proving that no non-adaptive strong one-way function with small input size and injectiveish post-processing requires some more care. The main idea is to first invert the post-processing. As the post-processing is injectiveish, there are only polynomially many preimages – we assume in the following that there exists only one, if multiple preimages exist, the following steps have to be done for each preimage. Next, we try to invert each parallel call to F – on average, this will succeed for roughly a $(1 - \frac{1}{\text{poly}})$ fraction of calls. In the last step, we show that the fraction of inverted calls is enough to reconstruct the (small) input s to the candidate strong one-way function, by showing that the entropy of s given the inverted calls is small enough to brute-force s , i.e., we show that we can invert the pre-processing with our partial knowledge of its output. Therefore, we now have an efficient adversary against any candidate strong one-way function with our restrictions.

3.3.3 Limitations of our result

The type of strong one-way functions we rule out has quite a few restrictions — however, most of the restrictions are directly motivated by what we want to show: The input size has to be small, because we want to show that no construction exists with small input size (and we know that with Yao’s construction, there exists a construction with large input size), and the one-way function should be non-adaptive, as length-preserving constructions exist for adaptive constructions.

The requirement that the post-processing has to be injectiveish, however, seems to be an artifact of our proof technique: If the post-processing would be highly compressing, we could not go over all preimages of the post-processing and wouldn’t know where to invert F using the Inv oracle.

Indeed, ruling out compressing post-processing seems to be impossible in our oracle world. The reason here is that the random function F destroys correlation: While the output of the pre-processing will be highly correlated for small inputs, the output of the parallel calls to F are uncorrelated. However, in the standard model, a weak one-way function would not destroy this correlation. Getting rid of the injectiveish post-processing constraint would therefore require an oracle world that takes this correlation into account.

3.3.4 My Contributions

This paper is a joint work of Chris Brzuska, Geoffroy Couteau, Pihla Karanko and me. The main ideas of the paper were discussed in regular meetings and are therefore owed to all authors equally. My main contribution in this paper lies in the construction of the oracles used for the oracle separation and our modeling of non-adaptive constructions.

3.3.5 Versions

This paper has been published at TCC 2021 [BCKR21]. A minor revision of this paper has been published in 2023 on ePrint [BCKR23], which I include in this thesis. The revision improves the

readability of the paper, changes a previously misleading comparison to secret sharing, and improves some of the parameters. Finally, a standard averaging argument was included for completeness.

3.4 On the Difference between Distributional and Existential Collision-Resistance

Chris Brzuska

In Submission

Christoph Egger

Publication 2

Pihla Karanko

Felix Rohrbach

In this paper, we also show a Simon-style oracle separation between two closely related definitions: *Existential* collision resistance and *distributional* collision resistance. Collision resistance is a property of hash functions, i.e., functions which have a smaller output length compared to their input length, which demands that it is computationally hard to find collisions for this hash function h , i.e., no adversary can find two $x \neq x'$ such that $h(x) = h(x')$. Collision-resistant hash (short CRH) functions play an important role in many cryptographic protocols, e.g. hashing the transcript into the key in the TLS handshake protocol [Res18] or hashing messages before signing them [HK06; Mir06].

Existential collision resistance is the more commonly used definition, which declares that an adversary may not find any such collision. On the other hand, distributional collision resistance, introduced by Dubrov and Ishai [DI06], requires the adversary to find a *random* collision: The distribution of collision given by the adversary should be close to the distribution of Simon’s oracle⁵, which samples a random x and a random preimage of $h(x)$ (see Figure 3.2).

```

COLh,n
-----
x ←ₛ {0, 1}n
x' ←ₛ {x' | h(x) = h(x')}
return (x, x')
```

FIGURE 3.2 — Simon’s oracle

A technical oddity of existential CRH functions is that if we want to have security against non-uniform adversaries, we need to *key* the hash function, i.e., the hash function depends on a key that is randomly chosen: If the hash function would be unkeyed, then a non-uniform adversary could hard-code a specific collision for each input length, breaking the security of existential collision resistance. Distributional collision-resistant (dCRH) functions, on the other hand, don’t have this problem: As an adversary is required to output a random collision, a hard-coded collision does not help them. Also, as distributional collision resistance is a weaker definition, it might be easier to construct.

Therefore, distributional collision resistance has some advantages compared to the existential version. Further, a recent construction of constant-round statistically hiding commitments from distributional CRH by Bitansky, Haitner, Komargodski and Yogev [BHKY19] shows that it is a stronger notion compared to one-way functions [HHR07].

However, whether we can build existential CRH functions from distributional ones has been an open problem. For one-way functions, we know that regular one-way functions can be built from distributional one-way functions [IL89], but a similar construction for collision resistance does not exist. In our paper, we present evidence that such a construction will not exist: We show that, for a

⁵Simon’s oracle here references to the breaker oracle used by Simon [Sim98] to show that collision-resistant hash functions cannot be built from one-way functions (see Section 3.1.2).

large class of reductions, no fully black-box construction of an existential from a distributional CRH is possible.

3.4.1 Oracles

Again, we use a Simon-style oracle separation to prove this lower bound: We create an oracle world relative to which distributional CRH functions exist, but existential CRH functions do not. For this, we add a PSPACE oracle as well as a random 2-regular function F (i.e., every image of F has two pre-images) to our oracle world. Now, we still need to define a breaker oracle which makes sure existential collision-resistance does not exist in our oracle world. Unfortunately, we cannot use Simon's oracle (cf. Fig. 3.2), as relative to it, even distributional collision resistance does not exist. Instead, we split the image of F into a hard and an easy set and then create an oracle that, for a construction h^F , only returns h -collisions that have F -collisions on the *easy* set. Therefore, intuitively, existential collision-resistance of any construction h should break, as our breaker oracle will find some collision, but F should still be distributional collision-resistant, as the breaker oracle will only return collisions from a subset of all collisions.

3.4.2 Restrictions

While we make considerable progress towards showing that black-box constructions of existential from distributional CRH functions are not possible, we have to add some restrictions to our result. First, we restrict our reductions to be *single-query*: The reduction can only make one query to the breaker oracle, and then repeated queries to F afterwards. This constraint is analogous to Folwarczný et al. [FGHMY24], who show that one-way functions do not imply, via a *single-query* black-box reduction, the hardness of any total NP search problem.

Secondly, we require constructions (and adversaries) to be *polynomially honest*, i.e., the input to any oracle call is required to be polynomially related to the security parameter, which disallows tiny queries, i.e., of constant or logarithmic length. Further, our result only rules out fully black-box constructions, i.e., the construction does not get access to the breaker oracle. Finally, for each construction h^F , we assume the oracle F is only called on one security parameter.

For future work, we see especially removing the first restriction as an interesting open question — see also our discussion of the restrictions in the paper in Section 1.4 and Section 2.

3.4.3 F is a dCRH

Now, we prove that in our oracle world, distributional CRH functions exist, but existential do not. We will start in this section by showing that F is a distributional CRH function. For this, we will use the compression technique by Gennaro and Trevisan [GT00], showing that the existence of a successful adversary against distributional collision-resistance would imply that we can compress F beyond information-theoretic limits.

The main idea of the compression algorithm is that for a hard collision $F(x) = F(x') = y$ the adversary will help us recover x' from x , and so we only need to save the mapping $F(x) = y$. However, the main technical challenge here is that for this to work, we have to encode on which randomness the adversary gives us hard collisions, and as the adversary may use a polynomial amount of randomness, encoding this randomness requires more bits than we save by leaving out the mapping $F(x') = y$. Therefore, we define a random subset of all randomness of size 2^n (independent of F), and then show

that if we only run the adversary on randomness from this subset, we still get sufficiently many distinct hard collisions. Further, as encoding “good” randomness in the subset is much more efficient, we can now use this to show that we can compress F using the adversary. As this is impossible, we have shown that such an adversary cannot exist.

3.4.4 No existential CRH

Next, we show that relative to our oracles, no existential CRH function exists. For this, we have to show that for any construction h^F , with high probability, there exists a collision our collision breaker can return (i.e., one h -collision that does not induce any F -collision in the hard set — we call such a collision, with some further technical restrictions, a “good” collision). For this, we define random variables X_i that draw a random collision (from the same distribution as Simon’s oracle) and set the random variable to 1 if it is a “good” collision, and to 0 otherwise. Next, we sum over a large number of X_i and want to show that, with high probability, the sum is at least 1 (i.e., there is at least one X_i that generated a “good” collision), showing that there exists a collision the oracle can return.

There are multiple ways to bound the unlikely event that none of the random variables samples a “good” collision, i.e., that the sum of random variables is 0. The simplest of such bounds is the Markov inequality. Unfortunately, the probability that a random collision does not induce a hard F -collision can be quite small, and in this case, Markov does not give us a strong enough bound that there exists at least one “good” collision. Another option is the Chebychev inequality for sums of random variables, which gives us a stronger bound, but requires the random variables to be pairwise independent, which they are not quite — if two h -collisions induce the same F -collision, then the probability that one of these h -collisions doesn’t have a hard F -collision depends on the other one.

However, we can use a more generalized Chebychev bound, which does not depend on the pairwise independence, but for which we have to bound the *covariance* of each pair of collisions, i.e. the dependence between them, which is the main technical challenge of this part.

3.4.5 My Contributions

This paper is a joint work of Chris Brzuska, Christoph Egger, Pihla Karanko and me. The main ideas of the paper were discussed in regular meetings and were written up jointly, and are therefore owed to all authors equally. While I joined the project after it had already started, I contributed to the paper significantly. In Section 6 (F is a dCRH) I contributed to the construction of the Encoder and Decoder algorithms of the compression argument, wrote Lemma 6.2 and contributed to Lemma 6.7 and 6.8. In Section 5 (no CRH), I developed the idea to combine Chebychev and Bienaymé to bound the sum of random variables, and also wrote the corresponding proofs (Sections 5.1 and 5.2), including the lemma bounding the probability of a large intersection of F -collisions (Lemma 5.4).

3.4.6 Versions

This paper has not been published yet.

3.5 How to Replace the Random Oracle Model as an Assumption

Bellare and Rogaway [BR93] introduced the random oracle model as general paradigm to give heuristic security arguments for protocols that have no proof in the standard model by replacing the hash function in the protocol with a truly random function. This model has been applied very successfully for many protocols.

I will discuss the random oracle model in more detail in Chapter 4.3. In this section, I will focus on how to replace the random oracle as an assumption: As already discussed in the introduction (Chapter 1.3), the random oracle is not sound, as counterexamples exist for protocols that are secure in the random oracle model, but won't be secure in the standard model. Therefore, a line of work has tried to replace the random oracle assumption with something that can be instantiated in the standard model. *Correlation Intractability* [CGH98] builds a hash function based on an internal compression function which should be indistinguishable from a random oracle, even if the adversary has access to the internal compression function. However, current constructions of correlation intractability require extremely strong assumptions [CCR16].

Another attempt at replacing the random oracle assumption is the *Universal Computational Extractors* [BHK13] (short UCE) framework. Here, a two-stage adversary has to distinguish between a real world or an idealized world, where the first stage interacts with either a keyed random oracle or a keyed hash function (without knowing the key) and the second stage gets access to the key, but no access to the functions anymore, with a limited amount of leakage send from the first stage to the second. However, it is not quite clear which versions of UCE are instantiable [BFM14; BST16].

3.6 Searching for ELF's in the Cryptographic Forest

Marc Fischlin

Published at **TCC 2023**

Felix Rohrbach

Publication **3**

In this paper, we focus on a new primitive to replace random oracle assumptions: Recently, Zhandry [Zha16; Zha19] proposed a new class of functions called extremely lossy functions (short ELF's). These extremely lossy functions are either very lossy, meaning that the image size of the function is only of polynomial size, or they are injective, based on a public key. Further, it should be indistinguishable whether the public key corresponds to a function in injective or in lossy mode. Now, of course, if we have a polynomial image size, there will always be some adversary which can distinguish the injective mode from the lossy one – therefore, we select the image size in lossy mode based on the number of queries from the adversary.

Definition 3.6.1 (Extremely Lossy Function). An extremely lossy function consists of two efficient algorithms (Gen, Eval) of which Gen is probabilistic and Eval is deterministic and it holds that:

- (a) For $r = 2^{\text{in}(\lambda)}$ and $\text{pk} \leftarrow_{\$} \text{Gen}(1^\lambda, r)$ the function $\text{Eval}(\text{pk}, \cdot) : \{0, 1\}^{\text{in}(\lambda)} \rightarrow \{0, 1\}^*$ is injective with overwhelming probability.
- (b) For $r < 2^{\text{in}(\lambda)}$ and $\text{pk} \leftarrow_{\$} \text{Gen}(1^\lambda, r)$ the function $\text{Eval}(\text{pk}, \cdot) : \{0, 1\}^{\text{in}(\lambda)} \rightarrow \{0, 1\}^*$ has an image size of at most r with overwhelming probability.

- (c) For any polynomials p and d there exists a polynomial q such that for any adversary \mathcal{A} with a runtime bounded by $p(\lambda)$ and any $r \in [q(\lambda), 2^{\ln(\lambda)}]$, algorithm \mathcal{A} distinguishes $\text{Gen}(1^\lambda, 2^{\ln(\lambda)})$ from $\text{Gen}(1^\lambda, r)$ with advantage at most $\frac{1}{d(\lambda)}$.

Now, while these ELF_s are relatively simple definition-wise, they enable us to remove random oracles in a number of applications, including boosting adaptive to selective security in signatures, constructing output-intractable hash functions, point obfuscation in the presence of auxiliary information [Zha16], probabilistic indistinguishability obfuscation [ACH20], instantiate the hash function for OAEP and Fujisaki-Okamoto transforms [MOZ22] and instantiating the hash-then-evaluate paradigm [BCEKM24], as some examples.

However, the biggest upsides of ELF_s are that they can be constructed from standard *ish* assumptions: Zhandry gives a construction that follows from the exponential hardness of the decisional Diffie-Hellman problem, which arguably holds for example over elliptic curves. This is in contrast to previous assumptions used to replace the random oracle, which needed complex assumptions or for which it is unclear whether they are instantiable.

Still, Zhandry is using a public-key assumption here to implement a primitive which does not necessarily look public-key. This motivates our research question for this paper: *Do we need public-key assumptions to build ELF_s?* In his paper, Zhandry asks whether it might be possible to build ELF_s from exponentially hard collision-resistant hash functions, and Holmgren and Lombardi [HL18] suggested that their construction of one-way product functions might be enough to build ELF_s from it.

We answer this question two-fold:

- ELF_s cannot be built in a black-box way from a large number of symmetric primitives, including all the ones suggested above. Indeed, everything that does exist relative to a random oracle is not enough to build an extremely lossy function.
- ELF_s do not imply key agreement (in a fully black-box way), therefore it might be possible to build ELF_s from a non-public-key assumption.

This still leaves the question which other assumptions would let us construct extremely lossy functions. We think that hardness of SZK, the class of problems with a statistical zero-knowledge proof, might be an interesting candidate here: similarly to extremely lossy functions, it lies somewhere between the classical set of symmetric cryptography, but does not imply public-key cryptography either.

We note that both separations also apply to (moderately) lossy functions, i.e., where the lossy mode is of exponential size. Indeed, we formulate the first separation between symmetric primitives and lossy functions, as this is the stronger result.

3.6.1 Oracle Separations

We use two Impagliazzo–Rudich style oracle separations for our result. For our separation between symmetric primitives and lossy functions, we show that relative to a $\text{PSPACE}^{\mathcal{O}'}$ oracle and a random oracle \mathcal{O} (the \mathcal{O}' is a random oracle independently chosen from \mathcal{O}), there exists an efficient distinguisher between the lossy mode and the injective mode of any candidate lossy function: First, we show that every candidate lossy function whose hardness depends on \mathcal{O} must make *heavy queries* to the random oracle, i.e., a small set of queries happens for many different inputs x to the lossy

function. The second realization is that the distinguisher only needs to know the random oracle \mathcal{O} at the positions of the heavy queries to distinguish between a lossy and an injective mode. Further, as these heavy queries can be found efficiently, the distinguisher queries these heavy queries to the random oracle and then uses this information to distinguish a function in lossy mode from one in injective mode. This technique is similar to the one used by Pietrzak, Rosen and Segev [PRS12] used to show that it is not possible to amplify the lossiness of a lossy function in a black-box way.

The second separation between ELF's and key agreement is even closer to the Impagliazzo–Rudich separation, as it reuses the adversary against key agreement. For this, we first build an extremely lossy function relative to a PSPACE oracle and a random permutation oracle Π , which is inefficient, i.e., it makes exponentially many queries to the underlying random permutation Π . In the next step, we show that we can simulate access to the ELF by an efficient, stateless algorithm Wrap^Π with only small simulation error. Now, as Wrap is efficient and stateless, we can view Wrap as part of the protocol participants — meaning we have a key agreement protocol relative to a PSPACE and a random permutation oracle, for which the Impagliazzo–Rudich adversary can compute the shared key.

3.6.2 A Note on Random Oracles and the Random Oracle Model

It might be unintuitive that it is not possible to build extremely lossy functions relative to a random oracle – after all, it is a random oracle what an ELF should replace. However, this might be unavoidable: As a replacement for a random oracle will never be as powerful as a random oracle, it might need some sort of property where it is stronger than a random oracle, to counter the imbalance – i.e., something that cannot be implemented relative to a random oracle.

I also want to highlight that we use the random oracle in two contexts here. The first one is in the lower bound, where we create an oracle world containing a random oracle, to then deduce a lower bound in the standard model. The second one is in the context an idealized model, the random oracle model, in which we build something relative to a random oracle (and then prove it secure in this model), and then heuristically assume that security holds for it in the standard model if we replace the random oracle with a cryptographically secure hash function.

3.6.3 My Contributions

This paper is a joint work by Marc Fischlin and me. The main ideas of the paper were discussed in regular meetings and were written up jointly, and are therefore owed to all authors equally. I took the lead in writing (and proving) chapter 4 (On the Impossibility of Building (E)LFs in Oraclecrypt). In chapter 5 (On the Impossibility of Building Key Agreement Protocols from (Extremely) Lossy Functions), I had significant contributions to the proof, especially in making the simulation stateless, which was essential for the proof to work. Further, I wrote the part extending the result from lossy functions to extremely lossy functions.

3.6.4 Versions

This paper has been published at TCC 2023 [FR23b]. The full version of the paper, which contains multiple proofs left out from the conference version due to page restrictions, has been published on ePrint [FR23a]. The peer-review for TCC happened based on the full version of the paper.

3.7 Contributions of This Chapter

In this chapter, I have presented four oracle separations to show lower bounds, a Simon-style separation in each of the first two papers, and two Impagliazzo–Rudich-style separations in the last paper. The first separation explains why we haven't seen improvements in non-adaptive constructions for strong from weak one-way functions in the last forty years, as we show that Yao's construction from 1982 is probably nearly optimal if we restrict ourselves to non-adaptive constructions. This also highlights the need for adaptive constructions to have constructions without such a large security loss.

In the second oracle separation, we show that a relatively new assumption, distributional collision resistance, is an interesting intermediate definition between one-way functions and existential collision resistance. As we already know constructions from distributional collision resistance, this separation shows that these constructions do not need to rely on existential collision resistance.

Finally, we show two oracle separations for extremely lossy functions, which establish this primitive somewhere between most symmetric primitives and public key cryptography. As to date, we only have one construction of extremely lossy functions, we hope that these separations help to guide constructions from different assumptions, e.g., a post-quantum secure ELF.

While these four separations may be useful for different reasons, they all show us how the different cryptographic primitives depend on each other, which in itself is a reason to study them, as it helps us understand how and why cryptography, on a very fundamental level, works.

4 Using Idealized Models for Constructions

In this chapter, I will present two works that focus on the constructive use of idealized models. In the first one, we construct a single-to-multi-theorem transformation for non-interactive statistical zero-knowledge arguments, which require a *common random string*, a shared, honestly-generated uniform string, to work. Such a string could be generated by a trusted (third) party, it could be from a trusted source in a protocol the non-interactive protocol is embedded in, or, if we would have access to a random oracle, it could be the output of the oracle on the all-zero string. But as we don't have either in general, we assume the existence of such a common random string as idealized model.

The second work introduces a variation of the random oracle model, the universal random oracle model, which models one universal random oracle for every adversary instead of a new random oracle for each adversary. We argue that the universal random oracle model is a closer representation of hash functions, as these are also fixed for all adversaries. Thereby, the heuristic step from a proof in the idealized model to an security argument in the standard model is a smaller step for the universal random oracle model compared to the usual random oracle model.

4.1 Non-Interactive Zero-Knowledge and the Common Reference String Model

Zero-Knowledge arguments [GMR85] are interactive proofs in which a prover convinces a verifier that a string x is in a language $L \in \text{NP}$, without revealing any further information. For this, the prover gets access to a witness w for $x \in L$, but should not reveal w to the verifier. This is formalized by a simulator ZKSim which outputs a transcript indistinguishable from a true transcript, while not having access to the witness w . Depending on how close the simulator is to the transcript, we distinguish between three different types of zero-knowledge: *computational*, if the simulator output of the simulator is computationally indistinguishable from a real transcript, *statistical*, if it is statistically indistinguishable, and *perfect*, if the distributions are identical.

Definition 4.1.1 ((Statistical) Zero-Knowledge Argument). Let (P, V) be a pair of interactive polynomial-time Turing machines, where we call P the prover, V the verifier, and $\langle P, V \rangle$ denotes the output of V after running together with P . We call (P, V) a zero-knowledge argument for an NP-relation R_L for the language L if the following three properties hold:

Completeness For every $(x, w) \in R_L$,

$$\Pr[\langle P(x, w), V(x) \rangle = 1] \geq \frac{2}{3}.$$

Soundness For every $x \notin L$ and every potentially malicious prover P^* ,

$$\Pr[\langle P^*(x), V(x) \rangle = 1] \leq \frac{1}{3}.$$

(Statistical) Zero-Knowledge For every interactive polynomial-time Turing machine V^* , there exists a simulator ZKSim such that the protocol transcript of $\langle P(x, w), V^*(x) \rangle$ and $\text{ZKSim}(x)$ are statistically indistinguishable.

We get computational or perfect zero-knowledge arguments by replacing the statistically indistinguishable with computationally or perfectly indistinguishable.

A famous example of a zero-knowledge argument is 3-colorability of a graph, which asks whether all vertices in a graph can be colored with three colors, without having two neighboring vertices with the same color.

Example 4.1.2 (Graph 3-Coloring). The zero-knowledge argument system (P, V) for graph 3-coloring works as follows: Both prover P and verifier V get as input a 3-colorable graph. P additionally gets a 3-coloring $\phi(v) \in \{1, 2, 3\}$ for every node v . The following protocol now happens between P and V :

- P: The prover selects a random permutation π over $\{1, 2, 3\}$ and sends commitments to $\pi(\phi(v))$ to the verifier for every node v
- V: The verifier selects two neighboring nodes (u, v) at random and sends them to the prover.
- P: The prover opens the commitments for $\pi(\phi(u))$ and $\pi(\phi(v))$.
- V: The verifier checks that the openings are to two different numbers and accept (return 1) in this case, otherwise it will reject (return 0)

For the interaction with an honest prover, the verifier will always accept. However, if the input graph is not 3-colorable, there will always be some probability that the verifier chooses a pair of nodes (u, v) which have the same color. By running the protocol multiple times and only accepting if the verifier accepts every round, we can amplify the rejection probability to over $\frac{2}{3}$.

For the zero-knowledge property, we can construct a simulator that chooses colors for every node at random and creates commitments for them. As the commitments are hiding, the verifier cannot distinguish the colors and therefore has to choose two edges (u, v) nearly independent of the colors. Now, with probability $\frac{2}{3}$, the colors of the two nodes differ and we can open the commitments – otherwise, we just try again with different colors.

For many applications, it is helpful if we can remove the interaction between the prover and the verifier, i.e., we only have a single message from the prover to the verifier. However, such non-interactive zero-knowledge protocols are impossible for languages outside of BPP: To decide whether $x \in L$, we use the generator to create a message from the prover to the verifier, and then check whether the verifier outputs 0 or 1 on receiving this message. As the simulator has to output something indistinguishable from a real transcript, the output of the verifier must also be close to the real transcript, giving us a probabilistic efficient decider for $x \in L$.

Blum, Feldman and Micali [BFM88] discovered that one can circumvent this impossibility result by assuming the existence of an honestly generated string accessible by all parties. This random string can be either uniformly at random, or generated from some distribution – in the former case, we call it the Common Random String (CRandS) model, in the latter case the Common Reference

String (CRS) or public parameters model. Generally, the common random string model is preferred, as it can be created more easily, e.g. by xor-ing multiple semi-trusted sources.

One problem with both models is that for many constructions, they are only secure for an a-priori fixed number of proofs (we call these constructions *single-theorem* or *bounded*), while preferably, we would like to have a construction that is secure for any polynomial number of proofs using the same string (*multi-theorem* or *unbounded*). Conveniently, for *computational* zero-knowledge arguments, Feige, Lapidot and Shamir [FLS90] show how to transform any single-theorem construction to a multi-theorem construction.

The Feige-Lapidot-Shamir transformation works as follows: Assume we have a single-theorem zero-knowledge argument system for any language in NP using a common random string crs, and we want to prove membership in some language L . Further, we assume that a pseudo-random generator $G : \{0, 1\}^n \mapsto \{0, 1\}^{3n}$ exists. We now extend crs by appending $3n$ bits which we call crs^{aux} , and create an OR-language L' : $x \in L$ or $\text{crs}^{\text{aux}} \in G(\{0, 1\}^n)$. Now, every zero-knowledge argument system is also a witness-indistinguishable argument system [FS90], i.e., it is indistinguishable which witness the prover used to prove a statement. Therefore, the honest prover can just use the witness for $x \in L$ to prove membership in the new language L' , and with overwhelming probability, the malicious prover will not be able to come up with a proof for $x \in L'$ if $x \notin L$, as the last $3n$ bits of the common random string will not be in the image of G with overwhelming probability. The simulator, however, can choose a crs^{aux} such that it is in the image of the pseudo-random generator. Conveniently, a witness-indistinguishable argument is always multi-theorem, so that we now have a multi-theorem zero-knowledge argument.

4.2 *Single-to-Multi-Theorem Transformations for Non-Interactive Statistical Zero-Knowledge*

Marc Fischlin
Felix Rohrbach

Published at PKC 2021
Publication 4

While the Feige-Lapidot-Shamir (FLS) transformation works for *computational* zero-knowledge, it does not extend to *statistical* (or *perfect*) zero-knowledge: As the simulator uses an image of G for crs^{aux} , the common random string in the simulation is only computationally indistinguishable from a real execution, but not statistically indistinguishable.

Therefore, our goal of this paper is to provide a similar transformation for statistical zero-knowledge arguments in the common random string model. We give transformations, one based on one-way permutations and one based on the LWE assumption. As a second contribution, we analyse different soundness definitions of non-interactive zero-knowledge arguments.

Note that we are not the first to present such transformations for statistical zero-knowledge in general: Indeed, there exists a folklore⁶ version of the FLS transform which works for statistical zero-knowledge, which, however, only works in the common *reference* string model, i.e., where the string is not uniformly random, while our constructions work in the common random string model. There exists one previous construction of a multi-theorem non-interactive statistical zero-knowledge argument in the common random string model by Libert et al. [LPWW20], but this construction only works for non-adaptive soundness and is not a transformation.

⁶See Appendix A of our paper for a presentation of the folklore result in our notation

4.2.1 Multi-Theorem from One-Way Permutations

Our first transformation for non-interactive statistical zero-knowledge arguments is based on one-way permutations and uses a dual approach to the FLS transformation: Instead of building the OR-language by saying $x \in L$ or $\text{crs}^{\text{aux}} \in G(\{0, 1\}^n)$, we replace the latter by $\text{crs}^{\text{aux}} \notin G(\{0, 1\}^n)$. As this is a co-NP relation, we use the Blum-Micali-Yao [Yao82; BM84] pseudo-random generator from a one-way permutation f :

$$G(s) = f^{|s|}(s) \parallel \text{hb}(s) \parallel \text{hb}(f(s)) \parallel \dots \parallel \text{hb}(f^{|s|-1}(s))$$

Note that with this pseudo-random generator, it is easy to show that crs^{aux} is not in the image of G by providing a seed s for which the output of $f^{|s|}(s)$ matches, but at least one of the hard-core bits does not match.

For the simulator, we now choose our common random string as

$$\text{crs}^{\text{aux}} \leftarrow G(s) \oplus 0^{|s|} \parallel t,$$

where t is chosen uniformly at random. If $t = 0^{|s|}$, then the simulator aborts, but this only happens with probability $2^{-|s|}$. Therefore, the crs^{aux} generated by the simulator is statistically close to a real execution of the protocol, giving us *statistical* zero-knowledge.

For the malicious prover, the crs^{aux} is also not in the image of G with high probability, but it has no way of proving this, as it is computationally bounded and G is pseudo-random. We show this by replacing the crs^{aux} for the prover by $G(s)$ for a random s in a game hop.

Otherwise, the transformation follows the idea of the FLS transform. We note that we can even extend this construction to perfect zero-knowledge, if we allow the simulation to run in expected polynomial time.

4.2.2 Multi-Theorem from LWE

Our second transformation for non-interactive statistical zero-knowledge arguments is based on the LWE assumption, motivated by a recent construction of single-theorem statistical zero-knowledge arguments from the LWE assumption by Peikert and Shiehian [PS19]. For this transformation, we use a *dual-mode commitment scheme*, i.e., a commitment scheme that can either be statistically-hiding or perfectly-binding, with both modes being indistinguishable:

Definition 4.2.1 (Dual-Mode Commitment Scheme). A non-interactive commitment scheme

$$\Gamma = (\text{Gen}_H, \text{Gen}_B, \text{Com})$$

is called a *dual-mode commitment scheme* if,

Statistically-Hiding Mode: The scheme $(\text{Gen}_H, \text{Com})$ is a statistically-hiding, computationally-binding commitment scheme. Furthermore, the output of Gen_H is statistically close to the uniform distribution.

Perfectly-Binding Mode: The scheme $(\text{Gen}_B, \text{Com})$ is a perfectly-binding, computationally-hiding commitment scheme.

Indistinguishability of Modes: The random variables Gen_H and Gen_B are computationally indistinguishable.

We will use two trapdoor functions defined by Gorbunov et al. [GVW15], that, as pointed out by Couteau and Hofheinz [CH19], can be used as a dual-mode commitment scheme based on the LWE assumption. Conveniently, the generation of the public key for the statistically-hiding mode consists of just sampling a random matrix A uniformly.

Now, we use this dual-mode commitment scheme to build a transformation for statistical zero-knowledge in the common random string model, following again the idea of the FLS transform. We sample crs^{aux} uniformly at random and interpret it as a public key pk plus a commitment c : $(\text{pk}, c) \leftarrow \text{crs}^{\text{aux}}$. Now, our OR-language L' is based on whether c is a commitment to 1: $x \in L$ or $\exists U : c = \text{Com}(\text{pk}, 1; U)$.

The simulator generates a public key $\text{pk} \leftarrow \text{Gen}_H$, computes a commitment to 1 as $c \leftarrow \text{Com}(\text{pk}, 1)$ and sets $\text{crs}^{\text{aux}} \leftarrow (\text{pk}, c)$. Now, because the pk in hiding mode is just a uniformly random matrix and the commitment is statistically hiding, crs^{aux} is statistically indistinguishable from a uniformly random crs^{aux} .

For the malicious prover, we can use multiple game hops to replace pk with a public key in *binding* mode and then replace c with a commitment to 0. As the commitment scheme is binding, there can be no witness for it being a commitment to 1, which forces any malicious prover to break the soundness of the underlying scheme. The rest of the proof now follows again the idea of the FLS transform.

4.2.3 Soundness of Non-Interactive Zero-Knowledge Arguments

Finally, we take a closer look at the soundness of non-interactive zero-knowledge arguments. Commonly, soundness is divided into two categories: *Adaptive* soundness, in which the malicious prover gets to decide for which $x \notin L$ it wants to fool the verifier on after seeing the crs, while for *non-adaptive* soundness, it has to commit to an $x \notin L$ before seeing the crs. We argue that there is another dimension that has been neglected in previous analysis, namely what happens if the malicious prover outputs an $x \in L$: The first option is to let the malicious prover fail the security game if it outputs an $x \in L$, which we call *penalizing* soundness. The second option is to restrict the game to malicious provers that always output $x \notin L$, which we call *exclusive* soundness. Indeed, both penalizing as well as exclusive soundness exists in literature. In total, we get five definitions: All combinations of adaptive, non-adaptive and exclusive, penalizing plus a fifth definition which we call non-adaptive/non-uniform (see Figure 3.4 in this paper for an overview of the definitions). A similar definition appeared in a concurrent work by Arte and Bellare [AB20].

We argue that this additional soundness dimension, penalizing or exclusive, matters for analysing non-interactive zero-knowledge arguments: In the exclusive case, the malicious prover knows that $x \notin L$, while for penalizing soundness, they might not. And indeed, this plays an important role in Pass' impossibility result [Pas16] ruling out black-box constructions of adaptively-sound non-interactive statistical zero-knowledge arguments from standard assumptions: The malicious prover here selects a statement x for which it does not know whether it is in L or not. Therefore, Pass' impossibility result only rules out adaptive/penalizing soundness, not the slightly weaker adaptive/exclusive soundness.

Further, we claim adaptive/exclusive soundness is already a useful property: Indeed, we show that the notion of adaptive/culpable soundness follows from adaptive/exclusive soundness, which has been introduced by Groth et al. [GOS06], and which suffices already for many applications like

universally composable non-interactive zero-knowledge arguments.

4.2.4 My Contributions

This paper is a joined work by Marc Fischlin and me. The main ideas of the paper were discussed in regular meetings and were written up jointly, and are therefore owed to all authors equally. I took the lead in writing and proving the LWE-based construction. Further, in the soundness section, I wrote the proof for the equivalence of the non-adaptive soundness notions.

4.2.5 Versions

This paper has been published at PKC 2021 [FR21]. Previously, a version had already been uploaded to ePrint in October 2020, which then has been updated when the conference version was published [FR20]. The version in this thesis is content-wise identical to the peer-reviewed version.

4.3 Random Oracles and Variations

The random oracle model, first used by Fiat and Shamir [FS87] and later formalized by Bellare and Rogaway [BR93], is probably the most frequently used idealized model in cryptography, replacing a hash function by a truly random function. Security proofs in this model have given heuristic security arguments for protocols that are hard to prove in the standard model.

One example in [BR93] are signatures from a trapdoor permutation, i.e., a permutation $f(\mathbf{pk}, \cdot)$ which everyone can compute using the public key, but the inverse $f^{-1}(\mathbf{sk}, \cdot)$ can only be computed using the secret key \mathbf{sk} . Now, to sign a message m , we would use a hash function H to hash the message and then apply the trapdoor to it:

$$\sigma \leftarrow f^{-1}(\mathbf{sk}, H(m))$$

In practice, this would be for example RSA signatures with full domain hash.

Now, we want to prove security of this signature scheme in the random oracle model, assuming that the trapdoor permutation is secure. We do this via a reduction to the one-wayness of the permutation: For the public key \mathbf{pk} for a trapdoor permutation and a random y in the image of it, we are supposed to compute x such that $f(\mathbf{pk}, x) = y$, given a successful adversary against the signature algorithm. For simplicity, assume that the adversary has to come up with a forgery without knowing any signatures. Then, the reduction will see a number of calls to the random oracle, followed by the adversary outputting a message together with a forgery of a signature. We can assume that the adversary has queried the message for which it creates a valid forgery to the random oracle. Therefore, the reduction modifies the answer of the random oracle for one query (chosen at random) to be y – as y is chosen randomly as well, this is unnoticeable by the adversary. This, however, means that the adversary now produces the preimage of y with polynomial probability, giving us a successful adversary against the trapdoor permutation.

While this is a toy example of how to prove statements in the random oracle model, it highlights two main techniques that are very useful in reductions: The reduction can observe queries of the underlying adversary to the random oracle, and it can modify them (as long as they still look random).

There are a number of variants of the random oracle model. The Auxiliary-Input Random Oracle Model (short AI-ROM), introduced by Unruh [Unr07], models random oracles for *non-uniform* adversaries that might have some non-uniform knowledge about the hash function in the standard model. One example here would be an unkeyed hash function, for which a non-uniform adversary might know a fixed collision in the function, breaking its collision resistance. To model this non-uniform knowledge, in the AI-ROM, the adversary gets access to the output of $z^{\mathcal{O}}(1^n)$, an arbitrary unbounded function with access to the oracle, but which does not have access to any other parameters of the game.

The global random oracle is another variant by Canetti et al. [CJS14] and later Camenisch et al. [CDGLN18], which focuses on the use of random oracles in the Universal Composability (UC) framework: If two components are proven UC-secure in the (normal) random oracle model, their composition is not necessarily secure if the random oracle is replaced with the same hash function for both components. The global random oracle model now says that a single, global random oracle functionality is available to all parties, and can therefore also be replaced with the same hash function everywhere.

4.4 A Random Oracle for All of Us

Marc Fischlin

Felix Rohrbach

Tobias Schmalz

Published at **AFRICACRYPT 2022**

Publication 5

In this paper, we present another variant of the ROM, the Universal Random Oracle Model, short UROM. Our motivation originates from a difference between how we use hash functions in the real world and how random oracles are modeled: Hash functions are standardized and in most cases fixed in a protocol description, while in the random oracle model, a new random oracle is drawn at random for each security game and adversary. For our UROM, we therefore want to once choose a universal random oracle, which is then used for all adversaries.

On a technical level, this is a question of order of quantifiers: Do we first sample a random oracle, and then say that for all adversaries security of some game holds (with high probability), or do we say that for all adversaries, the security of this game holds (with high probability) for a randomly chosen oracle? This difference can be overcome by using the Borel-Cantelli lemma, as done for example by Impagliazzo and Rudich [IR89]. However, this generic transformation from the ROM into the UROM has two downsides: First, it only works for uniform adversaries, as the Borel-Cantelli lemma requires the number of adversaries to be countable, and secondly, the asymptotic nature of the Borel-Cantelli lemma does not give us a tight security bound, infringing with the notion of concrete security.

4.4.1 Defining the UROM

We now give our definition of the universal random oracle model. For this, we denote by $\text{Game}^{\mathcal{A}, \mathcal{O}}$ a (probabilistic) security game for some security notion relative to the random oracle \mathcal{O} , which interacts with the adversary, and which outputs 0 or 1. If the security game $\text{Game}^{\mathcal{A}, \mathcal{O}}$ outputs 1, the adversary has been successful in this instance, and for a security notion to hold, this should happen with small probability.

Definition 4.4.1 (UROM). A security game Game is secure in the UROM if

$$\forall s \in \text{poly} \exists \varepsilon_s \in \text{negl} \forall \lambda \in \mathbb{N} : \Pr_{\mathcal{O}} \left[\forall \mathcal{A}_{\mathcal{O}} \in \text{SIZE}(s(\lambda)) : \Pr_{\text{Game}} \left[\text{Game}^{\mathcal{A}_{\mathcal{O}}, \mathcal{O}}(\lambda) \right] \leq \varepsilon_s(\lambda) \right] \geq 1 - \varepsilon_s(\lambda).$$

As described above, we now have one random oracle chosen for all adversaries. This means we have to split the randomness of sampling the oracle and sampling the randomness used by the game, resulting in the nested probability we have in the definition. We also use the negligible function ε_s twice: The inner use accounts for good guesses of the adversary in the game, while the outer use guards against “bad” oracles, i.e., if we happen to sample the all-zero random oracle, the game is probably not secure. The negligible function also depends on the maximal size of the adversary, as for example an adversary which includes many hard-coded preimages of the random oracle might affect a one-wayness game.

Defining the universal random oracle model the “correct” way was more difficult than we anticipated. Indeed, for an early attempt, we required that for all but a zero-measure of oracles, the game should be secure:

$$\Pr_{\mathcal{O}} \left[\begin{array}{l} \forall \mathcal{A}_{\mathcal{O}} \in \text{SIZE}(\text{poly}(\lambda)) \\ \exists \varepsilon_{\mathcal{A}, \mathcal{O}} \in \text{negl} \forall \lambda \in \mathbb{N} \end{array} : \Pr_{\text{Game}} \left[\text{Game}^{\mathcal{A}_{\mathcal{O}}, \mathcal{O}}(\lambda) \right] \leq \varepsilon_{\mathcal{A}, \mathcal{O}}(\lambda) \right] = 1.$$

However, maybe surprisingly, this definition is not sound: Indeed, there exists a game that is secure in this model, but is both intuitively insecure, as well as insecure in the ROM. For this, let us consider an adversary that wins with probability $\frac{1}{\lambda^2}$ over the choice of the random oracle. As this adversary wins with polynomial probability, we would consider the game insecure. However, the probability that such an adversary wins for infinitely many security parameters is zero — meaning that the game would be secure in the UROM. For the full example, see Appendix A of the paper.

4.4.2 UROM \Leftrightarrow AI-ROM

Next, we show that the UROM and the AI-ROM are equivalent. However, only the direction from UROM security to AI-ROM security is tight — therefore, a proof in the AI-ROM might result in looser security bound than a proof in the UROM. Comparing this to the Borel-Cantelli lemma, we now have a generic transformation from the AI-ROM to the UROM which now of course also works for non-uniform adversaries, but which still has the problem of not being tight.

Theorem 4.4.2 (AI-ROM \Rightarrow UROM). *If Game is secure in the AI-ROM then it is also secure in the UROM.*

Theorem 4.4.3 (UROM \Rightarrow AI-ROM, tightly). *If Game is secure in the UROM then it is also secure in the AI-ROM, with a tight security reduction.*

4.4.3 One-way functions exist in UROM

Finally, we prove that one-way functions exist in the UROM. We are using the compression technique originally introduced by Gennaro and Trevisan [GT00], which assumes the existence of a successful adversary, and then shows that the random oracle can be compressed beyond information-theoretic limits. Similar results exist for the AI-ROM [DGK17; CDGS18], which would by the previous section of course also apply to the UROM, but the idea here is to have a direct proof in the UROM. Further, this result also shows the flexibility of the UROM in choosing different bounds for the inner and the outer probability.

Theorem 4.4.4. *Let S be the maximum size of an adversary. Then, in the Universal Random Oracle Model, the random oracle is a one-way function with security bounds $\frac{1}{P}$ and $2^{-\lambda}$ for the inner and outer probability, under the condition that $P \cdot S \leq 2^{\lambda/4}$ and $\lambda > 55$:*

$$\Pr_{\mathcal{O}} \left[\exists \mathcal{A}_{\mathcal{O}} \in \text{SIZE}(S) : \Pr_{x \leftarrow \{0,1\}^{\lambda}} [\mathcal{A}_{\mathcal{O}}^{\mathcal{O}}(1^{\lambda}, \mathcal{O}(x)) \in \mathcal{O}^{-1}(\mathcal{O}(x))] > \frac{1}{P} \right] < 2^{-\lambda}$$

Unfortunately, we do not (yet) have an equivalent to bit-fixing, a proof technique used by Unruh [Unr07] to prove OAEP secure in the AI-ROM.

4.4.4 My Contributions

This paper is a joint work by Marc Fischlin, Tobias Schmalz and me. The main ideas of the paper were discussed in regular meetings and were written up jointly, and are therefore owed to all authors equally. Especially the definition of the UROM, which took many iterations to arrive at the final definition, was jointly created in these discussions. In the section showing the equivalence of UROM and AI-ROM, I contributed to both directions of the proof. Section 5 (Universal Random Oracles are One-way Functions) was written by me.

4.4.5 Versions

This paper was published at AFRICACRYPT 2022 [FRS22b]. The paper was afterwards also uploaded to ePrint [FRS22a] with some layout changes. The version in this paper is content-wise identical to the peer-reviewed version.

4.5 Contributions of This Chapter

In this chapter, I have presented two instances of constructively using idealized models. In the first paper, we give two constructions of a single-to-multi-theorem transformation for statistical zero-knowledge proofs in the common random string model. This transformation allows to use the same common random string for any polynomial number of statements to be proven. Further, we analyse different types of soundness relative to the common random string, and show that the difference between *exclusive* and *penalizing* soundness can be used to circumvent impossibility results.

In the second paper, we present the universal random oracle model: A new idealized model which maps more closely how we use hash functions (which we idealize by random oracles). We show that our new model is asymptotically equivalent to the auxiliary-input random oracle model, but while security in the AI-ROM follows from security in the UROM tightly, the implication in the other direction is not tight. Therefore, it might be possible to prove better concrete security in the UROM compared to the AI-ROM.

5 Conclusion & Outlook

In this thesis, I have looked at two applications of oracles in cryptography: First, how they are used in oracle separations to prove lower bounds, and secondly, how they are used in idealized models for constructive purposes. For oracle separations, I have presented three papers that highlight different facets of proving lower bounds: For example, the first lower bound targets the *efficiency* of constructions of strong from weak one-way functions, while the other lower bounds target the general existence of constructions from certain assumptions. Further, the first two separations are Simon-style separations between symmetric primitives, while the last paper places extremely lossy functions somewhere between the classical set of symmetric primitives and public-key cryptography. For idealized models, I presented a construction relative to the common random string model, as well as a new variant of the random oracle model, which aims to bring the random oracle model a step closer to how we use hash functions in practice.

One topic that appears in both chapters is the random oracle controversy: The question whether the security of a proof in the random oracle model transfers to the standard model if we replace the random oracle by a cryptographically secure hash function, e.g., SHA3. The recent work by Khovratovich, Rothblum, and Soukhanov [KRS25] gave the debate a new perspective by showing an arguably uncontrived counterexample for a specific use of the Fiat-Shamir transform, and might incentivize more research into alternatives to the random oracle model. This thesis discusses to different approaches in this context: The first one is to replace the random oracle model with an assumption that arguably exists in the standard model, in this case extremely lossy functions. The second one is to replace the random oracle model with a model that more closely resembles the use of hash functions in practice, which in this case is the universal random oracle model.

In general, of course, it is more desirable to have a proof with an assumption that reasonably holds, compared to a proof in an idealized model which cannot exist in reality. However, I see having different approaches available as important: While many applications have been found for extremely lossy functions where they can replace random oracles, ELF's are unlikely to work for every usage of random oracles. Therefore, in these cases, switching to a model that resembles the reality more closely (while staying within an idealized model) might be the best option available. Also, while having a proof for a protocol in the standard model is valuable, due to efficiency reasons, we will usually still use the protocol with SHA3 instead of replacing SHA3 with an ELF — so even in this case, the security of the protocol is still heuristic.

This thesis points to multiple directions for future research. Strengthening the lower bounds by attenuating the restrictions is an interesting endeavour. For the oracle separation between weak and strong one-way functions, it would be very interesting to extend this result to constructions with compressing post-processing. As already mentioned, this, however, is unlikely to work with our current set of oracles, as the random oracle F removes the correlation that might exist between the

inputs — therefore, if we want to invert a construction with compressing post-processing, we don't see this correlation and therefore don't even know for which values we have to invert F . Therefore, this would require some kind of oracle which takes this correlation into account, or some fundamentally different approach.

For the oracle separation between distributional and existential collision resistance, removing the single-query restriction, i.e., that only the first query is allowed to access the breaker oracle, and only afterwards the F -oracle can be queried, would strengthen the lower bound significantly. However, it is unclear how to prove such a separation, as our current one relies on h being chosen independently of F , which would not be the case if the adversary can construct h based on previous F -queries.

In general, it would be interesting to remove the two-oracle technique from Simon-style separations to not only exclude *fully* black-box reductions. However, then we would have to deal with recursive calls to the breaker oracle, for which no good framework exists yet. To develop such a framework, it might be interesting to see whether Simon's [Sim98] proof (which does not use the two-oracle technique) can be generalized and applied to a wider range of breaker oracles.

For extremely lossy functions, finding new assumptions from which to build them, and then maybe constructing post-quantum secure extremely lossy functions would be very exciting and would strengthen the role of ELF's as a replacement for random oracles. A first step for this would be to identify post-quantum assumptions that hold with exponential security as required by extremely lossy functions. Further, understanding the exact relationship to other assumptions, such as hard problems in the class SZK, would be helpful.

Finally, the universal random oracle model would benefit from further methods to prove security relative to the model. In the auxiliary-input random oracle model, Unruh [Unr07] shows that we can fix the random oracle on a fraction of its input based on the auxiliary input, and the rest of the random oracle is then nearly independent of the auxiliary input. Then, we can use techniques used relative to the random oracle model, such as lazy sampling, on the rest of the oracle. It would be interesting to see whether a similar technique can be developed for the universal random oracle model.

Bibliography

- [AB13] Prabhanjan Ananth and Raghav Bhaskar. “Non Observability in the Random Oracle Model”. In: *Provable Security*. Ed. by Willy Susilo and Reza Reyhanitabar. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 86–103. ISBN: 978-3-642-41227-1.
- [AB20] Vivek Arte and Mihir Bellare. “Dual-Mode NIZKs: Possibility and Impossibility Results for Property Transfer”. In: *INDOCRYPT 2020*. Ed. by Karthikeyan Bhargavan, Elisabeth Oswald, and Manoj Prabhakaran. Vol. 12578. LNCS. Springer, Cham, Dec. 2020, pp. 859–881. DOI: [10.1007/978-3-030-65277-7_38](https://doi.org/10.1007/978-3-030-65277-7_38).
- [ACH20] Thomas Agrikola, Geoffroy Couteau, and Dennis Hofheinz. “The Usefulness of Sparsifiable Inputs: How to Avoid Subexponential iO”. In: *PKC 2020, Part I*. Ed. by Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas. Vol. 12110. LNCS. Springer, Cham, May 2020, pp. 187–219. DOI: [10.1007/978-3-030-45374-9_7](https://doi.org/10.1007/978-3-030-45374-9_7).
- [AS15] Gilad Asharov and Gil Segev. “Limits on the Power of Indistinguishability Obfuscation and Functional Encryption”. In: *56th FOCS*. Ed. by Venkatesan Guruswami. IEEE Computer Society Press, Oct. 2015, pp. 191–209. DOI: [10.1109/FOCS.2015.21](https://doi.org/10.1109/FOCS.2015.21).
- [AW08] Scott Aaronson and Avi Wigderson. “Algebrization: a new barrier in complexity theory”. In: *40th ACM STOC*. Ed. by Richard E. Ladner and Cynthia Dwork. ACM Press, May 2008, pp. 731–740. DOI: [10.1145/1374376.1374481](https://doi.org/10.1145/1374376.1374481).
- [Bar01] Boaz Barak. “How to Go Beyond the Black-Box Simulation Barrier”. In: *42nd FOCS*. IEEE Computer Society Press, Oct. 2001, pp. 106–115. DOI: [10.1109/SFCS.2001.959885](https://doi.org/10.1109/SFCS.2001.959885).
- [BBF13] Paul Baecher, Christina Brzuska, and Marc Fischlin. “Notions of Black-Box Reductions, Revisited”. In: *ASIACRYPT 2013, Part I*. Ed. by Kazue Sako and Palash Sarkar. Vol. 8269. LNCS. Springer, Berlin, Heidelberg, Dec. 2013, pp. 296–315. DOI: [10.1007/978-3-642-42033-7_16](https://doi.org/10.1007/978-3-642-42033-7_16).
- [BCEKM24] Chris Brzuska, Geoffroy Couteau, Christoph Egger, Pihla Karanko, and Pierre Meyer. “Instantiating the Hash-Then-Evaluate Paradigm: Strengthening PRFs, PCFs, and OPRFs”. In: *SCN 24, Part II*. Ed. by Clemente Galdi and Duong Hieu Phan. Vol. 14974. LNCS. Springer, Cham, Sept. 2024, pp. 97–116. DOI: [10.1007/978-3-031-71073-5_5](https://doi.org/10.1007/978-3-031-71073-5_5).

- [BCKR21] Chris Brzuska, Geoffroy Couteau, Pihla Karanko, and Felix Rohrbach. “On Derandomizing Yao’s Weak-to-Strong OWF Construction”. In: *TCC 2021, Part II*. Ed. by Kobbi Nissim and Brent Waters. Vol. 13043. LNCS. Springer, Cham, Nov. 2021, pp. 429–456. DOI: [10.1007/978-3-030-90453-1_15](https://doi.org/10.1007/978-3-030-90453-1_15).
- [BCKR23] Chris Brzuska, Geoffroy Couteau, Pihla Karanko, and Felix Rohrbach. *On Derandomizing Yao’s Weak-to-Strong OWF Construction*. Cryptology ePrint Archive, Report 2023/1091. 2023. URL: <https://eprint.iacr.org/2023/1091>.
- [BFM14] Christina Brzuska, Pooya Farshim, and Arno Mittelbach. “Indistinguishability Obfuscation and UCEs: The Case of Computationally Unpredictable Sources”. In: *CRYPTO 2014, Part I*. Ed. by Juan A. Garay and Rosario Gennaro. Vol. 8616. LNCS. Springer, Berlin, Heidelberg, Aug. 2014, pp. 188–205. DOI: [10.1007/978-3-662-44371-2_11](https://doi.org/10.1007/978-3-662-44371-2_11).
- [BFM88] Manuel Blum, Paul Feldman, and Silvio Micali. “Non-Interactive Zero-Knowledge and Its Applications (Extended Abstract)”. In: *20th ACM STOC*. ACM Press, May 1988, pp. 103–112. DOI: [10.1145/62212.62222](https://doi.org/10.1145/62212.62222).
- [BFM90] Manuel Blum, Paul Feldman, and Silvio Micali. “Proving Security Against Chosen Cyphertext Attacks”. In: *CRYPTO’88*. Ed. by Shafi Goldwasser. Vol. 403. LNCS. Springer, New York, Aug. 1990, pp. 256–268. DOI: [10.1007/0-387-34799-2_20](https://doi.org/10.1007/0-387-34799-2_20).
- [BG81] Charles H. Bennett and John Gill. “Relative to a Random Oracle A , $\mathcal{P}^A \neq \mathcal{NP}^A \neq \text{co-}\mathcal{NP}^A$ with Probability 1”. In: *SIAM J. Comput.* 10.1 (Feb. 1981), pp. 96–113. ISSN: 0097-5397. DOI: [10.1137/0210008](https://doi.org/10.1137/0210008). URL: <https://doi.org/10.1137/0210008>.
- [BGS75] Theodore Baker, John Gill, and Robert Solovay. “Relativizations of the $\mathcal{P} = ?\mathcal{NP}$ Question”. In: *SIAM Journal on Computing* 4.4 (1975), pp. 431–442. DOI: [10.1137/0204037](https://doi.org/10.1137/0204037). URL: <https://doi.org/10.1137/0204037>.
- [BHK13] Mihir Bellare, Viet Tung Hoang, and Sriram Keelveedhi. “Instantiating Random Oracles via UCEs”. In: *CRYPTO 2013, Part II*. Ed. by Ran Canetti and Juan A. Garay. Vol. 8043. LNCS. Springer, Berlin, Heidelberg, Aug. 2013, pp. 398–415. DOI: [10.1007/978-3-642-40084-1_23](https://doi.org/10.1007/978-3-642-40084-1_23).
- [BHKY19] Nir Bitansky, Iftach Haitner, Ilan Komargodski, and Eylon Yogev. “Distributional Collision Resistance Beyond One-Way Functions”. In: *EUROCRYPT 2019, Part III*. Ed. by Yuval Ishai and Vincent Rijmen. Vol. 11478. LNCS. Springer, Cham, May 2019, pp. 667–695. DOI: [10.1007/978-3-030-17659-4_23](https://doi.org/10.1007/978-3-030-17659-4_23).
- [Ble98] Daniel Bleichenbacher. “Chosen Ciphertext Attacks Against Protocols Based on the RSA Encryption Standard PKCS #1”. In: *CRYPTO’98*. Ed. by Hugo Krawczyk. Vol. 1462. LNCS. Springer, Berlin, Heidelberg, Aug. 1998, pp. 1–12. DOI: [10.1007/BFb0055716](https://doi.org/10.1007/BFb0055716).
- [BM84] Manuel Blum and Silvio Micali. “How to Generate Cryptographically Strong Sequences of Pseudo-Random Bits”. In: *SIAM J. Comput.* 13.4 (1984), pp. 850–864. DOI: [10.1137/0213053](https://doi.org/10.1137/0213053). URL: <https://doi.org/10.1137/0213053>.

- [BMM24] Amos Beimel, Tal Malkin, and Noam Mazor. “Structural Lower Bounds on Black-Box Constructions of Pseudorandom Functions”. In: *CRYPTO 2024, Part V*. Ed. by Leonid Reyzin and Douglas Stebila. Vol. 14924. LNCS. Springer, Cham, Aug. 2024, pp. 459–488. DOI: [10.1007/978-3-031-68388-6_16](https://doi.org/10.1007/978-3-031-68388-6_16).
- [BR93] Mihir Bellare and Phillip Rogaway. “Random Oracles are Practical: A Paradigm for Designing Efficient Protocols”. In: *ACM CCS 93*. Ed. by Dorothy E. Denning, Raymond Pyle, Ravi Ganesan, Ravi S. Sandhu, and Victoria Ashby. ACM Press, Nov. 1993, pp. 62–73. DOI: [10.1145/168588.168596](https://doi.org/10.1145/168588.168596).
- [BST16] Mihir Bellare, Igors Stepanovs, and Stefano Tessaro. “Contention in Cryptoland: Obfuscation, Leakage and UCE”. In: *TCC 2016-A, Part II*. Ed. by Eyal Kushilevitz and Tal Malkin. Vol. 9563. LNCS. Springer, Berlin, Heidelberg, Jan. 2016, pp. 542–564. DOI: [10.1007/978-3-662-49099-0_20](https://doi.org/10.1007/978-3-662-49099-0_20).
- [CCR16] Ran Canetti, Yilei Chen, and Leonid Reyzin. “On the Correlation Intractability of Obfuscated Pseudorandom Functions”. In: *TCC 2016-A, Part I*. Ed. by Eyal Kushilevitz and Tal Malkin. Vol. 9562. LNCS. Springer, Berlin, Heidelberg, Jan. 2016, pp. 389–415. DOI: [10.1007/978-3-662-49096-9_17](https://doi.org/10.1007/978-3-662-49096-9_17).
- [CDGLN18] Jan Camenisch, Manu Drijvers, Tommaso Gagliardoni, Anja Lehmann, and Gregory Neven. “The Wonderful World of Global Random Oracles”. In: *EUROCRYPT 2018, Part I*. Ed. by Jesper Buus Nielsen and Vincent Rijmen. Vol. 10820. LNCS. Springer, Cham, Apr. 2018, pp. 280–312. DOI: [10.1007/978-3-319-78381-9_11](https://doi.org/10.1007/978-3-319-78381-9_11).
- [CDGS18] Sandro Coretti, Yevgeniy Dodis, Siyao Guo, and John P. Steinberger. “Random Oracles and Non-uniformity”. In: *EUROCRYPT 2018, Part I*. Ed. by Jesper Buus Nielsen and Vincent Rijmen. Vol. 10820. LNCS. Springer, Cham, Apr. 2018, pp. 227–258. DOI: [10.1007/978-3-319-78381-9_9](https://doi.org/10.1007/978-3-319-78381-9_9).
- [CGH98] Ran Canetti, Oded Goldreich, and Shai Halevi. “The Random Oracle Methodology, Revisited (Preliminary Version)”. In: *30th ACM STOC*. ACM Press, May 1998, pp. 209–218. DOI: [10.1145/276698.276741](https://doi.org/10.1145/276698.276741).
- [CH19] Geoffroy Couteau and Dennis Hofheinz. “Designated-Verifier Pseudorandom Generators, and Their Applications”. In: *EUROCRYPT 2019, Part II*. Ed. by Yuval Ishai and Vincent Rijmen. Vol. 11477. LNCS. Springer, Cham, May 2019, pp. 562–592. DOI: [10.1007/978-3-030-17656-3_20](https://doi.org/10.1007/978-3-030-17656-3_20).
- [CJS14] Ran Canetti, Abhishek Jain, and Alessandra Scafuro. “Practical UC security with a Global Random Oracle”. In: *ACM CCS 2014*. Ed. by Gail-Joon Ahn, Moti Yung, and Ninghui Li. ACM Press, Nov. 2014, pp. 597–608. DOI: [10.1145/2660267.2660374](https://doi.org/10.1145/2660267.2660374).
- [DGK17] Yevgeniy Dodis, Siyao Guo, and Jonathan Katz. “Fixing Cracks in the Concrete: Random Oracles with Auxiliary Input, Revisited”. In: *EUROCRYPT 2017, Part II*. Ed. by Jean-Sébastien Coron and Jesper Buus Nielsen. Vol. 10211. LNCS. Springer, Cham, Apr. 2017, pp. 473–495. DOI: [10.1007/978-3-319-56614-6_16](https://doi.org/10.1007/978-3-319-56614-6_16).
- [DI06] Bella Dubrov and Yuval Ishai. “On the randomness complexity of efficient sampling”. In: *38th ACM STOC*. Ed. by Jon M. Kleinberg. ACM Press, May 2006, pp. 711–720. DOI: [10.1145/1132516.1132615](https://doi.org/10.1145/1132516.1132615).

- [FGHMY24] Lukás Folwarczný, Mika Göös, Pavel Hubáček, Gilbert Maystre, and Weiqiang Yuan. “One-Way Functions vs. TFNP: Simpler and Improved”. In: *ITCS 2024*. Ed. by Venkatesan Guruswami. Vol. 287. LIPIcs, Jan. 2024, 50:1–50:14. DOI: [10.4230/LIPIcs.ITCS.2024.50](https://doi.org/10.4230/LIPIcs.ITCS.2024.50).
- [FLRSST10] Marc Fischlin, Anja Lehmann, Thomas Ristenpart, Thomas Shrimpton, Martijn Stam, and Stefano Tessaro. “Random Oracles with(out) Programmability”. In: *ASIACRYPT 2010*. Ed. by Masayuki Abe. Vol. 6477. LNCS. Springer, Berlin, Heidelberg, Dec. 2010, pp. 303–320. DOI: [10.1007/978-3-642-17373-8_18](https://doi.org/10.1007/978-3-642-17373-8_18).
- [FLS90] Uriel Feige, Dror Lapidot, and Adi Shamir. “Multiple Non-Interactive Zero Knowledge Proofs Based on a Single Random String (Extended Abstract)”. In: *31st FOCS*. IEEE Computer Society Press, Oct. 1990, pp. 308–317. DOI: [10.1109/FSCS.1990.89549](https://doi.org/10.1109/FSCS.1990.89549).
- [FR20] Marc Fischlin and Felix Rohrbach. *Single-to-Multi-Theorem Transformations for Non-Interactive Statistical Zero-Knowledge*. Cryptology ePrint Archive, Report 2020/1204. 2020. URL: <https://eprint.iacr.org/2020/1204>.
- [FR21] Marc Fischlin and Felix Rohrbach. “Single-to-Multi-theorem Transformations for Non-interactive Statistical Zero-Knowledge”. In: *PKC 2021, Part II*. Ed. by Juan Garay. Vol. 12711. LNCS. Springer, Cham, May 2021, pp. 205–234. DOI: [10.1007/978-3-030-75248-4_8](https://doi.org/10.1007/978-3-030-75248-4_8).
- [FR23a] Marc Fischlin and Felix Rohrbach. *Searching for ELFs in the Cryptographic Forest*. Cryptology ePrint Archive, Report 2023/1403. 2023. URL: <https://eprint.iacr.org/2023/1403>.
- [FR23b] Marc Fischlin and Felix Rohrbach. “Searching for ELFs in the Cryptographic Forest”. In: *TCC 2023, Part III*. Ed. by Guy N. Rothblum and Hoeteck Wee. Vol. 14371. LNCS. Springer, Cham, Nov. 2023, pp. 207–236. DOI: [10.1007/978-3-031-48621-0_8](https://doi.org/10.1007/978-3-031-48621-0_8).
- [FRS22a] Marc Fischlin, Felix Rohrbach, and Tobias Schmalz. *A Random Oracle for All of Us*. Cryptology ePrint Archive, Report 2022/906. 2022. URL: <https://eprint.iacr.org/2022/906>.
- [FRS22b] Marc Fischlin, Felix Rohrbach, and Tobias Schmalz. “A Random Oracle for All of Us”. In: *AFRICACRYPT 22*. Ed. by Lejla Batina and Joan Daemen. Vol. 2022. LNCS. Springer, Cham, July 2022, pp. 469–489. DOI: [10.1007/978-3-031-17433-9_20](https://doi.org/10.1007/978-3-031-17433-9_20).
- [FS87] Amos Fiat and Adi Shamir. “How to Prove Yourself: Practical Solutions to Identification and Signature Problems”. In: *CRYPTO’86*. Ed. by Andrew M. Odlyzko. Vol. 263. LNCS. Springer, Berlin, Heidelberg, Aug. 1987, pp. 186–194. DOI: [10.1007/3-540-47721-7_12](https://doi.org/10.1007/3-540-47721-7_12).
- [FS88] Lance Fortnow and Michael Sipser. “Are there interactive protocols for co-NP languages?” In: *Information Processing Letters* 28.5 (1988), pp. 249–251. ISSN: 0020-0190. DOI: [https://doi.org/10.1016/0020-0190\(88\)90199-8](https://doi.org/10.1016/0020-0190(88)90199-8). URL: <https://www.sciencedirect.com/science/article/pii/0020019088901998>.

- [FS90] Uriel Feige and Adi Shamir. “Witness Indistinguishable and Witness Hiding Protocols”. In: *22nd ACM STOC*. ACM Press, May 1990, pp. 416–426. DOI: [10.1145/100216.100272](https://doi.org/10.1145/100216.100272).
- [Gas19] William I. Gasarch. “Guest Column: The Third P =? NP Poll”. In: *SIGACT News Complexity Theory Column* 100 (2019). URL: <https://www.cs.umd.edu/users/gasarch/BLOGPAPERS/pollpaper3.pdf>.
- [GGK03] Rosario Gennaro, Yael Gertner, and Jonathan Katz. “Lower bounds on the efficiency of encryption and digital signature schemes”. In: *35th ACM STOC*. ACM Press, June 2003, pp. 417–425. DOI: [10.1145/780542.780604](https://doi.org/10.1145/780542.780604).
- [GILVZ90] Oded Goldreich, Russell Impagliazzo, Leonid A. Levin, Ramarathnam Venkatesan, and David Zuckerman. “Security Preserving Amplification of Hardness”. In: *31st FOCS*. IEEE Computer Society Press, Oct. 1990, pp. 318–326. DOI: [10.1109/SFCS.1990.89550](https://doi.org/10.1109/SFCS.1990.89550).
- [GK03] Shafi Goldwasser and Yael Tauman Kalai. “On the (In)security of the Fiat-Shamir Paradigm”. In: *44th FOCS*. IEEE Computer Society Press, Oct. 2003, pp. 102–115. DOI: [10.1109/SFCS.2003.1238185](https://doi.org/10.1109/SFCS.2003.1238185).
- [GKMRV00] Yael Gertner, Sampath Kannan, Tal Malkin, Omer Reingold, and Mahesh Viswanathan. “The Relationship between Public Key Encryption and Oblivious Transfer”. In: *41st FOCS*. IEEE Computer Society Press, Nov. 2000, pp. 325–335. DOI: [10.1109/SFCS.2000.892121](https://doi.org/10.1109/SFCS.2000.892121).
- [GMR85] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. “The Knowledge Complexity of Interactive Proof-Systems (Extended Abstract)”. In: *17th ACM STOC*. ACM Press, May 1985, pp. 291–304. DOI: [10.1145/22145.22178](https://doi.org/10.1145/22145.22178).
- [GOS06] Jens Groth, Rafail Ostrovsky, and Amit Sahai. “Perfect Non-interactive Zero Knowledge for NP”. In: *EUROCRYPT 2006*. Ed. by Serge Vaudenay. Vol. 4004. LNCS. Springer, Berlin, Heidelberg, May 2006, pp. 339–358. DOI: [10.1007/11761679_21](https://doi.org/10.1007/11761679_21).
- [GT00] Rosario Gennaro and Luca Trevisan. “Lower Bounds on the Efficiency of Generic Cryptographic Constructions”. In: *41st FOCS*. IEEE Computer Society Press, Nov. 2000, pp. 305–313. DOI: [10.1109/SFCS.2000.892119](https://doi.org/10.1109/SFCS.2000.892119).
- [GVW15] Sergey Gorbunov, Vinod Vaikuntanathan, and Daniel Wichs. “Leveled Fully Homomorphic Signatures from Standard Lattices”. In: *47th ACM STOC*. Ed. by Rocco A. Servedio and Ronitt Rubinfeld. ACM Press, June 2015, pp. 469–477. DOI: [10.1145/2746539.2746576](https://doi.org/10.1145/2746539.2746576).
- [HHR06] Iftach Haitner, Danny Harnik, and Omer Reingold. “On the Power of the Randomized Iterate”. In: *CRYPTO 2006*. Ed. by Cynthia Dwork. Vol. 4117. LNCS. Springer, Berlin, Heidelberg, Aug. 2006, pp. 22–40. DOI: [10.1007/11818175_2](https://doi.org/10.1007/11818175_2).
- [HHR07] Iftach Haitner, Jonathan J. Hoch, Omer Reingold, and Gil Segev. “Finding Collisions in Interactive Protocols - A Tight Lower Bound on the Round Complexity of Statistically-Hiding Commitments”. In: *48th FOCS*. IEEE Computer Society Press, Oct. 2007, pp. 669–679. DOI: [10.1109/FOCS.2007.27](https://doi.org/10.1109/FOCS.2007.27).

- [HK06] Shai Halevi and Hugo Krawczyk. “Strengthening Digital Signatures Via Randomized Hashing”. In: *CRYPTO 2006*. Ed. by Cynthia Dwork. Vol. 4117. LNCS. Springer, Berlin, Heidelberg, Aug. 2006, pp. 41–59. DOI: [10.1007/11818175_3](https://doi.org/10.1007/11818175_3).
- [HL18] Justin Holmgren and Alex Lombardi. “Cryptographic Hashing from Strong One-Way Functions (Or: One-Way Product Functions and Their Applications)”. In: *59th FOCS*. Ed. by Mikkel Thorup. IEEE Computer Society Press, Oct. 2018, pp. 850–858. DOI: [10.1109/FOCS.2018.00085](https://doi.org/10.1109/FOCS.2018.00085).
- [HR04] Chun-Yuan Hsiao and Leonid Reyzin. “Finding Collisions on a Public Road, or Do Secure Hash Functions Need Secret Coins?” In: *CRYPTO 2004*. Ed. by Matthew Franklin. Vol. 3152. LNCS. Springer, Berlin, Heidelberg, Aug. 2004, pp. 92–105. DOI: [10.1007/978-3-540-28628-8_6](https://doi.org/10.1007/978-3-540-28628-8_6).
- [IL89] Russell Impagliazzo and Michael Luby. “One-way Functions are Essential for Complexity Based Cryptography (Extended Abstract)”. In: *30th FOCS*. IEEE Computer Society Press, Oct. 1989, pp. 230–235. DOI: [10.1109/SFCS.1989.63483](https://doi.org/10.1109/SFCS.1989.63483).
- [Imp95] Russell Impagliazzo. “A Personal View of Average-Case Complexity”. In: *Proceedings of the Tenth Annual Structure in Complexity Theory Conference, Minneapolis, Minnesota, USA, June 19-22, 1995*. IEEE Computer Society, 1995, pp. 134–147. ISBN: 0-8186-7052-5. URL: <https://doi.org/10.1109/SCT.1995.514853>.
- [IR89] Russell Impagliazzo and Steven Rudich. “Limits on the Provable Consequences of One-Way Permutations”. In: *21st ACM STOC*. ACM Press, May 1989, pp. 44–61. DOI: [10.1145/73007.73012](https://doi.org/10.1145/73007.73012).
- [KRS25] Dmitry Khovratovich, Ron D. Rothblum, and Lev Soukhanov. *How to Prove False Statements: Practical Attacks on Fiat-Shamir*. Cryptology ePrint Archive, Paper 2025/118. 2025. URL: <https://eprint.iacr.org/2025/118>.
- [LPWW20] Benoît Libert, Alain Passelègue, Hoeteck Wee, and David J. Wu. “New Constructions of Statistical NIZKs: Dual-Mode DV-NIZKs and More”. In: *EUROCRYPT 2020, Part III*. Ed. by Anne Canteaut and Yuval Ishai. Vol. 12107. LNCS. Springer, Cham, May 2020, pp. 410–441. DOI: [10.1007/978-3-030-45727-3_14](https://doi.org/10.1007/978-3-030-45727-3_14).
- [LTW05] Henry Lin, Luca Trevisan, and Hoeteck Wee. “On Hardness Amplification of One-Way Functions”. In: *TCC 2005*. Ed. by Joe Kilian. Vol. 3378. LNCS. Springer, Berlin, Heidelberg, Feb. 2005, pp. 34–49. DOI: [10.1007/978-3-540-30576-7_3](https://doi.org/10.1007/978-3-540-30576-7_3).
- [Lu06] Chi-Jen Lu. “On the Complexity of Parallel Hardness Amplification for One-Way Functions”. In: *TCC 2006*. Ed. by Shai Halevi and Tal Rabin. Vol. 3876. LNCS. Springer, Berlin, Heidelberg, Mar. 2006, pp. 462–481. DOI: [10.1007/11681878_24](https://doi.org/10.1007/11681878_24).
- [Lu09] Chi-Jen Lu. “On the Security Loss in Cryptographic Reductions”. In: *EUROCRYPT 2009*. Ed. by Antoine Joux. Vol. 5479. LNCS. Springer, Berlin, Heidelberg, Apr. 2009, pp. 72–87. DOI: [10.1007/978-3-642-01001-9_4](https://doi.org/10.1007/978-3-642-01001-9_4).
- [Mau05] Ueli M. Maurer. “Abstract Models of Computation in Cryptography (Invited Paper)”. In: *10th IMA International Conference on Cryptography and Coding*. Ed. by Nigel P. Smart. Vol. 3796. LNCS. Springer, Berlin, Heidelberg, Dec. 2005, pp. 1–12. DOI: [10.1007/11586821_1](https://doi.org/10.1007/11586821_1).

- [MDK14] Bodo Möller, Thai Duong, and Krzysztof Kotowicz. *This POODLE Bites: Exploiting The SSL 3.0 Fallback*. Tech. rep. Google, 2014. URL: <https://web.archive.org/web/20141109023947/https://www.openssl.org/~bodo/ssl-poodle.pdf>.
- [Mir06] Ilya Mironov. “Collision-Resistant No More: Hash-and-Sign Paradigm Revisited”. In: *PKC 2006*. Ed. by Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin. Vol. 3958. LNCS. Springer, Berlin, Heidelberg, Apr. 2006, pp. 140–156. DOI: [10.1007/11745853_10](https://doi.org/10.1007/11745853_10).
- [MOZ22] Alice Murphy, Adam O’Neill, and Mohammad Zaheri. “Instantiability of Classical Random-Oracle-Model Encryption Transforms”. In: *ASIACRYPT 2022, Part IV*. Ed. by Shweta Agrawal and Dongdai Lin. Vol. 13794. LNCS. Springer, Cham, Dec. 2022, pp. 323–352. DOI: [10.1007/978-3-031-22972-5_12](https://doi.org/10.1007/978-3-031-22972-5_12).
- [Nec94] Vassiliy Ilyich Nechaev. “Complexity of a Determinate Algorithm for the Discrete Logarithm”. In: *Mathematical Notes* 55.2 (Feb. 1994), pp. 165–172. DOI: [10.1007/BF02113297](https://doi.org/10.1007/BF02113297).
- [Nie02] Jesper Buus Nielsen. “Separating Random Oracle Proofs from Complexity Theoretic Proofs: The Non-committing Encryption Case”. In: *CRYPTO 2002*. Ed. by Moti Yung. Vol. 2442. LNCS. Springer, Berlin, Heidelberg, Aug. 2002, pp. 111–126. DOI: [10.1007/3-540-45708-9_8](https://doi.org/10.1007/3-540-45708-9_8).
- [Pas16] Rafael Pass. “Unprovable Security of Perfect NIZK and Non-interactive Non-malleable Commitments”. In: *Comput. Complex.* 25.3 (2016), pp. 607–666. DOI: [10.1007/s00037-016-0122-2](https://doi.org/10.1007/s00037-016-0122-2). URL: <https://doi.org/10.1007/s00037-016-0122-2>.
- [Pie08] Krzysztof Pietrzak. “Compression from Collisions, or Why CRHF Combiners Have a Long Output”. In: *CRYPTO 2008*. Ed. by David Wagner. Vol. 5157. LNCS. Springer, Berlin, Heidelberg, Aug. 2008, pp. 413–432. DOI: [10.1007/978-3-540-85174-5_23](https://doi.org/10.1007/978-3-540-85174-5_23).
- [PRS12] Krzysztof Pietrzak, Alon Rosen, and Gil Segev. “Lossy Functions Do Not Amplify Well”. In: *TCC 2012*. Ed. by Ronald Cramer. Vol. 7194. LNCS. Springer, Berlin, Heidelberg, Mar. 2012, pp. 458–475. DOI: [10.1007/978-3-642-28914-9_26](https://doi.org/10.1007/978-3-642-28914-9_26).
- [PS19] Chris Peikert and Sina Shiehian. “Noninteractive Zero Knowledge for NP from (Plain) Learning with Errors”. In: *CRYPTO 2019, Part I*. Ed. by Alexandra Boldyreva and Daniele Micciancio. Vol. 11692. LNCS. Springer, Cham, Aug. 2019, pp. 89–114. DOI: [10.1007/978-3-030-26948-7_4](https://doi.org/10.1007/978-3-030-26948-7_4).
- [Res18] E. Rescorla. *The Transport Layer Security (TLS) Protocol Version 1.3*. <https://tools.ietf.org/html/rfc8446>. Aug. 2018.
- [RTV04] Omer Reingold, Luca Trevisan, and Salil P. Vadhan. “Notions of Reducibility between Cryptographic Primitives”. In: *TCC 2004*. Ed. by Moni Naor. Vol. 2951. LNCS. Springer, Berlin, Heidelberg, Feb. 2004, pp. 1–20. DOI: [10.1007/978-3-540-24638-1_1](https://doi.org/10.1007/978-3-540-24638-1_1).
- [Rud88] Steven Rudich. “Limits on the Provable Consequences of One-way Functions”. PhD thesis. University of California Berkeley, 1988.

- [Sha92] Adi Shamir. “IP = PSPACE”. In: *J. ACM* 39.4 (Oct. 1992), pp. 869–877. ISSN: 0004-5411. DOI: [10.1145/146585.146609](https://doi.org/10.1145/146585.146609). URL: <https://doi.org/10.1145/146585.146609>.
- [Sho97] Victor Shoup. “Lower Bounds for Discrete Logarithms and Related Problems”. In: *EUROCRYPT’97*. Ed. by Walter Fumy. Vol. 1233. LNCS. Springer, Berlin, Heidelberg, May 1997, pp. 256–266. DOI: [10.1007/3-540-69053-0_18](https://doi.org/10.1007/3-540-69053-0_18).
- [Sim98] Daniel R. Simon. “Finding Collisions on a One-Way Street: Can Secure Hash Functions Be Based on General Assumptions?” In: *EUROCRYPT’98*. Ed. by Kaisa Nyberg. Vol. 1403. LNCS. Springer, Berlin, Heidelberg, May 1998, pp. 334–345. DOI: [10.1007/BFb0054137](https://doi.org/10.1007/BFb0054137).
- [Unr07] Dominique Unruh. “Random Oracles and Auxiliary Input”. In: *CRYPTO 2007*. Ed. by Alfred Menezes. Vol. 4622. LNCS. Springer, Berlin, Heidelberg, Aug. 2007, pp. 205–223. DOI: [10.1007/978-3-540-74143-5_12](https://doi.org/10.1007/978-3-540-74143-5_12).
- [Wat00] John Watrous. “Succinct quantum proofs for properties of finite groups”. In: *41st FOCS*. IEEE Computer Society Press, Nov. 2000, pp. 537–546. DOI: [10.1109/SFCS.2000.892141](https://doi.org/10.1109/SFCS.2000.892141).
- [Yao82] Andrew Chi-Chih Yao. “Theory and Applications of Trapdoor Functions (Extended Abstract)”. In: *23rd FOCS*. IEEE Computer Society Press, Nov. 1982, pp. 80–91. DOI: [10.1109/SFCS.1982.45](https://doi.org/10.1109/SFCS.1982.45).
- [Zha16] Mark Zhandry. “The Magic of ELFs”. In: *CRYPTO 2016, Part I*. Ed. by Matthew Robshaw and Jonathan Katz. Vol. 9814. LNCS. Springer, Berlin, Heidelberg, Aug. 2016, pp. 479–508. DOI: [10.1007/978-3-662-53018-4_18](https://doi.org/10.1007/978-3-662-53018-4_18).
- [Zha19] Mark Zhandry. “The Magic of ELFs”. In: *Journal of Cryptology* 32.3 (July 2019), pp. 825–866. DOI: [10.1007/s00145-018-9289-9](https://doi.org/10.1007/s00145-018-9289-9).

PART II

Publications

On Derandomizing Yao’s Weak-to-Strong OWF Construction

Chris Brzuska¹ Geoffroy Couteau² Pihla Karanko¹ Felix Rohrbach³

¹ Aalto University, Finland, {chris.brzuska,pihla.karanko}@aalto.fi

² CNRS, IRIF, Université Paris Cité, France, geoffroy.couteau@ens.fr

³ TU Darmstadt, Germany, felix.rohrbach@cryptoplexity.de

Abstract

The celebrated result of Yao (FOCS’82) shows that concatenating $\lambda \cdot p(\lambda)$ copies of a *weak* one-way function (OWF) f , which can be inverted with probability $1 - \frac{1}{p(\lambda)}$, yields a *strong* OWF g , showing that weak and strong OWFs are black-box equivalent. Yao’s transformation is not *security-preserving*, i.e., the input to g needs to be much larger than the input to f . Understanding whether a larger input is inherent is a long-standing open question.

In this work, we explore necessary features of constructions which achieve short input length by proving the following: for any *direct product* construction of a strong OWF g from a weak OWF f , which can be inverted with probability $1 - \frac{1}{p(\lambda)}$, the input size of g must grow as $\Omega(p(\lambda))$. Here, *direct product* refers to the following structure: the construction g executes some arbitrary pre-processing function (independent of f) on its input s , obtaining a vector (x_1, \dots, x_l) , and outputs $f(x_1), \dots, f(x_l)$. When setting the pre-processing to be the identity, one recovers thus Yao’s construction.

Our result generalizes to functions g with post-processing, as long as the post-processing function is not too lossy. Thus, in essence, any weak-to-strong OWF hardness amplification must either (1) be very far from security-preserving, (2) use adaptivity, or (3) must be very far from a *direct-product* structure (in the sense that post-processing of the outputs of f is very lossy).

On a technical level, we use ideas from lower bounds for secret-sharing to prove the impossibility of derandomizing Yao in a black-box way. Our results are in line with Goldreich, Impagliazzo, Levin, Venkatesan, and Zuckerman (FOCS 1990) who derandomize Yao’s construction for *regular* weak OWFs by evaluating the OWF along a random walk on an expander graph—the construction is adaptive, since it alternates steps on the expander graph with evaluations of the weak OWF.

1 Introduction

In this work, we continue the study of constructions of strong one-way functions (OWFs) from weak OWFs. The classical weak-to-strong hardness amplification technique, due to Yao [Yao82], uses direct product amplification which is not security preserving¹. Our main result shows that the

¹In a security-preserving construction, the input length of the strong OWF is linear in that of the weak OWF.

increase in the input size is inherent for *direct product* constructions. Namely, any direct product black-box construction of a strong OWF from a $(1 - 1/p(\lambda))$ -weak OWF must have input length at least $\Omega(p(\lambda))$.

Weak and strong OWFs.

An $\alpha(\lambda)$ -secure OWF $f : \{0, 1\}^\lambda \mapsto \{0, 1\}^\lambda$ is an efficiently computable function such that any probabilistic polynomial-time adversary \mathcal{A} can invert f with probability at most $\alpha(\lambda)$. When α is a negligible function, we say that f is a *strong* OWF; when $\alpha(\lambda) = 1 - 1/p(\lambda)$ for a polynomial p , we say that f is a *weak* OWF. The seminal work of Yao [Yao82] shows that weak OWFs imply strong OWFs, via a standard *direct product* hardness amplification: given a weak OWF f , define $g(x_1, \dots, x_l) = f(x_1) \parallel \dots \parallel f(x_l)$. Then, Yao proved that g is a strong OWF for $l > |x_i|p(|x_i|)$.

Adaptive vs. non-adaptive construction.

In this paper we study *non-adaptive* weak-to-strong OWF constructions, that is, constructions where the calls to the weak OWF can be made in parallel. I.e., a strong OWF construction g that makes calls to a weak OWF f is called *non-adaptive* if g ’s calls to f only depend on g ’s input, but not on the output of f on any of these inputs. Yao’s construction is a simple, non-adaptive construction where each call to f uses an independent chunk of the input. In general, non-adaptive constructions can make correlated calls to f though.

We say that a construction is *adaptive*, if the output of (at least) one call to f is used to determine the input to another f call. That is, adaptive constructions cannot compute all calls to f in parallel. For the toy constructions on the right, g_1 is non-adaptive (it does not matter whether g_1 computes $f(x)$ or $f(x + 1)$ first) and g_2 is adaptive (g_2 must make the inner f call first).

On the (in)efficiency of Yao’s construction.

The construction of Yao is generic: it turns an *arbitrary* weak OWF f into a strong OWF g and just depends on the hardness of f . In addition, g has an appealing simple direct-product structure. In turn, g is suboptimal w.r.t. its computational complexity:

1. g makes a *large number of calls* to the underlying weak OWF, and
2. g is *not security preserving*: g ’s input length is polynomially larger than the input length of f .

Many celebrated cryptographic reductions are similarly not security-preserving and have a high number of calls—the HILL construction of pseudorandom generator from any OWF being perhaps one of the most well-known examples [HILL99]. In beautiful works, a decade ago, Haitner, Reingold, Vadhan and Zheng [HRV10; VZ12] developed rich tools for computational entropy, and improved the original n^8 seed length by HILL to $\mathcal{O}(n^3)$, where n is the input length of the OWF—since further improvements seem extremely hard to obtain, it is natural to ask whether large lower bounds on the input size are inherent.

In a seminal result [IR89], Impagliazzo and Rudich formalize the notions of *black-box* constructions/reductions, and develop methods to establish their limitations. Informally, a (fully) black-box construction of a primitive C from a primitive P treats both P and any adversary \mathcal{A} against P

in a black-box way. Following this breakthrough result, a long line of work (see [KST99; GT00; GGG03; LTW05; Lu06; CRSTVW07; Wee07; Lu09]) has been devoted to proving limitations on the *efficiency* of black-box reductions. Our work continues this successful line of work.

To our knowledge, three previous works study black-box limitations on the efficiency of Yao’s construction. Lin, Trevisan, and Wee [LTW05] address the first of the two limitations above: they show that any fully black-box construction of an $\varepsilon(\lambda)$ -secure OWF from a $(1 - \delta(\lambda))$ -secure OWF f must make at least $q = \Omega((1/\delta) \cdot \log(1/\varepsilon))$ calls to f . They also show that fully black-box constructions cannot be perfectly security-preserving: if f has input size λ , the input size of the strong OWF must be at least $\lambda + \Omega(\log 1/\varepsilon) - O(\log q)$. Later, Lu [Lu09] extends the results of [LTW05] to the weakly black-box setting with bounded non-uniformity. Moreover, Lu [Lu06] shows that a *non-adaptive* fully black-box construction (i.e., a construction where all the calls to f are made in parallel) cannot amplify security beyond $\text{poly}(\lambda)$ if the algorithm implementing the reduction has constant depth, and its size is below $2^{\text{poly}(\lambda)}$.

1.1 On Security-Preserving Amplification of Weak OWFs

The above result leaves open one of the most intriguing limitations of Yao’s construction: the fact that it causes a polynomial blowup in the input size. While [LTW05] shows that *some* blowup in the input size is unavoidable, it leaves a huge gap: starting with a $(1 - 1/p(\lambda))$ -secure OWF f with input length λ , Yao’s construction requires an input size $\lambda^2 \cdot p(\lambda)$ to build *any* strong OWF, while the result of [LTW05] only shows that to build an *extremely strong* OWF, say a $2^{-\mu \cdot \lambda}$ -secure OWF (for some constant μ), one needs input size at least $(1 + \mu) \cdot \lambda - \log p$.

In a sense, the proof of [LTW05] cannot do much better, because it rules out even *adaptive* fully black-box reductions. However, in this setting, it is actually known that we can do much better than Yao’s construction and obtain an almost security-preserving construction, if we start from a *regular* (i.e. outputs have the same number of preimages) weak OWF, and use adaptivity. Indeed, the work of Goldreich, Impagliazzo, Levin, Venkatesan, and Zuckerman [GILVZ90] provides precisely such a construction, using random walks on expander graphs. Following that, Haitner, Harnik and Reingold [HHR06] present another almost security-preserving adaptive construction using hash function calls instead of expander steps; and their construction is secure even when the regularity parameters is not known.

This leaves us in between two extremes: on the one hand, Yao’s construction is non-adaptive (hence optimally parallelizable: if one starts with a parallelizable weak OWF, one ends up with a parallelizable strong OWF), extremely simple (it has a straightforward direct product structure) and works for arbitrary OWFs; however, it is not security-preserving. On the other hand, the constructions [GILVZ90] and [HHR06] are almost security-preserving, but are considerably more involved, require adaptive calls, and work only for regular OWFs. Improving this state of affairs is a long-standing and intriguing open problem.

1.2 Our Contribution

In this work, we make progress on understanding the *limits* of non-adaptive constructions. Specifically, we show that any *direct product* black-box construction of strong OWF from a $(1 - 1/p(\lambda))$ -secure OWF cannot be security preserving, in a strong sense: it requires an input length of at least $\Omega(p(\lambda))$. While this still leaves a gap with respect to Yao’s construction, which has input

length $O(\lambda^2 \cdot p(\lambda))$, this gap vanishes asymptotically when p grows. By *direct product* construction, we mean a construction g of strong OWF with the following structure: on input s , $g(s)$ outputs $(f(x_1), \dots, f(x_\ell))$, where f is the weak OWF, and (x_1, \dots, x_ℓ) are computed from s arbitrarily, but without calling f (we call the mapping from s to (x_1, \dots, x_ℓ) the *pre-processing*). This is a natural generalization of *Yao-style* constructions of strong OWFs (we recover Yao’s construction by letting the pre-processing be the identity function). Furthermore, our result generalizes to the setting where some post-processing (independent of f) is applied to the outputs $(f(x_1), \dots, f(x_\ell))$, whenever this post-processing is *not too lossy*: we prove that whenever each output of the post-processing has at most polynomially many preimages, the same $\Omega(p(\lambda))$ input length bound holds. We summarize the results in the following theorem:

Theorem 1.1 (Informal). *Let f be a $(1 - 1/p(\lambda))$ -secure OWF (a weak OWF). Let g be any non-adaptive construction, with not-too-compressing post-processing, of input length $< cp(\lambda)$ for any constant c . Then, it is impossible to prove, in a fully black-box way, that g is a strong OWF.*

Observe that if we could generalize our result to arbitrary (f -independent) post-processing functions, the above would capture all non-adaptive constructions. Hence, in essence, our result says the following: any (fully black-box) construction of strong OWF from a weak OWF must either (1) be very far from security preserving, or (2) use adaptivity, or (3) compute a highly non-injective function of the outputs of the non-adaptive calls (i.e., be very far from a “direct product” structure).

1.3 Relation to Correlated-Product and Correlated-Input Security

Usually, parallel concatenation of cryptographic primitives on *independent* inputs preserves security. For example, if f and g are one-way functions, then so is $(x_1, x_2) \mapsto (f(x_1), g(x_2))$. However, things might change significantly when x_1 and x_2 are *correlated*, e.g., sampled jointly from a high min-entropy source. Variants of this problem have been studied on many occasions in cryptography, and have profound connections to the feasibility of cryptography with weak sources of randomness, leakage-resilient cryptography, related-key attacks, or deterministic encryption (to name a few); see e.g. Wichs [Wic13] for discussions on cryptography with correlated sources. In addition, security for correlated inputs has proven to be a very useful assumption by itself: one-wayness under correlated product (i.e., one-wayness of $f(x_1), \dots, f(x_k)$ for (x_1, \dots, x_k) sampled from a joint distribution) has been used to build CCA secure cryptosystems [RS09; HLO12], and correlated-input secure hash functions have found numerous applications such as OT extension [IKNP03], trapdoor hash function [DGIMMO19], constrained pseudorandom functions [AMNYY18], password-based login [GOR11], and many more.

A general and natural question to ask is: which type of constructions *preserve* hardness, when the inputs are jointly sampled from a high min-entropy source, rather than being sampled independently? This is a fundamental question in itself, because this setting occurs in real-life use of standard cryptographic construction (when they are misused, when the source of randomness is imperfect, or when the adversary has access to some leakage on the inputs), but also due to the many applications outlined above.

It is well-known that not all constructions will preserve security under correlated inputs. For example, even though the map $x \mapsto x^e \bmod n$ is believed to be one-way when n is a product of two large safe primes (this is the RSA assumption), the extended euclidean algorithm provides an efficient inverter for the map $x \mapsto (x^{e_1}, x^{e_2}) \bmod n$ whenever $\gcd(e_1, e_2) = 1$ (this example is

taken from [HLO12]). Hence, there are specific functions f_i (here, $f_i : x \mapsto x^{e_i}$) and specific correlations of the inputs (here, the equality correlation: the same input x is used for all functions) such that correlated-product security breaks down. However, this leaves open the possibility that some specific input correlations preserve correlated-product security (for example, this is the case when the correlated-inputs are indistinguishable from random, e.g. when sampled as the output of a PRG), or that some specific functions maintain correlated-product security for general correlations.

Our results can be cast in the context of correlated-product security: we show that even though Yao’s construction of OWF, which is a very natural and seminal construction, is provably secure (with a black-box proof) when used with random and independent inputs, it breaks down for *any possible correlated source*, whenever the entropy of the source is below $p(\lambda)$. This provides a natural example of a construction, from a weak OWF f , where correlated-product security cannot be generically shown to hold (in a black-box way) for *arbitrary* sources, unless they contain enough entropy such that all of the correlated inputs can have independent entropy. In contrast, [RS09] shows that when f is instantiated as a *lossy trapdoor function*, then $f(x_1), \dots, f(x_k)$ is one-way for correlated inputs (x_1, \dots, x_k) , and [HLO12] shows that assuming OWFs, there *exists* a correlated-product secure function. Our results provide a partial complementary perspective to this line of work.

Comparison to [Wic13]. Wichs [Wic13] also studies, among other questions, the one-wayness of constructions of the form $(f(x_1), \dots, f(x_k))$ for inputs (x_1, \dots, x_k) sampled from a correlated source. Our results are incomparable: we show that for a *generic weak OWF* f , and for any *fixed* distribution over the inputs (x_1, \dots, x_k) with $o(k)$ bits of entropy, the one-wayness of $f(x_1), \dots, f(x_k)$ does not follow from that of f in a black-box way. In contrast, [Wic13] shows that for an *arbitrary function* f , there is no black-box reduction (to any standard hardness assumption) of one-wayness of $(f(x_1), \dots, f(x_k))$ when the x_i can come from *arbitrarily correlated distributions*, even with high per-input entropy. That is, [Wic13] handles a considerably larger class of constructions and reductions to many possible assumptions, but only rules out a much more stringent security notion (where one-wayness must hold even when the input distributions are not fixed a priori and can be correlated arbitrarily). To state the difference using quantifiers (we omit the hardness parameter of the weak OWF for simplicity), we show that

\exists weakOWF $f \forall$ pre-proc: $g(x) := f(\text{pre-proc}(x)_1), \dots, f(\text{pre-proc}(x)_l)$ is **not** a strong OWF

while Wichs [Wic13] shows that

\exists corr-srce $\forall f$: $g(x) := f(\text{corr-srce}(x)_1), \dots, f(\text{corr-srce}(x)_l)$ is **not** a strong OWF

that is, the quantification is in different order, as is highlighted in pink. To put it differently, we *rule out* fully black-box constructions of the form

\forall weakOWF $f \exists$ pre-proc: $g(x) := f(\text{pre-proc}(x)_1), \dots, f(\text{pre-proc}(x)_l)$ is a strong OWF

and Wichs rules out fully black-box constructions of the form

\forall corr-srce $\exists f$: $g(x) := f(\text{corr-srce}(x)_1), \dots, f(\text{corr-srce}(x)_l)$ is a strong OWF.

1.4 Related Work

We already discussed several works on bounding the efficiency of black-box reductions [KST99; GT00; GGGK03; LFW05; Lu06; CRSTVW07; Wee07; Lu09], including some specifically targeting hardness

amplifications of one-way functions, and related work on correlated-product security. Besides, our black-box separations use some established tools (in addition to key new technical insights, which we cover afterwards) such as the two-oracle technique of [Sim98; HR04] where one oracle implements the base primitive and the second oracle breaks all constructions built from this primitive. We use the Borel-Cantelli style technique from [MMNPs16] to extract a single oracle from a distribution of random oracles analogously to the seminal work on black-box separations by Impagliazzo and Rudich [IR89].

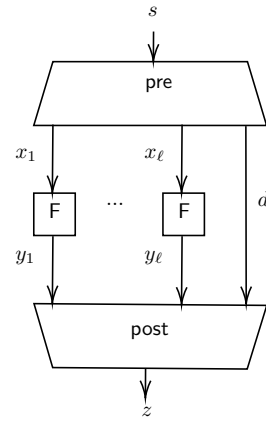
Hardness amplification of functions, via direct products and related constructions, have a rich history, which goes well beyond one-way functions and is too vast to be covered here. In particular, amplifying the hardness of *computing* boolean functions (rather than *inverting* functions) using direct product constructions is at the heart of major lines of work on worst-case to average-case reductions, constructions of non-cryptographic pseudorandom generators, circuit lower bounds, and many more – see e.g. [Lip91; BFL91; BFNW93; Imp95; GNW95; IW97; STV01; Tre03; HVV04; Tre05; Vio05; SV08] and references therein.

1.5 Technical Overview

To prove our black-box separations, we exhibit an oracle relative to which there is a weak one-way function, yet all strong one-way functions with an appropriate structure can be inverted efficiently with constant probability. The standard method to do so is to design oracles relative to which the starting primitive (here, the weak one-way function) clearly exists and is the *only possible source of hardness*. For example, in the seminal work by Impagliazzo and Rudich (IR) on the separation of key exchange from OWFs [IR90], IR introduce a random oracle, which is a strong OWF with high probability, as well as assuming $P = NP$, thereby ruling out most other (stronger) cryptographic primitives. In our setting, we instantiate this intuition by choosing three oracles:

- (1) A PSPACE oracle, which *destroys* all possible sources of hardness,
- (2) a random oracle F , which instantiates the weak OWF, and
- (3) an inverter INV , which inverts F on a (roughly) $1 - 1/p$ fraction of its inputs, effectively turning it into a weak OWF. Note that a random oracle F alone would already be a strong OWF, if we did not weaken it by adding INV .

In this oracle world, we consider *non-adaptive* constructions of strong OWFs g from the weak OWF F . Since we wish to rule out (relativizing) *fully black-box* reductions (as defined by Reingold, Trevisan and Vadhan [RTV04]), we do not give g access to INV . In fact, this is inherent in our setting: observe that given access to INV , it is not too hard to build a strong OWF (e.g. the strong OWF can perform a random walk starting from the input x , until it lands on a hard input y – which can be tested using INV – and



```

g(s)
-----
x1, ..., x_l, d ← pre(s)
for i = 1..l
    yi ← F(xi)
z ← post(y1, ..., y_l, d)
return z

```

FIGURE 1.1 — (n, m) -non-adaptive construction. F is the weak OWF. Length of d can be arbitrary, $|x_i| = |y_i| = m$ and $|s| = n$.

outputs $F(y)$). In general, whenever one can efficiently test which inputs are hard, constructing a security-preserving OWF becomes feasible – and it is precisely the lack of any such tester that makes it highly nontrivial to improve over Yao’s seminal construction. Since we rule out fully black-box reductions, we do not let g access INV and thus, g does not know where the easy inputs are.

Modeling non-adaptive constructions.

A non-adaptive construction can be thought of as a circuit which first has a pre-processing layer, followed by a layer of parallel calls to a weak OWFs and then some post-processing, see Figure 1.1. When the construction omits the post-processing layer, as in Yao’s construction, this corresponds to a *direct product* construction. The input size n of the construction might be different from the input size m of the weak OWF. As a starting point, we consider what happens when the construction does not use any post-processing, as is the case in Yao’s construction. When there is no post-processing, the additional data d in Figure 1.1 only reduces the input domain and does not add security. Thus, w.l.o.g., we assume that there is no d .

Inverting direct product constructions.

Considering the simple case with no post-processing and no d , the first observation is that g must make more than $p(m)$ calls to the weak OWF, since otherwise *all* the calls will be easy to invert with constant probability. In that case the adversary could simply invert all the weak OWF calls and then use PSPACE to invert the pre-processing layer, thus inverting g with constant probability.

Now that g makes at least $p(m)$ calls to the weak OWF, we can make the main observation of the paper: if we can invert a $1 - 1/p(m)$ fraction of the weak OWF calls and λ is about the same as $p(m)$, then the remaining entropy of the input s cannot be very high, on the average. This is formalized in Lemma 5.4. This is because the number of calls to the weak OWF is at least the same order of magnitude as the length of the input to the strong OWF. Hence, there is not enough entropy in the strong OWF input to distribute among all the weak OWF calls, so most of the calls will end up having very little entropy of their own, i.e. entropy that is not shared with other calls.

Now the probability that an adversary can indeed invert a $1 - 1/p(m)$ fraction of the weak OWF calls is high, since that is the expected fraction of easy calls. Since the entropy of the input s is low, given the easy calls, and the adversary has the PSPACE oracle, the adversary can guess s with high probability. Note that low entropy alone is not enough to guess s , since inverting pre-processing might be inefficient, hence we also need PSPACE.

To summarize, we know that there must be many calls to the underlying weak one-way function—and since we can also show that each of them must have a non-trivial amount of entropy (i.e., information about the input)—we can show that we can invert all non-adaptive constructions without post-processing, unless λ is larger than any constant times $p(m)$, establishing the first lower bound on the randomness efficiency of non-adaptive constructions. Note that Yao’s construction consumes $n = m^2 p(m)$ many bits.

On strong OWFs with injectiveish post-processing.

We sketched above why constructions without post-processing (direct product constructions) cannot be strongly one-way. It is relatively easy to extend the above argument to constructions with *not too lossy* post-processing, i.e., constructions where any output of the post-processing has at

most polynomially many preimages: the inverter chooses a uniformly random value amongst the (polynomial size) list of all possible preimages of the post-processing, and applies the previous inversion attack on the candidate. It then succeeds with probability $\frac{1}{\text{poly}}$ times the success probability of the previous attack.

Relation to Threshold Secret Sharing.

The pre-processing `pre` in Figure 1.1 is somewhat analogous to a threshold secret sharing scheme, where the participants’ shares correspond to the values x_i and the secret together with the dealer’s randomness corresponds to the strong OWF input s . On average, we learn the ‘shares’ of all but a $\frac{1}{p(m)}$ fraction of the ‘participants’.

The analogy is somewhat weak though, since the OWF inverter needs to find one (whole) pre-image s , while a secret sharing scheme is broken if the adversary finds the secret which is a (known) function of s (even if they cannot recover the dealer’s randomness, that is, all of s). Also, computational security suffices for OWFs while secret sharing schemes usually aim for information-theoretic security.

Interestingly, in the proof of our negative result, the adversary has a PSPACE oracle, and hence, in the proof we care about the *information theoretic* security. This is because PSPACE can invert computationally hard pre-processing, as long as it is not information theoretically impossible, i.e. the input does not have too high entropy conditioned on the adversary’s knowledge. Hence, even though the object we study is not formally related to secret sharing, our proof is inspired by previous work on secret sharing by Blundo, Santis, and Vaccaro (BSV [BSV96]).

BSV discuss the minimum amount of randomness needed by an information-theoretically secure secret sharing scheme. BSV prove that if the secret length is m and there are l participants, then the dealer needs to use $l \cdot m$ bits of randomness (to choose both the secret and the participants’ shares). The length of this randomness corresponds to the analogous number in Yao’s weak to strong OWF construction (when number of weak OWF calls is $l > mp(m)$, we use lm input length) and it is close to the analogous number that we get in this paper (input length to strong OWF needs to be $\mathcal{O}(p(m))$, i.e. there is an m^2 gap between our lower bound and Yao’s upper bound).

It is intuitive that some gap should exist between the information theoretically secure secret sharing scheme and our more relaxed “mostly secure secret sharing scheme”, where the adversary is allowed to learn a function of the input as long as they cannot learn the entire input. However, the OWF construction and secret sharing schemes are not formally related and a better lower bound than ours might be possible to obtain.

2 Preliminaries

We use $x \leftarrow S$ for sampling x uniformly from set S , and $y \leftarrow \mathcal{A}(x)$ for running randomized algorithm \mathcal{A} on x with uniformly random coins and assigning the result to y . We write $\Pr_{\mathcal{A}}[1 = \mathcal{A}(x)]$ for the probability over the randomness of \mathcal{A} that \mathcal{A} , on input x , returns 1. We now introduce weak and strong one-way functions, oracle algorithms, relativization, black-box reductions and entropy.

Definition 2.1 (One-Way Functions). Let $f : \{0, 1\}^* \rightarrow \{0, 1\}^*$ be a polynomial-time computable function. f is called a (*strong*) one-way function (OWF), if for every probabilistic polynomial-time

algorithm \mathcal{A} there exists a negligible function $\epsilon : \mathbb{N} \rightarrow [0, 1]$ such that for every n ,

$$\Pr_{\mathcal{A}, x \leftarrow \{0,1\}^n} [\mathcal{A}(1^n, f(x)) \in f^{-1}(f(x))] \leq \epsilon(n).$$

Further, f is called a *weak* one-way function, if there exists a polynomial $p(n)$ such that for every probabilistic polynomial-time algorithm \mathcal{A} there exists a $N_0 \in \mathbb{N}$ such that for all $n \geq N_0$:

$$\Pr_{\mathcal{A}, x \leftarrow \{0,1\}^n} [\mathcal{A}(1^n, f(x)) \in f^{-1}(f(x))] \leq 1 - \frac{1}{p(n)}.$$

In this case we sometimes say that f is a *p-weak* OWF.

Definition 2.2 (Oracle Algorithms). The complexity of an oracle algorithm (e.g., Turing Machine) is the number of steps it makes, where an oracle query is counted as a single step.

In particular, a probabilistic polynomial-time (PPT) oracle algorithm makes at most a polynomial number of queries. Since our oracle algorithms have access to a PSPACE oracle, our runtime discussions only consider the number of oracle calls the algorithm makes.

Definition 2.3 (Relativizing Statements). We say that a statement about algorithms *relativizes* if it also holds whenever all algorithms are given oracle access to an arbitrary (deterministic) function O .

To rule out a relativizing statement, we first argue about *distributions* of oracles and then show the existence of a single oracle using the following Borel-Cantelli style theorem.

Theorem 2.4 ([MMNPs16], Lemma 2.9). *Let (E_1, E_2, \dots) be a sequence of events such that $\exists c \forall m \in \mathbb{N} : \Pr[E_m] \geq c$, where constant c is $0 < c < 1$. Then,*

$$\Pr \left[\bigwedge_{k=1}^{\infty} \bigvee_{m>k} E_m \right] \geq c \tag{2.1}$$

Intuitively, Theorem 2.4 says that if an event happens with constant probability for all m , then, with constant probability, the event happens infinitely often.

Entropy. Throughout this paper, the term *entropy* refers to *Shannon entropy* which satisfies a *chain rule*.

Definition 2.5 (Shannon Entropy). Let X be a random variable and let $\text{dom}(X)$ be its domain, then

$$H(X) := - \sum_{z \in \text{dom}(X)} \Pr[X = z] \cdot \log_2(\Pr[X = z]),$$

is the *Shannon entropy* of X .

Lemma 2.6 (Chain Rule for Entropy). *Let X_1, \dots, X_n be random variables. Then the following holds*

$$H(X_1, \dots, X_n) = H(X_1) + H(X_2|X_1) + \dots + H(X_n|X_1, \dots, X_{n-1}).$$

We use also other simple but useful properties of entropy. In particular, Definition 2.5 implies that entropy is non-negative. Also, the entropy $H(X)$ of a random variable X is always more or equal to the entropy $H(f(X))$ of the random variable $f(X)$ for any deterministic function f —if f is injective, the entropy is preserved, if f is not injective, it decreases. Finally, for any three random variables X, Y, Z , we have that $H(X|Y) \geq H(X|Y, Z)$, i.e., conditioning on additional information maintains or decreases the entropy of a random variable.

3 Main Results

In this section, we introduce different types of constructions of strong OWF from weak OWF which we study in this paper (Section 3.1) and state our main theorems (Section 3.2). In particular, we introduce non-adaptive constructions, non-adaptive constructions without post-processing and non-adaptive constructions with *injectiveish* post-processing.

3.1 Black-box constructions and reductions

Definition 3.1 (Non-adaptive). A construction $g = (\text{pre}, \text{post})$ from a weak one-way function F is non-adaptive, if it computes its output as $\text{post}(F(\text{pre}(s)))$ (see Fig 1.1). The number of queries ℓ is induced by pre . (n, m) -NA denotes a non-adaptive construction with input length n based on a weak OWF F whose input length is m .

Definition 3.2 (Non-adaptive, no post-processing construction). We say that a construction $g = (\text{pre}, \text{post})$ is a (n, m) -NANPP, if it is (n, m) -NA and the post-processing function is the identity function, i.e., $\text{post}(y_1, \dots, y_\ell, d) := y_1 || \dots || y_\ell || d$.

Definition 3.3 (Non-adaptive, injectiveish post-processing constr.). We say that a construction $g = (\text{pre}, \text{post})$ is a (n, m) -NAIPP, if it is (n, m) -NA and the post-processing function is almost injective, that is, every image of post has at most a polynomial (in n) number of preimages.

Note that the identity function is injective and thus, in particular, is *injectiveish*. Therefore, every NANPP is also a NAIPP, but the converse does not hold. Likewise, both NANPP and NAIPP are NA constructions, but the converse does not hold. Since we are interested in ruling out constructions, whenever we rule out NAIPP, we also rule out NANPP.

We formalized the kind of constructions our negative results capture, and now specify which type of reduction proofs our theorems rule out. Namely, our results concern BBB-style proofs following the notation of [BBF13] or fully black-box proofs following the notation of [RTV04]. Since we consider parametrized definitions, we here state a customized version of fully black-box security which precisely captures the quantifiers our negative results capture.

Definition 3.4 (Fully Black-Box Proof). We say that a proof that weak OWF implies strong OWF is *fully black-box* if it establishes a *relativizing* statement of the following type:

$$\forall \text{poly } p, \exists \text{ poly-time computable } g_p, \forall \text{poly } q, \exists PPT \mathcal{R}_q \forall p\text{-weak OWF } F, \mathcal{A} : \quad (3.2)$$

$$\left(\Pr_{x \leftarrow \mathfrak{s}\{0,1\}^\lambda} [g_p^F(\mathcal{A}(1^\lambda, g_p^F(x))) = g_p^F(x)] > \frac{1}{q(\lambda)} \text{ for infinitely many } \lambda \in \mathbb{N} \right)$$

$$\Rightarrow \left(\Pr_{x \leftarrow \mathfrak{s}\{0,1\}^\lambda} [F(\mathcal{R}_q^{\mathcal{A}, F}(1^\lambda, F(x))) = F(x)] > 1 - \frac{1}{p(\lambda)} \text{ for inf. many } \lambda \right) \quad (3.3)$$

In this case, we also refer to the *construction* g as fully black-box.

Intuitively, line (3.2) says that an adversary \mathcal{A} breaks the strong OWF g^F and line (3.3) says that the reduction $\mathcal{R}^{\mathcal{A}}$ breaks the weak-OWF F .

Remark. Typically, in the definition of fully black-box, the pink parts are omitted. That is, the polynomial p is considered part of the definition of F and the polynomial q is considered as part of the definition of the adversary \mathcal{A} , namely its success probability. We allow the construction g to depend on the polynomial p and the reduction \mathcal{R} to depend on q , since we seek to cover a larger and meaningful class of proofs. In particular, Yao's original proof building strong OWFs from weak OWFs is fully black-box in the sense of Definition 3.4, but would not be covered if the construction were not allowed to depend on p or if the reduction was not allowed to depend on q .

3.2 Theorems

We now state our main theorems, all of which rely on the two-oracle technique. Namely, we construct a distribution over oracles $(\mathcal{O}_1, \mathcal{O}_2)$ such that \mathcal{O}_1 is a *weak* one-way function and \mathcal{O}_2 helps to invert the *strong* one-way function and is part of the adversary. Since we rule out black-box reductions rather than provide an oracle separation, only the reduction has access to the (adversary) oracle \mathcal{O}_2 while the construction does not (cf. Section 1.5). Corollary 3.7 extracts a single oracle from the oracle distribution, using the Borel-Cantelli style argument Theorem 2.4. However, we prefer to state our theorem in terms of oracle distributions since this matches the technical core arguments of our separation results more closely.

Theorem 3.5 (NANPP Impossibility). \forall constant \mathbf{d} , \forall poly p , $\forall (n, m)$ -NANPP g with input length $n \leq \mathbf{d} \cdot p(m)$, \exists poly-query \mathcal{A} , \exists poly $q(n) = n^c$, $c \in \mathbb{N}_+$, \forall PPT \mathcal{R} , \exists distribution \mathcal{D} over pairs of oracles $(\mathcal{O}_1, \mathcal{O}_2)$:

$$\Pr_{(\mathcal{O}_1, \mathcal{O}_2) \leftarrow \mathcal{D}} [\text{Bad}_m^{\mathcal{R}, \mathcal{A}, g}] = \text{constant} < 1$$

where $\text{Bad}_m^{\mathcal{R}, \mathcal{A}, g}$ is an indicator variable that is 1 iff at least one of the following is true:

1. Weak OWF breaks:

$$\Pr_{x \leftarrow \mathcal{S}_{\{0,1\}^m}, \mathcal{R}} [\mathcal{R}^{\mathcal{A}^{\mathcal{O}_1, \mathcal{O}_2}, \mathcal{O}_1, \mathcal{O}_2}(1^m, \mathcal{O}_1(x)) \in \mathcal{O}_1^{-1}(\mathcal{O}_1(x))] \geq 1 - \frac{1}{p(m)}.$$

2. Strong OWF is secure-ish:

$$\Pr_{s \leftarrow \mathcal{S}_{\{0,1\}^n}, \mathcal{A}} [\mathcal{A}^{\mathcal{O}_1, \mathcal{O}_2}(1^n, g^{\mathcal{O}_1}(s)) \in (g^{\mathcal{O}_1})^{-1}(g^{\mathcal{O}_1}(s))] \leq \frac{1}{q(n)}.$$

Remark. In the definition of the bad event, the oracles are *fixed* and the randomness is taken only over the sampling of x as well as the internal randomness of \mathcal{A} and \mathcal{R} , respectively. Thus, $\text{Bad}_m^{\mathcal{R}, \mathcal{A}, g}$ is indeed a well-defined event once the oracles $(\mathcal{O}_1, \mathcal{O}_2)$ have been sampled from \mathcal{D} .

Theorem 3.6 (NAIPP Impossibility). \forall constant \mathbf{d} , \forall poly p , $\forall (n, m)$ -NAIPP g with input length $n \leq \mathbf{d} \cdot p(m)$, \exists poly-query \mathcal{A} , \exists poly $q(n) = n^c$, $c \in \mathbb{N}_+$, \forall PPT \mathcal{R} , \exists distribution \mathcal{D} over pairs of oracles $(\mathcal{O}_1, \mathcal{O}_2)$:

$$\Pr_{(\mathcal{O}_1, \mathcal{O}_2) \leftarrow \mathcal{D}} [\text{Bad}_m^{\mathcal{R}, \mathcal{A}, g}] = \text{constant} < 1$$

where $\text{Bad}_m^{\mathcal{R}, \mathcal{A}, g}$ is the same indicator variable as in Theorem 3.5.

We use the same oracle distribution for Theorem 3.6 and Theorem 3.5, see Section 4 for its definition. Theorem 3.6 implies Theorem 3.5, so it would suffice to prove Theorem 3.6. However, we found the presentation to be easier to follow when presenting the proof of the weaker Theorem 3.5 first (Section 5.2) and then discussing the generalization to the proof of Theorem 3.6 (Section 6). For both theorems, we prove that relative to $\mathcal{O}_1, \mathcal{O}_2$, oracle \mathcal{O}_1 is a weak OWF. Before proving the

theorems for oracle distributions, we now extract a single oracle from the distribution where the bad event happens with *constant* probability. Since the standard Borel-Cantelli lemma requires the probability to be less than $1/m^2$, we here use the strengthened version by Mahmoody, Mohammed, Nematihaji, Pass and Shelat [MMNPs16].

Corollary 3.7 (Main). *Let \mathbf{d} be any constant. There is no fully black-box (n, m) -NAIPP construction of a OWF from a $p(m)$ -weak OWF with $n \leq \mathbf{d} \cdot p(m)$.*

Proof. Recall that a black-box proof means the following:

$$\begin{aligned} & \forall \text{poly } p, \exists \text{ poly-time computable } g, \forall \text{poly } q, \exists \text{PPT } \mathcal{R} \forall p\text{-weak OWF } \mathbf{F}, \mathcal{A} : \\ & (\mathcal{A} \text{ inverts } g) \Rightarrow (\mathcal{R}^{\mathcal{A}} \text{ inverts } \mathbf{F}) \text{ Formally:} \\ & \left(\Pr_{x \leftarrow \mathfrak{s}\{0,1\}^\lambda} [g^{\mathbf{F}}(\mathcal{A}(1^\lambda, g^{\mathbf{F}}(x))) = g^{\mathbf{F}}(x)] > \frac{1}{q(\lambda)} \text{ for infinitely many } \lambda \in \mathbb{N} \right) \\ & \Rightarrow \left(\Pr_{x \leftarrow \mathfrak{s}\{0,1\}^\lambda} [\mathbf{F}(\mathcal{R}^{\mathcal{A}, \mathbf{F}}(1^\lambda, \mathbf{F}(x))) = \mathbf{F}(x)] > 1 - \frac{1}{p}(\lambda) \text{ for infi. many } \lambda \in \mathbb{N} \right) \end{aligned}$$

In order to rule out a black-box proof, we thus define an oracle \mathcal{O}_1 (and an oracle \mathcal{O}_2 helping the adversary) such that the following holds:

$$\begin{aligned} & \forall \text{poly } p, \forall \text{ poly-time } g^{\mathcal{O}_1}, \exists \text{poly } q, \forall \text{PPT } \mathcal{R}^{\mathcal{O}_1, \mathcal{O}_2} \exists \mathcal{A}^{\mathcal{O}_1, \mathcal{O}_2}, \exists \mathcal{O}_1, \mathcal{O}_2 : \\ & \mathcal{A} \text{ breaks } g^{\mathcal{O}_1}, \text{ but } \mathcal{R} \text{ does not } p\text{-invert } \mathcal{O}_1. \text{ Formally:} \\ & (1) \Pr_{x \leftarrow \mathfrak{s}\{0,1\}^\lambda} [g^{\mathcal{O}_1}(\mathcal{A}^{\mathcal{O}_1, \mathcal{O}_2}(1^\lambda, g^{\mathcal{O}_1}(x))) = g^{\mathcal{O}_1}(x)] > \frac{1}{q(\lambda)} \text{ for inf. many } \lambda \in \mathbb{N} \\ & (2) \Pr_{x \leftarrow \mathfrak{s}\{0,1\}^\lambda} [\mathcal{O}_1(\mathcal{R}^{\mathcal{A}, \mathcal{O}_1, \mathcal{O}_2}(1^\lambda, \mathcal{O}_1(x))) = \mathcal{O}_1(x)] < 1 - \frac{1}{p}(\lambda) \\ & \text{for all but finitely many } \lambda \in \mathbb{N} \end{aligned}$$

Strictly speaking, we only need to prove the above for *some polynomial* p , but since our proof establishes (1) and (2) for *all polynomial* p anyways, we prefer to state this stronger statement here. Now, let us fix a polynomial p , a candidate NAIPP g , a polynomial q (s.t. it satisfies Theorem 3.6) and a candidate reduction \mathcal{R} and show the existence of an adversary and a p -weak OWF \mathbf{F} .

By Theorem 3.6, there is an oracle distribution over pairs $(\mathcal{O}_1, \mathcal{O}_2)$, and an adversary \mathcal{A} such that the probability of the bad event $\text{Bad}_m^{\mathcal{R}, \mathcal{A}, g}$ is constant in m . We show that there exists a *fixed* oracle pair $(\mathcal{O}_1, \mathcal{O}_2)$ for which the bad event $\text{Bad}_m^{\mathcal{R}, \mathcal{A}, g}$ in Theorem 3.6 happens only for finitely many m . From that it follows that there is a fixed oracle pair for which $\mathcal{A}^{\mathcal{O}_1, \mathcal{O}_2}$ breaks the candidate strong OWF $g^{\mathcal{O}_1}$ infinitely many often, but the reduction $\mathcal{R}^{\mathcal{A}^{\mathcal{O}_1, \mathcal{O}_2}}$ inverts the weak OWF \mathcal{O}_1 well enough at most on finitely many m . Thus, it suffices to show via Theorem 2.4, that Theorem 3.6 implies that there is an oracle relative to which $\text{Bad}_m^{\mathcal{R}, \mathcal{A}, g}$ happens only for finitely many m .

By Theorem 3.6, we have

$$\Pr_{(\mathcal{O}_1, \mathcal{O}_2) \leftarrow \mathfrak{s}\mathcal{D}} [\text{Bad}_m^{\mathcal{R}, \mathcal{A}, g}] = \text{constant} < 1.$$

Hence, the constant probability version of Borel-Cantelli (Theorem 2.4) yields

$$\Pr_{(\mathcal{O}_1, \mathcal{O}_2) \leftarrow \mathfrak{s}\mathcal{D}} \left[\bigwedge_{m=1}^{\infty} \bigvee_{m > k} \text{Bad}_m^{\mathcal{R}, \mathcal{A}, g} \right] = \text{constant} < 1,$$

which means that, with constant probability, there is a k for which no $m > k$ satisfies $\text{Bad}_m^{\mathcal{R}, \mathcal{A}, g}$. Taking such an oracle pair $(\mathcal{O}_1, \mathcal{O}_2)$ concludes the proof of Corollary 3.7.

4 Oracle Distributions

In this section, we define the oracle (distribution)s we rely on. Firstly, a PSPACE creates a world where no one-way functions exist. Then, we add an oracle (distribution) F in order to create a world where weak one-way functions exist, and finally, we add an oracle (distribution) \mathcal{O}_2 which breaks NANPP and NAIPP constructions. The adversary will have access to \mathcal{O}_2 , PSPACE and F while the candidate strong OWF construction only has access to PSPACE and F, but not to \mathcal{O}_2 . We recall from Section 1.5 that it is *necessary* to not give the construction access to the information which parts of F are easy and which parts are hard, and not giving the construction access to \mathcal{O}_2 is related to this necessary restriction, since the adversary (modeled by \mathcal{O}_2) uses the information of which parts are easy. On a technical-conceptual level, it is meaningful to not give the construction access to the adversary (modeled by \mathcal{O}_2), since the adversary is *inefficient*, while the construction is efficient (in this (oracle) world where all algorithms have access to PSPACE and F). We consider an inefficient adversary since we rule out black-box reduction which work for *any* black-box adversary that breaks the strong OWF, including inefficient ones.

As mentioned before, we denote our adversary by \mathcal{O}_2 . We encode the pair of oracles PSPACE and F into a single oracle \mathcal{O}_1 so that we are aligned with the terminology of a two-oracle separation result (and this is also convenient notation in the proof).

Definition 4.1 (Oracle Distributions). Let \mathbf{p} be any fixed polynomial. The oracle distribution $D_{\mathbf{p}}$ over oracles \mathcal{O}_1 and \mathcal{O}_2 samples permutations Π_m of the elements in $\{0, 1\}^m$ for every $m \in \mathbb{N}$ and a random subset $\text{EASY}_{\text{in}}^m$ of $\{0, 1\}^m$ s.t. $|\text{EASY}_{\text{in}}^m| = \lceil (1 - 1/\mathbf{p}(m))2^m \rceil$. We define

$$\mathcal{O}_1 := (\text{PSPACE}, \text{F}) \text{ and } \mathcal{O}_2 := \text{INV},$$

where F and INV behave as follows:

$\frac{\text{F}(x)}{m \leftarrow x }$	$\frac{\text{INV}(y)}{m \leftarrow y }$
$y \leftarrow \Pi_m(x)$	if $y \in \text{EASY}_{\text{out}}^m$
return y	return $\text{F}^{-1}(y)$
	else return \perp

Here, we use $\text{EASY}_{\text{out}}^m := \Pi_m(\text{EASY}_{\text{in}}^m)$.

Remark. Throughout this paper we treat $(1 - 1/\mathbf{p}(m))2^m$ as an integer, omitting the ceil function since the difference does not affect our proofs.

5 Proof of Theorem 3.5

We split the proof of Theorem 3.5 into two parts. We first show that the probability of Case 1 (weak OWF breaks) of the bad event introduced in Theorem 3.5 is smaller than *any* constant (Section 5.1), and then we show that the probability of Case 2 (strong OWF is secure-ish) of the bad event introduced in Theorem 3.5 is a small constant (Section 5.2). Recall that both probabilities are (only) over the sampling of the oracles \mathcal{O}_1 and \mathcal{O}_2 .

5.1 $\mathcal{R}^{\mathcal{A}}$ is not a successful weak OWF inverter

In this section, we show that the probability (over the oracle distributions) that F is not a $2\mathbf{p}(m)$ -weak OWF is small.

Theorem 5.1 (F is Weak OWF). *For all constants \mathbf{c} , for all polynomials \mathbf{p} , for all poly-query $\mathcal{A}^{\mathbf{F}, \text{PSPACE}, \text{INV}}$, for all adversaries \mathcal{R} making polynomially many (in m) queries to the oracles \mathbf{F} , PSPACE , INV , $\mathcal{A}^{\mathbf{F}, \text{PSPACE}, \text{INV}}$,*

$$\Pr_{\mathbf{F}, \text{PSPACE}, \text{INV} \leftarrow \mathcal{D}_{\mathbf{p}}} \left[\text{Succlnv}_{\mathcal{A}, \mathcal{R}}^{\mathbf{F}, \text{PSPACE}, \text{INV}} \geq 1 - \frac{1}{2\mathbf{p}(m)} \right] \leq 1/\mathbf{c}$$

where $\text{Succlnv}_{\mathcal{A}, \mathcal{R}}^{\mathbf{F}, \text{PSPACE}, \text{INV}}$ is defined as

$$\Pr_{x \leftarrow \{0,1\}^m, \mathcal{R}} \left[\mathcal{R}^{\mathbf{F}, \text{PSPACE}, \text{INV}, \mathcal{A}^{\mathbf{F}, \text{PSPACE}, \text{INV}}} (1^m, F(x)) \in F^{-1}(F(x)) \right].$$

When we define $\mathbf{p}(m) := \frac{1}{2}p(m)$, the above is equivalent to

$$\Pr_{(\mathcal{O}_1, \mathcal{O}_2) \leftarrow \mathcal{D}} \left[\text{Case 1 of } \text{Bad}_m^{\mathcal{R}, \mathcal{A}, g} \right] \leq 1/\mathbf{c},$$

where $\mathcal{D} := \mathcal{D}_{\mathbf{p}}$, $\mathcal{O}_1 := \mathbf{F}, \text{PSPACE}$, $\mathcal{O}_2 := \text{INV}$ and $\text{Bad}_m^{\mathcal{R}, \mathcal{A}, g}$ is defined as in Theorem 3.5.

The proof of Theorem 5.1 uses standard techniques for arguing one-wayness of a random oracle. We include the proof in Appendix B for completeness.

5.2 \mathcal{A} is a successful strong OWF inverter

We prove that an adversary with access to the oracles \mathbf{F} , INV and PSPACE (cf. Section 4), can break all short input NANPP constructions which have access to \mathbf{F} and PSPACE only.

Theorem 5.2 (Inverting OWF Candidate). *\forall constant \mathbf{d} , \forall poly \mathbf{p} , $\forall (n, m)$ -NANPP g with input length $n \leq \mathbf{d}\mathbf{p}(m)$, \exists poly-query $\mathcal{A}^{\mathbf{F}, \text{INV}, \text{PSPACE}}$, \exists constant $c > 0$ s.t.*

$$\Pr_{(\mathbf{F}, \text{INV}) \leftarrow \mathcal{D}_{\mathbf{p}}} \left[\Pr_{s, \mathcal{A}} \left[\mathcal{A}^{\mathbf{F}, \text{INV}, \text{PSPACE}} \text{ inverts } g(s) \right] \leq c \right] = \text{constant} < 1$$

This implies that

$$\Pr_{(\mathcal{O}_1, \mathcal{O}_2) \leftarrow \mathcal{D}} \left[\text{Case 2 of } \text{Bad}_m^{\mathcal{R}, \mathcal{A}, g} \right] = \text{constant} < 1$$

where $\mathcal{D} := \mathcal{D}_{\mathbf{p}}$, $\mathcal{O}_1 := \mathbf{F}, \text{PSPACE}$, $\mathcal{O}_2 := \text{INV}$ and $\text{Bad}_m^{\mathcal{R}, \mathcal{A}, g}$ is defined as in Theorem 3.5.

Interestingly, Theorem 5.2 does not depend on the number of calls to \mathbf{F} in the strong OWF construction g . That is, if the input length of the construction g is too short, then no number of calls to \mathbf{F} can make it a strong OWF. Now, let $\mathbf{p}(m)$ be a fixed polynomial. We now start by proving Theorem 5.2 for constructions which make few \mathbf{F} -queries, and later, we prove the theorem for larger number of queries. More precisely, we show that no matter what the input length to g is, for every constant c , g must make at least $l > c \cdot \mathbf{p}(m)$ calls to \mathbf{F} , otherwise all the \mathbf{F} calls are easy with constant probability, which makes inverting g trivial.

Proposition 5.3 (Easy inversion if few \mathbf{F} -Calls). *Consider a NANPP $g = (\text{pre}, \text{post})$ with $\text{post}(y_1, \dots, y_l, d) = y_1 \parallel \dots \parallel y_l \parallel d$. For all constants c , if $\text{pre}(s) = (x_1, \dots, x_l, d)$ induces at most $l \leq c\mathbf{p}(m)$*

(parallel) calls to F , then all $y_i := F(x_i)$ are in $\text{EASY}_{\text{out}}^m$ with constant probability, more precisely, $\exists \text{constant} > 0$ s.t.

$$\Pr_{F \leftarrow \mathcal{S}D_{\mathbf{p}}}[\Pr_s[\forall y_i \in g(s) : y_i \in \text{EASY}_{\text{out}}^m] > \text{constant}] > \text{constant} > 0 \quad (5.4)$$

In particular, with constant probability over the choice of the oracle F , g can be inverted with non-negligible (constant) probability by a poly-query adversary.

Proof. Suppose there are $l \leq c\mathbf{p}(m)$ parallel calls to F . Denote by y_1, \dots, y_l the outputs of the parallel calls to F . Now, when considering the randomness of choosing $\text{EASY}_{\text{in}}^m$, we have

$$\begin{aligned} & \Pr_{F \leftarrow \mathcal{S}D_{\mathbf{p}}, s}[y_1, \dots, y_l \in \text{EASY}_{\text{out}}^m] \\ & \geq \underbrace{\sum_s 2^{-|s|}}_{=1} \Pr_{F \leftarrow \mathcal{S}D_{\mathbf{p}}}[y_1 \in \text{EASY}_{\text{out}}^m | s] \cdot \dots \cdot \Pr_{F \leftarrow \mathcal{S}D_{\mathbf{p}}}[y_l \in \text{EASY}_{\text{out}}^m | s] \\ & = \left(1 - \frac{1}{\mathbf{p}(m)}\right)^l \geq \left(1 - \frac{1}{\mathbf{p}(m)}\right)^{c\mathbf{p}(m)} \geq \left(\frac{1}{4}\right)^c \quad \forall \mathbf{p}(m) > 2. \end{aligned}$$

where the first inequality is an equality iff $y_i \neq y_j \forall i \neq j$ and the second inequality follows since $(1 - \frac{1}{x})^x$ converges monotonously to $\frac{1}{e}$ and is greater than $\frac{1}{4}$ whenever $x \geq 2$. Now since $(\frac{1}{4})^c$ is constant, we can use a simple averaging argument to prove (5.4): namely, we show

$$\Pr_{F \leftarrow \mathcal{S}D_{\mathbf{p}}}[\Pr_s[y_1, \dots, y_l \in \text{EASY}_{\text{out}}^m] > 1/4^{c+1}] > 1/4^c - 1/4^{c+1} \quad (5.5)$$

as follows:

$$\begin{aligned} 1/4^c & \leq \Pr_{F \leftarrow \mathcal{S}D_{\mathbf{p}}, s}[y_1, \dots, y_l \in \text{EASY}_{\text{out}}^m] \\ & = \sum_F \Pr_{F \leftarrow \mathcal{S}D_{\mathbf{p}}}[F] \underbrace{\Pr_s[y_1, \dots, y_l \in \text{EASY}_{\text{out}}^m | F]}_{:=a_F} \\ & \leq \sum_{F \text{ s.t. } a_F > 1/4^{c+1}} \Pr_{F \leftarrow \mathcal{S}D_{\mathbf{p}}}[F] + \sum_{F \text{ s.t. } a_F \leq 1/4^{c+1}} \Pr_{F \leftarrow \mathcal{S}D_{\mathbf{p}}}[F] 1/4^{c+1} \\ & \leq \Pr_{F \leftarrow \mathcal{S}D_{\mathbf{p}}}[F \text{ s.t. } a_F > 1/4^{c+1}] + 1/4^{c+1} \end{aligned}$$

which proves (5.5).

In the case where all y_1, \dots, y_l are all easy, \mathcal{A} can invert y_1, \dots, y_l using INV oracle. Note that there is only a single pre-image x_i per y_i and thus, given the list x_1, \dots, x_l , \mathcal{A} can use the PSPACE oracle to find an s such that $\text{pre}(s) = x_1, \dots, x_l$.

Due to Proposition 5.3, for the remainder of this section, we can focus on constructions where pre makes more than $c \cdot \mathbf{p}(m)$ calls. Also in the case where g makes many queries, we can always invert the easy fraction of (y_1, \dots, y_l) . However, if many queries are made, then (with high probability) some y_i will also be hard. Of course, if pre-processing $\text{pre}(s) = (x_1, \dots, x_l)$ distributes the entropy well, then knowing some of the x_i might suffice to restrict the set of suitable candidate values s to a polynomial-sized set, and once a polynomial-sized set of candidates is obtained, a random candidate s is a suitable pre-image with high enough probability. How well does this strategy work when considering *arbitrary* pre-processing pre ?

To analyze this strategy, we study the entropy of the hard values x_i given $(1 - \frac{1}{\mathbf{p}(m)})\ell$ many easy values x_i (note that in expectation, $(1 - \frac{1}{\mathbf{p}(m)})\ell$ many values are easy) and seek to prove that

their entropy is low. Towards that goal, we look at the entropy of the $\frac{1}{\mathbf{p}(m)}\ell$ many first x_i under a fixed permutation π and given the $\ell - \frac{1}{\mathbf{p}(m)}\ell$ many last x_i under that permutation. That is, we are interested in the entropy

$$h(\pi) := H(X_{\pi(1)}, \dots, X_{\pi(\frac{\ell}{\mathbf{p}(m)})} | X_{\pi(\frac{\ell}{\mathbf{p}(m)}+1)}, \dots, X_{\pi(\ell)}),$$

where X_i is the random variable defined as follows: sample a uniformly random s from $\{0,1\}^n$, compute $\text{pre}(s)$ and take the i th output (i.e. the input to the i th F-call in g). We now prove that the expectation of entropy $h(\pi)$ is small. Lemma 5.4 (Small Entropy Expectation) is our main conceptual lemma.

Lemma 5.4 (Small Entropy Expectation). *Suppose $\mathbf{p}(m)$ divides ℓ . Then,*

$$\mathbb{E}_{\pi \leftarrow \mathfrak{s}\Pi(\ell)}[h(\pi)] \leq \frac{n}{\mathbf{p}(m)},$$

which is equivalent to

$$\mathbb{E}_{\pi \leftarrow \mathfrak{s}\Pi(\ell)} \left[H(X_{\pi(1)}, \dots, X_{\pi(\frac{\ell}{\mathbf{p}(m)})} | X_{\pi(\frac{\ell}{\mathbf{p}(m)}+1)}, \dots, X_{\pi(\ell)}) \right] \leq \frac{n}{\mathbf{p}(m)}. \quad (5.6)$$

Proof.

Let's consider a permutation π of the weak OWF inputs $x_{\pi(1)}, \dots, x_{\pi(\ell)}$. Let's divide the inputs x_i into $\mathbf{p}(m)$ equal-sized blocks as follows:

$$\underbrace{x_{\pi(1)}, \dots, x_{\pi(\frac{\ell}{\mathbf{p}(m)})}}_{\text{one block}}, \underbrace{x_{\pi(\frac{\ell}{\mathbf{p}(m)}+1)}, \dots, x_{\pi(2\frac{\ell}{\mathbf{p}(m)})}}_{\text{one block}}, x_{\pi(2\frac{\ell}{\mathbf{p}(m)}+1)}, \dots, x_{\pi(\ell)}$$

Each pink index starts a new block. Let's denote the set of the pink indices by

$$J := \left\{ 1, \frac{\ell}{\mathbf{p}(m)} + 1, 2\frac{\ell}{\mathbf{p}(m)} + 1, \dots, (\mathbf{p}(m) - 1)\frac{\ell}{\mathbf{p}(m)} + 1 \right\}.$$

Now, let S denote the seed random variable of length n and consider the following sum

$$\sum_{j \in J} \mathbb{E}_{\pi \leftarrow \mathfrak{s}\Pi(\ell)} \left[H \left(\underbrace{X_{\pi(j)}, \dots, X_{\pi(j+\frac{\ell}{\mathbf{p}(m)}-1)}}_{j\text{-th block}} \mid \underbrace{X_{\pi(j+\frac{\ell}{\mathbf{p}(m)})}, \dots, X_{\pi(\ell)}}_{\text{all } X_i \text{ after the } j\text{-th block}} \right) \right] \quad (5.7)$$

$$= \mathbb{E}_{\pi \leftarrow \mathfrak{s}\Pi(\ell)} \left[\sum_{j \in J} H \left(X_{\pi(j)}, \dots, X_{\pi(j+\frac{\ell}{\mathbf{p}(m)}-1)} \mid X_{\pi(j+\frac{\ell}{\mathbf{p}(m)})}, \dots, X_{\pi(\ell)} \right) \right] \quad (5.8)$$

$$= \mathbb{E}_{\pi \leftarrow \mathfrak{s}\Pi(\ell)} [H(X_{\pi(1)}, \dots, X_{\pi(\ell)})] \quad (5.9)$$

$$\leq \mathbb{E}_{\pi \leftarrow \mathfrak{s}\Pi(\ell)} [H(S)] \quad (5.10)$$

$$= n \quad (5.11)$$

where (5.8) holds by linearity of expectation, (5.9) holds by the chain rule for entropy (Lemma 2.6) and Inequality (5.10) holds because (X_1, \dots, X_ℓ) are computed by applying the deterministic function pre on S , and applying a deterministic function cannot increase entropy—the Inequality becomes equality if and only if pre is injective. Finally, Equality (5.11) follows since $H(S) = |S| = n$.

Now, from (5.7)-(5.11), we have

$$\begin{aligned} n &\geq \sum_{j \in J} \mathbb{E}_{\pi \leftarrow \mathfrak{s}\Pi(\ell)} \left[H \left(X_{\pi(j)}, \dots, X_{\pi(j + \frac{\ell}{\mathbf{p}(m)} - 1)} \mid X_{\pi(j + \frac{\ell}{\mathbf{p}(m)})}, \dots, X_{\pi(\ell)} \right) \right] \\ &\geq \sum_{j \in J} \mathbb{E}_{\pi \leftarrow \mathfrak{s}\Pi(\ell)} \left[H \left(X_{\pi(j)}, \dots, X_{\pi(j + \frac{\ell}{\mathbf{p}(m)} - 1)} \mid X_{\pi(i)}, i = 1, \dots, j - 1, j + \frac{\ell}{\mathbf{p}(m)}, \dots, \ell \right) \right] \end{aligned} \quad (5.12)$$

$$= \sum_{j \in J} \mathbb{E}_{\pi' \leftarrow \mathfrak{s}\Pi(\ell)} \left[H \left(X_{\pi'(1)}, \dots, X_{\pi'(\frac{\ell}{\mathbf{p}(m)})} \mid X_{\pi'(\frac{\ell}{\mathbf{p}(m)} + 1)}, \dots, X_{\pi'(\ell)} \right) \right] \quad (5.13)$$

$$= \mathbf{p}(m) \mathbb{E}_{\pi' \leftarrow \mathfrak{s}\Pi(\ell)} \left[H \left(X_{\pi'(1)}, \dots, X_{\pi'(\frac{\ell}{\mathbf{p}(m)})} \mid X_{\pi'(\frac{\ell}{\mathbf{p}(m)} + 1)}, \dots, X_{\pi'(\ell)} \right) \right] \quad (5.14)$$

where (5.12) follows since conditioning on more random variables can only decrease entropy. Here, we additionally condition on all $X_{\pi(i)}$ for $i < j$ and not only on those for $i \geq j + \frac{\ell}{\mathbf{p}(m)}$. In Equation (5.13) we change to a more convenient indexing where we perform a bijective mapping on all permutations $\pi'(1) = \pi(j), \dots, \pi'(\frac{\ell}{\mathbf{p}(m)}) = \pi(j + \frac{\ell}{\mathbf{p}(m)} - 1)$ thus maintaining the same distribution over all permutations. Since the summands do not depend on j anymore and $|J| = \mathbf{p}(m)$, Equation (5.14) follows which concludes the proof of Lemma 5.4.

With Lemma 5.4 at hand, we now turn to the proof of Theorem 5.2.

Proof of Theorem 5.2. Let g be a (n, m) -NANPP g with input length $n \leq \mathbf{dp}(m)$ and let ℓ be the number of queries to F which g makes. The adversary \mathcal{A} (described on the right) now tries to invert all y_1, \dots, y_ℓ using INV and puts a placeholder \perp as x_i when inversion fails. \mathcal{A} then computes a random pre-image of the pre-processing that matches the known x_i s and d , which is possible in polynomial-time using the PSPACE oracle. We now argue that a random pre-image of the pre-processing, that matches the known x_i s and d , is an actual preimage of $y_1 \parallel \dots \parallel y_\ell \parallel d$ under g with constant probability.

```

 $\frac{\mathcal{A}(y_1 \parallel \dots \parallel y_\ell \parallel d)}{\text{for } i \in 1, \dots, \ell$ 
   $x_i \leftarrow \text{INV}(y_i)$ 
 $s \leftarrow \mathfrak{s} \text{pre}^{-1}(x_1, \dots, x_\ell, d)$ 
return  $s$ 

```

From now on, we assume that $|d| = 0$. This is w.l.o.g. because the data d is known to the adversary, so it cannot add entropy. Further and also w.l.o.g., we assume that $\mathbf{p}(m)$ divides ℓ for all m, n (if there was a remainder, we could add constant dummy F -calls until there is no remainder. Such F -calls would not make g weaker nor stronger, so our result would still hold.) Note that if $\ell \leq \mathbf{p}(m)$, then with constant probability all x_i are easy and INV inverts all of them (cf. Theorem 5.3). In that case \mathcal{A} can use the PSPACE oracle to find a correct preimage s with probability 1. Hence, we can assume that $\ell > \mathbf{p}(m)$.

First, recall that $h(\pi)$ denotes the entropy of the hard values conditioned on knowing the easy values, i.e., $h(\pi)$ is equal to

$$H(X_{\pi(1)}, \dots, X_{\pi(\frac{\ell}{\mathbf{p}(m)})} \mid X_{\pi(\frac{\ell}{\mathbf{p}(m)} + 1)}, \dots, X_{\pi(\ell)}).$$

Now, Lemma 5.4 (Small Entropy Expectation) established that the expectation of entropy $h(\pi)$ is small:

$$\mathbb{E}_{\pi \leftarrow \mathfrak{s}\Pi(\ell)} [h(\pi)] \leq \frac{n}{\mathbf{p}(m)} < \mathbf{d}, \quad (5.15)$$

where we use that Theorem 5.2 assumes that $\mathbf{dp}(m) \geq n$. We now want to lower bound the probability that the remaining entropy $h(\pi)$ is less than $\frac{2n}{\mathbf{p}(m)}$. We call such a permutation π *good* and next we lower bound the probability of permutation π being good. By Markov's inequality

$$\mathbb{E}_{\pi \leftarrow \mathfrak{s}\Pi(\ell)}[h(\pi)] \geq \frac{2n}{\mathbf{p}(m)} \cdot \Pr_{\pi \leftarrow \mathfrak{s}\Pi(\ell)} \left[h(\pi) \geq \frac{2n}{\mathbf{p}(m)} \right].$$

Equivalently,

$$\Pr_{\pi \leftarrow \mathfrak{s}\Pi(\ell)} \left[h(\pi) \geq \frac{2n}{\mathbf{p}(m)} \right] \leq \frac{\mathbf{p}(m)}{2n} \cdot \mathbb{E}_{\pi \leftarrow \mathfrak{s}\Pi(\ell)}[h(\pi)] \stackrel{(5.15)}{\leq} \frac{\mathbf{p}(m)}{2n} \cdot \frac{n}{\mathbf{p}(m)} = \frac{1}{2}.$$

Hence,

$$\Pr_{\pi \in \Pi(\ell)} \left[h(\pi) < \frac{2n}{\mathbf{p}(m)} \right] \geq \frac{1}{2}. \quad (5.16)$$

Now we know that a permutation π is good with probability at least $\frac{1}{2}$. If the permutation π is good, then the remaining entropy of the input is small and thus, some inputs are very likely (cf. Lemma A.1 (Predictable Inputs) in Appendix A) and thus likely chosen by adversary \mathcal{A} which chooses a random pre-image amongst the possible candidates. With this intuition of the proof in mind, we can now lower-bound the probability of \mathcal{A} 's success formally.

$$\begin{aligned} & \Pr_{\mathfrak{F},s}[\mathcal{A} \text{ inverts } g(s)] \\ & \geq \Pr_{\mathfrak{F}} \left[\exists \pi : x_{\pi(1)}, \dots, x_{\pi\left(\left(1-\frac{1}{\mathbf{p}(m)}\right)\ell\right)} \in \text{EASY}_{\text{in}}^m \right] \\ & \quad \cdot \Pr_s \left[\mathcal{A} \text{ inverts } g(s) \mid \exists \pi : x_{\pi(1)}, \dots, x_{\pi\left(\left(1-\frac{1}{\mathbf{p}(m)}\right)\ell\right)} \in \text{EASY}_{\text{in}}^m \right] \\ & \geq \frac{1}{2} \Pr_s \left[\mathcal{A} \text{ inverts } g(s) \mid \underbrace{\exists \pi : x_{\pi(1)}, \dots, x_{\pi\left(\left(1-\frac{1}{\mathbf{p}(m)}\right)\ell\right)} \in \text{EASY}_{\text{in}}^m}_{:=B} \right] \end{aligned} \quad (5.17)$$

$$\geq \frac{1}{2} \Pr_s \left[\underbrace{\mathbb{H}\left(X_{\pi(1)}, \dots, X_{\pi\left(\frac{\ell}{\mathbf{p}(m)}\right)} \mid X_{\pi\left(\frac{\ell}{\mathbf{p}(m)}+1\right)}, \dots, X_{\pi(\ell)}\right) < \frac{2n}{\mathbf{p}(m)}}_{:=C} \mid B \right] \quad (5.18)$$

$$\cdot \Pr_s \left[\Pr_{s'} \left[\begin{array}{l} \forall k \in \pi(1), \dots, \pi(\ell/\mathbf{p}(m)): \\ \text{pre}(s')_k = \text{pre}(s)_k \end{array} \mid \begin{array}{l} \forall j \in \pi(\ell/\mathbf{p}(m)+1), \dots, \pi(\ell): \\ \text{pre}(s')_j = \text{pre}(s)_j \end{array} \right] > \frac{1}{2^{2\mathbf{d}+1}} \mid C, B \right] \quad (5.19)$$

$$\cdot \Pr_s \left[\mathcal{A} \text{ inverts } g(s) \mid \Pr_{s'} \left[\begin{array}{l} \forall k \in \pi(1), \dots, \pi(\ell/\mathbf{p}(m)): \\ \text{pre}(s')_k = \text{pre}(s)_k \end{array} \mid \begin{array}{l} \forall j \in \pi(\ell/\mathbf{p}(m)+1), \dots, \pi(\ell): \\ \text{pre}(s')_j = \text{pre}(s)_j \end{array} \right] > \frac{1}{2^{2\mathbf{d}+1}} \wedge C, B \right] \quad (5.20)$$

$$\geq \frac{1}{2} \cdot \frac{1}{2} \cdot \left(1 - \frac{2 \cdot \mathbf{d}}{2\mathbf{d}+1} \right) \cdot \frac{1}{2^{2\mathbf{d}+1}} = \text{constant} \quad (5.21)$$

where (5.17) follows from the fact that whether x_i is easy or not follows binomial distribution with $(1 - \frac{1}{\mathbf{p}(m)})\ell$ many easy values in expectation. Inequality (5.18) uses chain rule of probability. The fractions at (5.21) follow from the lemmas, namely, the probability on line (5.18) is less than 1/2 by Lemma 5.4 (Small Entropy Expectation) and probability on line (5.19) is less than $1 - \frac{2 \cdot \mathbf{d}}{2\mathbf{d}+1}$ by

Lemma A.1 (Predictable Inputs). The last fraction follows from the definition of adversary \mathcal{A} and the probability statement at (5.20). Namely, if adversary guesses a random s' which is consistent with the known x_i , and we condition the probability on such s' being correct $\frac{1}{2^{2d+1}}$ of the time, adversary must be right $\frac{1}{2^{2d+1}}$ of the time.

Now that we know that

$$\Pr_{\mathbb{F},s}[\mathcal{A} \text{ inverts } g(s)] \geq \text{const} > 0,$$

we can use an averaging argument to show that $\Pr_{\mathbb{F}}[\Pr_s[\mathcal{A} \text{ inverts } g(s)] > \text{const} > 0] \geq \text{const} > 0$: Suppose $\Pr_{\mathbb{F},s}[\mathcal{A} \text{ inverts } g(s)] \geq c_1 > 0$. Now

$$\begin{aligned} c_1 &\leq \Pr_{\mathbb{F},s}[\mathcal{A} \text{ inverts } g(s)] = \sum_{\mathbb{F}} \Pr_{\mathbb{F}}[\mathbb{F}] \underbrace{\Pr_s[\mathcal{A} \text{ inverts } g(s) \mid \mathbb{F}]}_{=: a_{\mathbb{F}}} \\ &= \sum_{\mathbb{F} \text{ s.t. } a_{\mathbb{F}} > c_1/2} \Pr_{\mathbb{F}}[\mathbb{F}] \underbrace{\Pr_s[\mathcal{A} \text{ inverts } g(s) \mid \mathbb{F}]}_{\leq 1} + \sum_{\mathbb{F} \text{ s.t. } a_{\mathbb{F}} \leq c_1/2} \Pr_{\mathbb{F}}[\mathbb{F}] \underbrace{\Pr_s[\mathcal{A} \text{ inverts } g(s) \mid \mathbb{F}]}_{\leq 1} \\ &\leq \Pr_{\mathbb{F}}[a_{\mathbb{F}} > c_1/2] + c_1/2 \end{aligned}$$

Since we established that indeed, $\Pr_{\mathbb{F}}[\Pr_s[\mathcal{A} \text{ inverts } g(s)] > c_1/2] \geq c_1/2$, where c_1 is a constant > 0 , this concludes the proof of Theorem 5.2. \square

Theorem 3.5 follows from the Theorems 5.2 and 5.1 by union bound, namely

$$\begin{aligned} \Pr_{(\mathcal{O}_1, \mathcal{O}_2) \leftarrow \mathcal{D}}[\text{Bad}_m^{\mathcal{R}, \mathcal{A}, g}] &= \Pr[\text{Case 1 of } \text{Bad}_m^{\mathcal{R}, \mathcal{A}, g} \text{ or Case 2 of } \text{Bad}_m^{\mathcal{R}, \mathcal{A}, g}] \\ &\leq 1/c + \underbrace{\text{constant from Theorem 5.2}}_{\leq 1 - c_1/2 \text{ i.e. prob. that } g \text{ is secure-ish}} < 1 \end{aligned}$$

Note that since the constant c in Theorem 5.1 can be made arbitrarily large, in particular, it can be chosen s.t. $1/c + \text{constant from Theorem 5.2} < 1$.

6 Constructions with post-processing

In this section, we prove Theorem 3.6. Towards this goal, we use the oracles \mathbb{F} , INV and PSPACE (cf. Section 4), and show that there are no short input NAIPP constructions under the oracles.

Theorem 6.1 (No Strong OWFs with Injectiveish Post-Processing). \forall poly \mathbf{p} , $\forall (n, m)$ -NAIPP g with input length $n \leq \frac{1}{4}\mathbf{p}(m)$, \exists poly $q(n) = n^c$, $c \in \mathbb{N}_+$, \exists poly-query $\mathcal{A}^{\mathbb{F}, \text{INV}, \text{PSPACE}}$ such that

$$\Pr_{(\mathbb{F}, \text{INV}) \leftarrow \mathcal{D}_{\mathbf{p}}}[\Pr_s, \text{coins of } \mathcal{A}[\mathcal{A}^{\mathbb{F}, \text{INV}, \text{PSPACE}} \text{ inverts } g(s)] \leq q(n)] = \text{constant} < 1$$

$$\text{and thus } \Pr_{(\mathcal{O}_1, \mathcal{O}_2) \leftarrow \mathcal{D}}[\text{Case 2 of } \text{Bad}_m^{\mathcal{R}, \mathcal{A}, g}] = \text{constant} < 1$$

where $\text{Bad}_m^{\mathcal{R}, \mathcal{A}, g}$ is defined as in Theorem 3.6.

Theorems 5.1 and 6.1 together imply Theorem 3.6 by union bound analogously to the NANPP case. It thus remains to prove Theorem 6.1.

Proof. Let g be (n, m) -NAIPP which makes ℓ queries to F and let \mathcal{A} be the adversary on the right which samples a uniformly random pre-image of z under post , then inverts the easy queries and returns a seed s which is consistent with the pre-image of the easy values. Firstly observe that \mathcal{A} runs in polynomial-time since it can use the PSPACE oracle for inverting post . Moreover, it makes only a polynomial number of queries since ℓ is a polynomial.

As the post-processing of g is almost injective, $y_1, \dots, y_\ell, d \leftarrow \text{post}^{-1}(z)$ returns the values y_1, \dots, y_ℓ, d which the one-wayness experiment used to compute z with probability $\frac{1}{\text{poly}(n)}$. This probability is independent of F . If y_1, \dots, y_ℓ, d are indeed the correct values, then adversary \mathcal{A} also finds a pre-image s with constant probability by the same arguments as in Theorem 5.2. Thus, the overall success of \mathcal{A} is $\frac{1}{\text{poly}(n)} \cdot \text{constant}$ which is inverse polynomial as required by Theorem 6.1. \square

Acknowledgments

We thank the anonymous reviewers for valuable comments. Parts of this work have been funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – SFB 1119 – 236615297 and by the Academy of Finland.

References

- [AMNYY18] Nuttapon Attrapadung, Takahiro Matsuda, Ryo Nishimaki, Shota Yamada, and Takashi Yamakawa. “Constrained PRFs for NC^1 in Traditional Groups”. In: *CRYPTO 2018, Part II*. Ed. by Hovav Shacham and Alexandra Boldyreva. Vol. 10992. LNCS. Springer, Cham, Aug. 2018, pp. 543–574. DOI: [10.1007/978-3-319-96881-0_19](https://doi.org/10.1007/978-3-319-96881-0_19).
- [BBF13] Paul Baecher, Christina Brzuska, and Marc Fischlin. “Notions of Black-Box Reductions, Revisited”. In: *ASIACRYPT 2013, Part I*. Ed. by Kazue Sako and Palash Sarkar. Vol. 8269. LNCS. Springer, Berlin, Heidelberg, Dec. 2013, pp. 296–315. DOI: [10.1007/978-3-642-42033-7_16](https://doi.org/10.1007/978-3-642-42033-7_16).
- [BFL91] László Babai, Lance Fortnow, and Carsten Lund. “Non-deterministic exponential time has two-prover interactive protocols”. In: *Computational complexity 1.1* (1991), pp. 3–40.
- [BFNW93] László Babai, Lance Fortnow, Noam Nisan, and Avi Wigderson. “BPP has subexponential time simulations unless exptime has publishable proofs”. In: *Computational Complexity 3.4* (1993), pp. 307–318.
- [BSV96] Carlo Blundo, Alfredo De Santis, and Ugo Vaccaro. “Randomness in Distribution Protocols”. In: *Inf. Comput.* 131.2 (1996), pp. 111–139.

- [CRSTVW07] Ran Canetti, Ronald L. Rivest, Madhu Sudan, Luca Trevisan, Salil P. Vadhan, and Hoeteck Wee. “Amplifying Collision Resistance: A Complexity-Theoretic Treatment”. In: *CRYPTO 2007*. Ed. by Alfred Menezes. Vol. 4622. LNCS. Springer, Berlin, Heidelberg, Aug. 2007, pp. 264–283. DOI: [10.1007/978-3-540-74143-5_15](https://doi.org/10.1007/978-3-540-74143-5_15).
- [DGIMMO19] Nico Döttling, Sanjam Garg, Yuval Ishai, Giulio Malavolta, Tamer Mour, and Rafail Ostrovsky. “Trapdoor Hash Functions and Their Applications”. In: *CRYPTO 2019, Part III*. Ed. by Alexandra Boldyreva and Daniele Micciancio. Vol. 11694. LNCS. Springer, Cham, Aug. 2019, pp. 3–32. DOI: [10.1007/978-3-030-26954-8_1](https://doi.org/10.1007/978-3-030-26954-8_1).
- [GGK03] Rosario Gennaro, Yael Gertner, and Jonathan Katz. “Lower bounds on the efficiency of encryption and digital signature schemes”. In: *35th ACM STOC*. ACM Press, June 2003, pp. 417–425. DOI: [10.1145/780542.780604](https://doi.org/10.1145/780542.780604).
- [GILVZ90] Oded Goldreich, Russell Impagliazzo, Leonid A. Levin, Ramarathnam Venkatesan, and David Zuckerman. “Security Preserving Amplification of Hardness”. In: *31st FOCS*. IEEE Computer Society Press, Oct. 1990, pp. 318–326. DOI: [10.1109/FSCS.1990.89550](https://doi.org/10.1109/FSCS.1990.89550).
- [GNW95] Oded Goldreich, Noam Nisan, and Avi Wigderson. “On Yao’s XOR lemma”. In: Technical Report TR95–050, Electronic Colloquium on Computational Complexity. 1995.
- [GOR11] Vipul Goyal, Adam O’Neill, and Vanishree Rao. “Correlated-Input Secure Hash Functions”. In: *TCC 2011*. Ed. by Yuval Ishai. Vol. 6597. LNCS. Springer, Berlin, Heidelberg, Mar. 2011, pp. 182–200. DOI: [10.1007/978-3-642-19571-6_12](https://doi.org/10.1007/978-3-642-19571-6_12).
- [GT00] Rosario Gennaro and Luca Trevisan. “Lower Bounds on the Efficiency of Generic Cryptographic Constructions”. In: *41st FOCS*. IEEE Computer Society Press, Nov. 2000, pp. 305–313. DOI: [10.1109/SFCS.2000.892119](https://doi.org/10.1109/SFCS.2000.892119).
- [HHR06] Iftach Haitner, Danny Harnik, and Omer Reingold. “On the Power of the Randomized Iterate”. In: *CRYPTO 2006*. Ed. by Cynthia Dwork. Vol. 4117. LNCS. Springer, Berlin, Heidelberg, Aug. 2006, pp. 22–40. DOI: [10.1007/11818175_2](https://doi.org/10.1007/11818175_2).
- [HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. “A Pseudorandom Generator from any One-way Function”. In: *SIAM Journal on Computing* 28.4 (1999), pp. 1364–1396.
- [HLO12] Brett Hemenway, Steve Lu, and Rafail Ostrovsky. “Correlated Product Security from Any One-Way Function”. In: *PKC 2012*. Ed. by Marc Fischlin, Johannes Buchmann, and Mark Manulis. Vol. 7293. LNCS. Springer, Berlin, Heidelberg, May 2012, pp. 558–575. DOI: [10.1007/978-3-642-30057-8_33](https://doi.org/10.1007/978-3-642-30057-8_33).
- [HR04] Chun-Yuan Hsiao and Leonid Reyzin. “Finding Collisions on a Public Road, or Do Secure Hash Functions Need Secret Coins?” In: *CRYPTO 2004*. Ed. by Matthew Franklin. Vol. 3152. LNCS. Springer, Berlin, Heidelberg, Aug. 2004, pp. 92–105. DOI: [10.1007/978-3-540-28628-8_6](https://doi.org/10.1007/978-3-540-28628-8_6).

- [HRV10] Iftach Haitner, Omer Reingold, and Salil P. Vadhan. “Efficiency improvements in constructing pseudorandom generators from one-way functions”. In: *42nd ACM STOC*. Ed. by Leonard J. Schulman. ACM Press, June 2010, pp. 437–446. DOI: [10.1145/1806689.1806750](https://doi.org/10.1145/1806689.1806750).
- [HVV04] Alexander Healy, Salil P. Vadhan, and Emanuele Viola. “Using nondeterminism to amplify hardness”. In: *36th ACM STOC*. Ed. by László Babai. ACM Press, June 2004, pp. 192–201. DOI: [10.1145/1007352.1007389](https://doi.org/10.1145/1007352.1007389).
- [IKNP03] Yuval Ishai, Joe Kilian, Kobbi Nissim, and Erez Petrank. “Extending Oblivious Transfers Efficiently”. In: *CRYPTO 2003*. Ed. by Dan Boneh. Vol. 2729. LNCS. Springer, Berlin, Heidelberg, Aug. 2003, pp. 145–161. DOI: [10.1007/978-3-540-45146-4_9](https://doi.org/10.1007/978-3-540-45146-4_9).
- [Imp95] Russell Impagliazzo. “Hard-Core Distributions for Somewhat Hard Problems”. In: *36th FOCS*. IEEE Computer Society Press, Oct. 1995, pp. 538–545. DOI: [10.1109/SFCS.1995.492584](https://doi.org/10.1109/SFCS.1995.492584).
- [IR89] Russell Impagliazzo and Steven Rudich. “Limits on the Provable Consequences of One-Way Permutations”. In: *21st ACM STOC*. ACM Press, May 1989, pp. 44–61. DOI: [10.1145/73007.73012](https://doi.org/10.1145/73007.73012).
- [IR90] Russell Impagliazzo and Steven Rudich. “Limits on the Provable Consequences of One-way Permutations”. In: *CRYPTO’88*. Ed. by Shafi Goldwasser. Vol. 403. LNCS. Springer, New York, Aug. 1990, pp. 8–26. DOI: [10.1007/0-387-34799-2_2](https://doi.org/10.1007/0-387-34799-2_2).
- [IW97] Russell Impagliazzo and Avi Wigderson. “P = BPP if E Requires Exponential Circuits: Derandomizing the XOR Lemma”. In: *29th ACM STOC*. ACM Press, May 1997, pp. 220–229. DOI: [10.1145/258533.258590](https://doi.org/10.1145/258533.258590).
- [KST99] Jeong Han Kim, Daniel R. Simon, and Prasad Tetali. “Limits on the Efficiency of One-Way Permutation-Based Hash Functions”. In: *40th FOCS*. IEEE Computer Society Press, Oct. 1999, pp. 535–542. DOI: [10.1109/SFFCS.1999.814627](https://doi.org/10.1109/SFFCS.1999.814627).
- [Lip91] Richard Lipton. “New directions in testing”. In: *Distributed computing and cryptography 2* (1991), pp. 191–202.
- [LTW05] Henry Lin, Luca Trevisan, and Hoeteck Wee. “On Hardness Amplification of One-Way Functions”. In: *TCC 2005*. Ed. by Joe Kilian. Vol. 3378. LNCS. Springer, Berlin, Heidelberg, Feb. 2005, pp. 34–49. DOI: [10.1007/978-3-540-30576-7_3](https://doi.org/10.1007/978-3-540-30576-7_3).
- [Lu06] Chi-Jen Lu. “On the Complexity of Parallel Hardness Amplification for One-Way Functions”. In: *TCC 2006*. Ed. by Shai Halevi and Tal Rabin. Vol. 3876. LNCS. Springer, Berlin, Heidelberg, Mar. 2006, pp. 462–481. DOI: [10.1007/11681878_24](https://doi.org/10.1007/11681878_24).
- [Lu09] Chi-Jen Lu. “On the Security Loss in Cryptographic Reductions”. In: *EURO-CRYPT 2009*. Ed. by Antoine Joux. Vol. 5479. LNCS. Springer, Berlin, Heidelberg, Apr. 2009, pp. 72–87. DOI: [10.1007/978-3-642-01001-9_4](https://doi.org/10.1007/978-3-642-01001-9_4).

- [MMNPs16] Mohammad Mahmoody, Ameer Mohammed, Soheil Nematihaji, Rafael Pass, and abhi shelat. *A Note on Black-Box Separations for Indistinguishability Obfuscation*. Cryptology ePrint Archive, Report 2016/316. 2016. URL: <https://eprint.iacr.org/2016/316>.
- [RS09] Alon Rosen and Gil Segev. “Chosen-Ciphertext Security via Correlated Products”. In: *TCC 2009*. Ed. by Omer Reingold. Vol. 5444. LNCS. Springer, Berlin, Heidelberg, Mar. 2009, pp. 419–436. DOI: [10.1007/978-3-642-00457-5_25](https://doi.org/10.1007/978-3-642-00457-5_25).
- [RTV04] Omer Reingold, Luca Trevisan, and Salil P. Vadhan. “Notions of Reducibility between Cryptographic Primitives”. In: *TCC 2004*. Ed. by Moni Naor. Vol. 2951. LNCS. Springer, Berlin, Heidelberg, Feb. 2004, pp. 1–20. DOI: [10.1007/978-3-540-24638-1_1](https://doi.org/10.1007/978-3-540-24638-1_1).
- [Sim98] Daniel R. Simon. “Finding Collisions on a One-Way Street: Can Secure Hash Functions Be Based on General Assumptions?” In: *EUROCRYPT’98*. Ed. by Kaisa Nyberg. Vol. 1403. LNCS. Springer, Berlin, Heidelberg, May 1998, pp. 334–345. DOI: [10.1007/BFb0054137](https://doi.org/10.1007/BFb0054137).
- [STV01] Madhu Sudan, Luca Trevisan, and Salil Vadhan. “Pseudorandom generators without the XOR lemma”. In: *Journal of Computer and System Sciences* 62.2 (2001), pp. 236–266.
- [SV08] Ronen Shaltiel and Emanuele Viola. “Hardness amplification proofs require majority”. In: *40th ACM STOC*. Ed. by Richard E. Ladner and Cynthia Dwork. ACM Press, May 2008, pp. 589–598. DOI: [10.1145/1374376.1374461](https://doi.org/10.1145/1374376.1374461).
- [Tre03] Luca Trevisan. “List-Decoding Using The XOR Lemma”. In: *44th FOCS*. IEEE Computer Society Press, Oct. 2003, pp. 126–135. DOI: [10.1109/SFCS.2003.1238187](https://doi.org/10.1109/SFCS.2003.1238187).
- [Tre05] Luca Trevisan. “On uniform amplification of hardness in NP”. In: *37th ACM STOC*. Ed. by Harold N. Gabow and Ronald Fagin. ACM Press, May 2005, pp. 31–38. DOI: [10.1145/1060590.1060595](https://doi.org/10.1145/1060590.1060595).
- [Vio05] Emanuele Viola. “The complexity of constructing pseudorandom generators from hard functions”. In: *computational complexity* 13.3-4 (2005), pp. 147–188.
- [VZ12] Salil P. Vadhan and Colin Jia Zheng. “Characterizing pseudoentropy and simplifying pseudorandom generator constructions”. In: *44th ACM STOC*. Ed. by Howard J. Karloff and Toniann Pitassi. ACM Press, May 2012, pp. 817–836. DOI: [10.1145/2213977.2214051](https://doi.org/10.1145/2213977.2214051).
- [Wee07] Hoeteck Wee. “One-Way Permutations, Interactive Hashing and Statistically Hiding Commitments”. In: *TCC 2007*. Ed. by Salil P. Vadhan. Vol. 4392. LNCS. Springer, Berlin, Heidelberg, Feb. 2007, pp. 419–433. DOI: [10.1007/978-3-540-70936-7_23](https://doi.org/10.1007/978-3-540-70936-7_23).
- [Wic13] Daniel Wichs. “Barriers in cryptography with weak, correlated and leaky sources”. In: *ITCS 2013*. Ed. by Robert D. Kleinberg. ACM, Jan. 2013, pp. 111–126. DOI: [10.1145/2422436.2422451](https://doi.org/10.1145/2422436.2422451).

- [Yao82] Andrew Chi-Chih Yao. "Theory and Applications of Trapdoor Functions (Extended Abstract)". In: *23rd FOCS*. IEEE Computer Society Press, Nov. 1982, pp. 80–91. DOI: [10.1109/SFCS.1982.45](https://doi.org/10.1109/SFCS.1982.45).

Appendix A Additional Lemmas and Proofs

Lemma A.1 (Predictable Inputs). *Consider $(X_1, \dots, X_\ell) = \text{pre}(s)$ for uniformly random s . If (a fixed) permutation π is good, i.e. such that the entropy of the $X_{\pi(i)}$ satisfies*

$$\mathbb{H}\left(X_{\pi(1)}, \dots, X_{\pi\left(\frac{\ell}{\mathbf{p}(m)}\right)} \mid X_{\pi\left(\frac{\ell}{\mathbf{p}(m)}+1\right)}, \dots, X_{\pi(\ell)}\right) < \frac{2n}{\mathbf{p}(m)}$$

then

$$\Pr_s \left[\Pr_{s'} \left[\begin{array}{c} \forall k = \pi(i), i = 1, \dots, \frac{\ell}{\mathbf{p}(m)} : \\ \text{pre}(s')_k = \text{pre}(s)_k \end{array} \mid \begin{array}{c} \forall j = \pi(i), i = \frac{\ell}{\mathbf{p}(m)} + 1, \dots, \ell : \\ \text{pre}(s')_j = \text{pre}(s)_j \end{array} \right] > \frac{1}{2^{2\mathbf{d}+1}} \right] \geq 1 - \frac{2 \cdot \mathbf{d}}{2^{\mathbf{d}+1}}$$

Proof. Since $n \leq \mathbf{d}\mathbf{p}(m)$, we get that

$$\mathbb{H}\left(X_{\pi(1)}, \dots, X_{\pi\left(\frac{\ell}{\mathbf{p}(m)}\right)} \mid X_{\pi\left(\frac{\ell}{\mathbf{p}(m)}+1\right)}, \dots, X_{\pi(\ell)}\right) < \frac{2n}{\mathbf{p}(m)} \leq 2 \cdot \mathbf{d} \quad (1.22)$$

Let $S_{h,e} \subseteq \{0, 1\}^m$ be defined as

$$S_{h,e} = \left\{ s : \Pr_{s'}[p_h(s') = p_h(s) \mid p_e(s') = p_e(s)] < \frac{1}{2^{2\mathbf{d}+1}} \right\},$$

where

$$p_e(s) := \text{pre}(s)_{\pi\left(\frac{\ell}{\mathbf{p}(m)}+1\right)}, \dots, \text{pre}(s)_{\pi(\ell)}$$

and

$$p_h(s) := \text{pre}(s)_{\pi(1)}, \dots, \text{pre}(s)_{\pi\left(\frac{\ell}{\mathbf{p}(m)}\right)}.$$

Using (1.22) and the definition of conditional Shannon entropy, we get that

$$\begin{aligned} 2 \cdot \mathbf{d} &> \mathbb{H}\left(\underbrace{X_{\pi(1)}, \dots, X_{\pi\left(\frac{\ell}{\mathbf{p}(m)}\right)}}_{=: p_h(s)} \mid \underbrace{X_{\pi\left(\frac{\ell}{\mathbf{p}(m)}+1\right)}, \dots, X_{\pi(\ell)}}_{=: p_e(s)}\right) \\ &= \sum_{s \in \{0,1\}^m} \Pr_{s'}[p_h(s') = p_h(s) \text{ and } p_e(s') = p_e(s)] \cdot |\log \Pr_{s'}[p_h(s') = p_h(s) \mid p_e(s') = p_e(s)]| \\ &= \sum_{s \in S_{h,e}} \Pr_{s'}[p_h(s') = p_h(s) \text{ and } p_e(s') = p_e(s)] \cdot |\log \Pr_{s'}[p_h(s') = p_h(s) \mid p_e(s') = p_e(s)]| \\ &\quad + \sum_{s \notin S_{h,e}} \Pr_{s'}[p_h(s') = p_h(s) \text{ and } p_e(s') = p_e(s)] \cdot |\log \Pr_{s'}[p_h(s') = p_h(s) \mid p_e(s') = p_e(s)]| \\ &\geq \left(\sum_{s \in S_{h,e}} \Pr_{s'}[p_h(s') = p_h(s) \text{ and } p_e(s') = p_e(s)] \right) \cdot \left| \log \frac{1}{2^{2\mathbf{d}+1}} \right| \end{aligned}$$

$$\begin{aligned}
& + \left(\sum_{s \notin S_{h,e}} \Pr_{s'}[p_h(s') = p_h(s) \text{ and } p_e(s') = p_e(s)] \right) \cdot |\log 1| \\
& \geq \Pr_s \left[\Pr_{s'}[p_h(s') = p_h(s) \mid p_e(s') = p_e(s)] < \frac{1}{2^{2\mathbf{d}+1}} \right] \cdot (2\mathbf{d} + 1) + 0
\end{aligned}$$

where \log is the base-2 logarithm and the last inequality is equality exactly when pre-processing is injective. Now

$$\begin{aligned}
2 \cdot \mathbf{d} & \geq \Pr_s \left[\Pr_{s'}[p_h(s') = p_h(s) \mid p_e(s') = p_e(s)] < \frac{1}{2^{2\mathbf{d}+1}} \right] \cdot (2\mathbf{d} + 1) \\
\Leftrightarrow \frac{2 \cdot \mathbf{d}}{2\mathbf{d} + 1} & \geq \Pr_s \left[\Pr_{s'}[p_h(s') = p_h(s) \mid p_e(s') = p_e(s)] < \frac{1}{2^{2\mathbf{d}+1}} \right] \\
\Rightarrow \Pr_s \left[\Pr_{s'}[p_h(s') = p_h(s) \mid p_e(s') = p_e(s)] \geq \frac{1}{2^{2\mathbf{d}+1}} \right] \\
& = 1 - \Pr_s \left[\Pr_{s'}[p_h(s') = p_h(s) \mid p_e(s') = p_e(s)] < \frac{1}{2^{2\mathbf{d}+1}} \right] \\
& > 1 - \frac{2 \cdot \mathbf{d}}{2\mathbf{d} + 1}
\end{aligned}$$

which proves the statement.

Appendix B Proof of Theorem 5.1 (F is a weak OWF)

In order to prove Theorem 5.1, we show that F is a weak OWF with inversion probability $1 - 1/2\mathbf{p}(m)$ with all but small constant probability over $D_{\mathbf{p}}$. Namely, we show that for all polynomials \mathbf{p} , for all poly-query $\mathcal{A}^{\mathbf{F}, \text{PSPACE}, \text{INV}}$, for all adversaries \mathcal{R} making polynomially many (in m) queries to the oracles $F, \text{PSPACE}, \text{INV}, \mathcal{A}^{\mathbf{F}, \text{PSPACE}, \text{INV}}$, the probability

$$\Pr_{F, \text{PSPACE}, \text{INV} \leftarrow \mathfrak{s} D_{\mathbf{p}}} \left[\text{Succlnv}_{\mathcal{A}, \mathcal{R}}^{\mathbf{F}, \text{INV}} \geq 1 - \frac{1}{2\mathbf{p}(m)} \right] \leq 1/\mathbf{c}, \quad (2.23)$$

where $\text{Succlnv}_{\mathcal{A}, \mathcal{R}}^{\mathbf{F}, \text{INV}}$ is defined as

$$\Pr_{x \leftarrow \mathfrak{s} \{0,1\}^m, \mathcal{R}, \mathcal{A}} \left[\mathcal{R}^{\mathbf{F}, \text{PSPACE}, \text{INV}, \mathcal{A}^{\mathbf{F}, \text{PSPACE}, \text{INV}}} (1^m, F(x)) \in F^{-1}(F(x)) \right].$$

Proof (Proof of Theorem 5.1). Fix \mathbf{p} , \mathcal{R} and \mathcal{A} . Since \mathcal{A} and \mathcal{R} both make polynomially many queries to the same oracles, \mathcal{R} can simply simulate \mathcal{A} . Thus, w.l.o.g., we can assume that \mathcal{R} only makes queries to F, PSPACE and INV . Additionally, we consider \mathcal{R} to be a computationally unbounded algorithm so that w.l.o.g., we can assume that it does not make queries to the PSPACE oracle.

Let q be a polynomial such that adversary \mathcal{R} makes exactly $q(m)$ queries to the oracle F and an arbitrary number of queries to INV . Since we let the adversary \mathcal{R} make an arbitrary number of queries to INV , that is, the adversary can be assumed to know the sets $\text{EASY}_{\text{in}}^m$ and $\text{EASY}_{\text{out}}^m$ and how F maps $\text{EASY}_{\text{in}}^m$ to $\text{EASY}_{\text{out}}^m$ completely. This only makes the adversary stronger. Importantly, using INV does not give the adversary any information on F on the *hard* values (only the fact that the values are hard, but no information on how $\text{HARD}_{\text{in}}^m$ maps to $\text{HARD}_{\text{out}}^m$).

For conciseness, we now denote $\mathcal{R} := \mathcal{R}^{\mathbf{F}, \text{PSPACE}, \text{INV}, \mathcal{A}^{\mathbf{F}, \text{PSPACE}, \text{INV}}}$ and we omit the PSPACE oracle everywhere—the PSPACE oracle is deterministic anyway. Moreover, we denote \mathcal{R} 's queries to \mathbf{F} by $x_1, \dots, x_{q(m)}$ and the \mathcal{R} 's guess for the pre-image of its input y by $x_{q(m)+1}$.

Eventually we want to bound the following:

$$\begin{aligned} & \Pr_{\mathbf{F}, \text{INV}} \left[\text{SuccInv}_{\mathcal{A}, \mathcal{R}}^{\mathbf{F}, \text{INV}} \geq 1 - \frac{1}{2^{\mathbf{p}(m)}} \right] \\ &= \Pr_{\mathbf{F}, \text{INV}} \left[\Pr_{x, \mathcal{R}} [\mathcal{R}(\mathbf{F}(x)) \in \mathbf{F}^{-1}(\mathbf{F}(x))] \geq 1 - \frac{1}{2^{\mathbf{p}(m)}} \right] \\ &\leq \Pr_{\mathbf{F}, \text{INV}} \left[\Pr_{x, \mathcal{R}} [\mathcal{R}(\mathbf{F}(x)) \in \mathbf{F}^{-1}(\mathbf{F}(x)) \mid x \notin \text{EASY}_{\text{in}}^m] \geq \frac{1}{2^{\mathbf{p}(m)}} \right] \end{aligned} \quad (2.24)$$

where the last inequality follows from

$$\begin{aligned} & \Pr_{x, \mathcal{R}} [\mathcal{R}(\mathbf{F}(x)) \in \mathbf{F}^{-1}(\mathbf{F}(x))] \\ &= \Pr_{x, \mathcal{R}} [\mathcal{R}(\mathbf{F}(x)) \in \mathbf{F}^{-1}(\mathbf{F}(x)) \mid x \notin \text{EASY}_{\text{in}}^m] \underbrace{\Pr_{x, \mathcal{R}} [x \notin \text{EASY}_{\text{in}}^m]}_{< 1} \\ &\quad + \underbrace{\Pr_{x, \mathcal{R}} [\mathcal{R}(\mathbf{F}(x)) \in \mathbf{F}^{-1}(\mathbf{F}(x)) \mid x \in \text{EASY}_{\text{in}}^m]}_{\leq 1} \underbrace{\Pr_{x, \mathcal{R}} [x \in \text{EASY}_{\text{in}}^m]}_{= 1 - 1/2^{\mathbf{p}(m)}}. \end{aligned}$$

We first compute a bound when sampling oracles \mathbf{F} , INV , input x and randomness for \mathcal{R} together and then deduce (2.24) via averaging. In order to bound the probability of \mathcal{R} successfully inverting, given that x is not in the easy set, we first bound the successful inversion event by the event that for any $i \leq q + 1$, $\mathbf{F}(x_i) = \mathbf{F}(x)$ (and not just for x_{q+1}). Equation (+) then splits the big OR into $q + 1$ disjoint events.

$$\begin{aligned} & \Pr_{\mathbf{F}, \text{INV}, x, \mathcal{R}} [\mathcal{R}(\mathbf{F}(x)) \in \mathbf{F}^{-1}(\mathbf{F}(x)) \mid x \notin \text{EASY}_{\text{in}}^m] \\ &\leq \Pr_{\mathbf{F}, \text{INV}, x, \mathcal{R}} \left[\bigvee_{1 \leq i \leq q+1} \mathbf{F}(x_i) = \mathbf{F}(x) \mid x \notin \text{EASY}_{\text{in}}^m \right] \\ &\stackrel{(+)}{=} \sum_{i=1}^{q(m)+1} \Pr_{\mathbf{F}, \text{INV}, x, \mathcal{R}} \left[\mathbf{F}(x_i) = \mathbf{F}(x) \mid \begin{array}{l} \mathbf{F}(x_1), \dots, \mathbf{F}(x_{i-1}) \neq \mathbf{F}(x) \wedge \\ x \notin \text{EASY}_{\text{in}}^m \end{array} \right] \\ &\quad \cdot \Pr_{\mathbf{F}, \text{INV}, x, \mathcal{R}} [\mathbf{F}(x_1), \dots, \mathbf{F}(x_{i-1}) \neq \mathbf{F}(x) \mid x \notin \text{EASY}_{\text{in}}^m] \\ &\leq \sum_{i=1}^{q(m)+1} \Pr_{\mathbf{F}, \text{INV}, x, \mathcal{R}} \left[\mathbf{F}(x_i) = \mathbf{F}(x) \mid \begin{array}{l} \mathbf{F}(x_1), \dots, \mathbf{F}(x_{i-1}) \neq \mathbf{F}(x) \wedge \\ x \notin \text{EASY}_{\text{in}}^m \end{array} \right] \end{aligned} \quad (2.25)$$

Finally, the probabilities are maximized when all the x_i are distinct and happen to lie in $\{0, 1\}^m \setminus \text{EASY}_{\text{in}}^m$ which is a set of size $\frac{1}{2^{\mathbf{p}(m)}} 2^m$. The function \mathbf{F} induces a bijection from $\{0, 1\}^m \setminus \text{EASY}_{\text{in}}^m$ to $\{0, 1\}^m \setminus \text{EASY}_{\text{out}}^m$ and thus, for the $i + 1$ -th query, we need to reduce the set size of $\{0, 1\}^m \setminus \text{EASY}_{\text{in}}^m$ by i , since—in the worst case—we already know i values—note that the sum now goes from 0 to q instead of 1 to $q + 1$. Hence, the probability that $\mathbf{F}(x_{i+1}) = \mathbf{F}(x)$ is upper bounded by 1 divided by $\frac{1}{2^{\mathbf{p}(m)}} 2^m - i$. Thus, we can upper bound (2.25) by the sum

$$\Pr_{\mathbf{F}, \text{INV}, x, \mathcal{R}} [\mathcal{R}(\mathbf{F}(x)) \in \mathbf{F}^{-1}(\mathbf{F}(x)) \mid x \notin \text{EASY}_{\text{in}}^m] \leq \sum_{i=0}^{q(m)} \frac{1}{\frac{1}{2^{\mathbf{p}(m)}} 2^m - i}. \quad (2.26)$$

In order to derive an upper bound for (2.24), we need to split the probability into sampling \mathbf{F} , INV and sampling x, \mathcal{R} .

By Markov's inequality, for any real number r ,

$$\begin{aligned}
& \Pr_{\mathbf{F}, \text{INV}} \left[\Pr_{x, \mathcal{R}} \left[\mathcal{R}(F(x)) \in F^{-1}(F(x)) \mid x \notin \text{EASY}_{\text{in}}^m \right] \geq r \right] \\
& \leq \frac{\mathbb{E}_{\mathbf{F}, \text{INV}} \left[\Pr_{x, \mathcal{R}} \left[\mathcal{R}(F(x)) \in F^{-1}(F(x)) \mid x \notin \text{EASY}_{\text{in}}^m \right] \right]}{r} \\
& = \frac{\Pr_{\mathbf{F}, \text{INV}, x, \mathcal{R}} \left[\mathcal{R}(F(x)) \in F^{-1}(F(x)) \mid x \notin \text{EASY}_{\text{in}}^m \right]}{r} \quad \text{|by def of expectation} \\
& \leq \frac{\sum_{i=0}^{q(m)} \frac{1}{\mathbf{p}(m) 2^{m-i}}}{r} \quad \text{|by (2.26)}
\end{aligned}$$

Now we plug in the correct value $r = \frac{1}{2\mathbf{p}(m)}$ from (2.24) and get

$$\frac{\sum_{i=0}^{q(m)} \frac{1}{\mathbf{p}(m) 2^{m-i}}}{\frac{1}{2\mathbf{p}(m)}} = \sum_{i=0}^{q(m)} \frac{2\mathbf{p}(m)}{\mathbf{p}(m) 2^{m-i}} \leq \sum_{i=0}^{q(m)} \frac{2^{m/4}}{2^{m/2}} = q(m) 2^{m/4} = \text{negl}(m) < \frac{1}{\mathbf{c}}$$

where the first inequality holds for any big enough m and the last inequality holds for any big enough m since \mathbf{c} is a constant.

On the Difference between Distributional and Existential Collision-Resistance

Chris Brzuska¹ Christoph Egger² Pihla Karanko¹ Felix Rohrbach³

¹ Aalto University, Finland, [chris.brzuska,pihla.karanko}@aalto.fi](mailto:{chris.brzuska,pihla.karanko}@aalto.fi)

² Chalmers University of Technology and University of Gothenburg, Sweden, christoph.egger@chalmers.se

³ TU Darmstadt, Germany, felix.rohrbach@cryptoplexity.de

Abstract

Distributional collision-resistant hash functions (dCRH) are a relaxation of (existential) collision-resistant hash-functions (CRH), since a dCRH adversary is only successful if it produces collisions (w, w_{coll}) , where w is close to uniformly distributed and w_{coll} is almost uniform amongst all collisions with w . dCRH is a natural intermediate security notion between one-wayness and CRH and indeed suffices to build constant-round statistically hiding commitments which we cannot build from one-way functions alone.

We initiate the study of the relation between distributional and existential collision resistance by ruling out black-box constructions of (salted) CRH from dCRH, which can be proven via a reduction which makes a single query to the CRH adversary, followed by an arbitrary polynomial number of queries to the dCRH primitive.

In essence, our oracle consists of a random 2-to-1 function F out of which a fraction of collisions is marked as *hard*. In order to ensure that F remains a dCRH in the presence of a collision-finder which breaks any CRH h^F , we define the collision-finder to return a collision (w, w_{coll}) for h^F which does not induce a collision in the hard set of F . The core difficulty in the proof is to argue that such a *good* collision (w, w_{coll}) exists with high probability over F and the choice of the hard set.

We sample $2^{2|w|}$ random collisions (w^i, w_{coll}^i) on h^F and argue that over the choice of F and the hard set, each has probability $1/4 \cdot 2^{-|w|}$ of being good. Thus, in expectation, there should be $1/4 \cdot 2^{-|w|} 2^{2|w|} = 1/4 \cdot 2^{|w|}$ many good collisions—and we only need a single one. However, computing the tail bound for this sum of random variables X_i , which are 1 if the i th collision (w^i, w_{coll}^i) is good and 0 otherwise, is non-trivial, since they are not even *pairwise* independent. The analysis of this tail bound is the main technical novelty of our work.

1 Introduction

A collision-resistant hash-function (CRH) h is a compressing function which makes it hard to find collisions $x \neq x'$ such that $h(x) = h(x')$. CRHs are used, e.g., to hash the transcript into the key in the TLS handshake protocol [Res18] or to hash messages before signing them [HK06; Mir06]. Interestingly, CRHs do not fit into Impagliazzo’s linear hierarchy of cryptographic primitives [Imp95]:

in his seminal work, Simon [Sim98] shows that, while CRHs imply one-way functions (OWFs), OWFs do not imply CRHs in a black-box way. Simon designs two oracles, a random compressing function F , and a breaker oracle $\text{COL}_{h,n}$ that gives a random collision for any candidate CRH construction h (cf. Fig. 1.2). Relative to F and $\text{COL}_{h,n}$, he then shows that OWFs exist, but CRHs do not, ruling out any relativizing construction of CRH from OWF. In fact, Haitner, Hoch, Reingold, and Segev [HHS07] even show that CRH are (black-box) *incomparable* to public-key encryption. Therefore, CRH lie neither in Minicrypt, nor in Cryptomania [Imp95], cf. Fig. 1.1.

A technical-conceptual nuisance of CRHs is their inability to achieve concrete or non-uniform security. Namely, an adversary with a hard-coded collision (in its code or in its non-uniform advice) efficiently breaks collision-resistance, see [Rog06; BL13; BBKPV16] for a discussion. *Salted* collision-resistance avoids these definitional difficulties and also provides better practical security against so-called *rainbow* attacks [Oec03].

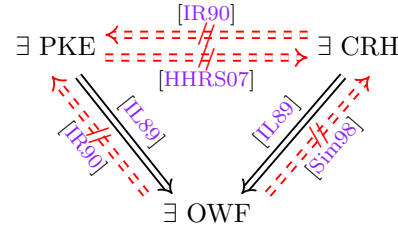


FIGURE 1.1 — PKE vs. CRH

Definition 1.1 (Salted CRH). We say that a polynomial-time computable function h with $\forall x \in \{0, 1\}^n : |h(x)| \leq n - 1$ is a *collision resistant hash-function (CRH)* if for all PPT \mathcal{A} :

$$\Pr_{\text{slt} \leftarrow \{0,1\}^n} [(x, x') \leftarrow \mathcal{A}(1^n, \text{slt}) : h_{\text{slt}}(x) = h_{\text{slt}}(x'), x \neq x']$$

is negligible in n .

1.1 Distributional collision-resistance

In this work, we focus on *distributional* collision-resistant hash-functions (dCRH) which require an adversary to come up with not just any collision, but with a (*marginally*) *uniform* one, i.e., distributed as Simon’s oracle, which samples a random input and then a random second pre-image (including possible pseudo-collisions). We will call this type of sampling a *Simon-random collision*.¹

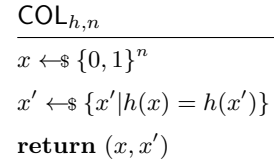


FIGURE 1.2 — $\text{COL}_{h,n}$

dCRH were introduced by Dubrov and Ishai (DI [DI06]) as a relaxation of CRH. An adversary who finds a random collision also breaks (existential) collision-resistance, but the converse is not necessarily true: if h is a dCRH, then $h'(x||b) := h(x)$ is also a dCRH, but h' is not a CRH since for any bitstring x , we have that $h'(x||0) = h'(x||1)$. A qualitatively interesting difference is that unlike CRH, *unsalted* dCRH are a meaningful definition against concrete and non-uniform adversaries (with arbitrary polynomial-size advice), since hardcoding a finite set of collisions does not allow to sample Simon-random ones. Unsalted dCRH are a slightly stronger notion than salted dCRH, see Appendix C for a discussion. Since we show oracle separations for constructions *from* dCRH in this paper, we consider the following simpler and slightly stronger *unsalted* dCRH definition which only makes our results stronger.

¹Note that this type of sampling a collision is different from choosing uniformly from the set of all collisions. Indeed, the choice of sampling collisions is crucial in Simon’s separation [Sim98], see also [Bae14] for a counter example.

Definition 1.2 (Unsalted dCRH). We say that a polynomial-time computable function h with $\forall x \in \{0, 1\}^n : |h(x)| \leq n - 1$ is an (*unsalted*) *distributional CRH* if there exists a polynomial $p(n)$ such that for all PPT \mathcal{A} :

$$\text{SD}(\mathcal{A}(1^n), \text{COL}_{h,n}) \geq \frac{1}{p(n)},$$

where Fig. 1.2 defines $\text{COL}_{h,n}$.

Distributional collision-resistance naturally implies one-way functions, since if a dCRH is not a *distributional* one-way function, we can sample almost Simon-random collisions by sampling uniformly random pre-images, and distributional OWFs are known to imply OWFs [IL89]. A priori, it is not obvious whether dCRHs are *more useful* than OWFs. Recently, however, Bitansky, Haitner, Komargodski and Yogev (BHKY [BHKY19]) built constant-round statistically hiding commitments from dCRH, which we cannot construct from OWFs in a black-box way [HHRS07]. In turn, dCRHs also seem to be easier to construct than CRHs: Komargodski and Yogev (KY [KY18]) construct dCRH from average-case hard statistical zero-knowledge (SZK), which is not known to yield CRH. Thus, dCRH are not only a natural relaxation of CRH, they also might constitute a useful intermediate notion between OWFs and CRH.

Note that DI [DI06] originally defined dCRH so that an adversary cannot sample *negligibly* close to Simon-random. BHKY [BHKY19] observe that this definition is not known to imply OWFs since distributional one-way functions require a *fixed* inverse polynomial threshold rather than an adversary-dependent threshold. BHKY thus use a dCRH variant with *fixed* inverse polynomial threshold. In this paper, we also focus on the BHKY definition of dCRH (Definition 1.2) and sometimes refer to it as $\frac{1}{\text{poly}}$ -dCRH for clarity. When we consider negligibly close sampling, we refer to this notion as *negl*-dCRH.

1.2 Multi-collision resistance

An alternative way to relax collision resistance is to require the adversary to find $c > 2$ many distinct x_1, \dots, x_c such that for all $1 \leq i, j \leq c$, $h(x_i) = h(x_j)$. *Multi-collision resistant hash-functions (MCRHs)* were introduced more recently by both Komargodski, Naor and Yogev (KNY [KNY18]) and, concurrently, by Bitansky, Tauman Kalai and Paneth (BKP [BKP18]). BKP show that MCRHs can provide meaningful security against adversaries with *a priori fixed* polynomial-size non-uniform advice, since a non-uniform adversary cannot find significantly more colliding values than the size of its advice. This insight enables BKP to construct 3-message zero knowledge arguments against arbitrary polynomial-size non-uniform adversaries from MCRHs, a longstanding open problem [HT98].

A further interesting weakening of CRH are universal one-way hash functions (UOWHF) introduced by Naor and Yung [NY89], where the adversary has to find a collision *with a particular* value x which the adversary chooses before seeing the salt. Rompel [Rom90] shows that UOWHFs are equivalent to OWFs, subsuming them in Minicrypt.

1.3 Relations between different notions of collision resistance

A CRH is also a (*negl*-)dCRH as well as a c -MCRH for any $c > 2$ and thus

$$\text{CRH} \Rightarrow \frac{1}{\text{poly}}\text{-dCRH} \Rightarrow \text{negl-dCRH}$$

CRH \Rightarrow MCRH.

Can we construct a CRH from any of the weaker notions? How do dCRH and MCRH relate to one another? Komargodski and Yagev (KY [KY18]) show that surprisingly, 3-MCRHs imply dCRHs. Using a clever non-black-box construction which makes calls to an abstract *adversary* as well as additional ideas from coding theory, Rothblum and Vasudevan (RV [RV22]) are able to use a worst-case collision sampler (rather than a random collision sampler as KY) and construct CRHs from c -MCRHs for $c \in \{3, 4\}$. This result was recently extended by Buzek and Tessaro [BT24] to all constant c . Furthermore, Simon’s [Sim98] separation between OWF and CRH is also a separation between OWF and *distributional* CRH: Simon’s breaker oracle returns a collision for each candidate CRH function and indeed produces a perfectly Simon-random collision and therefore also breaks any candidate (negl-)dCRH.

In conclusion, we now have two conceptually intriguing relaxations of CRH. Firstly, we have c -MCRH, which is quite close to CRH: For constant parameters, there are known transformations to CRH, and it is quite plausible that the two definitions are even equivalent, meaning that it is probably hard to build MCRH from anything that doesn’t already imply CRH as well. Secondly, we have dCRH, which is known to be a stronger assumption than OWFs, but its relation to CRH is not clear yet.

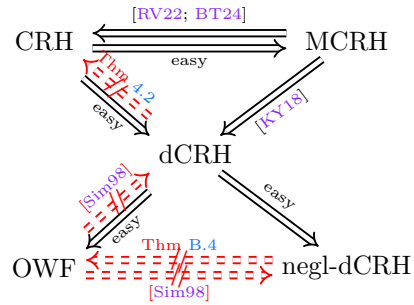


FIGURE 1.3 — Implications and separations for collision resistance notions.

1.4 Our contribution

We initiate the study of separating CRH from dCRH. A full separation would show that there exists no relativizing construction of (salted) CRH from (unsalted) *distributional* CRH. We show a result along these lines, but need to add three restrictions. We first state the restrictions and then turn to the interpretation of our partial separation. See Section 2 for a discussion of how we use the three restrictions in our proof.

1-ColFind-poly-F. Similar to Simon’s separation of CRH from OWF [Sim98], we design a function oracle F (dCRH) and a breaker oracle ColFind which breaks any CRH. We now model the reduction \mathcal{A} who tries to break the dCRH as making one query to the breaker oracle ColFind in the beginning, and only afterwards repeated queries to F , with no further ColFind queries. This captures, in particular, natural reduction approaches, where the reduction \mathcal{A} queries some h^F to its CRH collision-finder, obtains (w, w_{coll}) and then returns an F -collision which is *induced* by (w, w_{coll}) in the sense that the set of queries $\text{qry}_h^F(w)$ and $\text{qry}_h^F(w_{\text{coll}})$ contain $x \neq x'$ such that $F(x) \neq F(x')$. Analogously, in a recent work, Folwarczný, Göös, Hubáček, Maystre, and Yuan [FGHMY24] show that it is impossible to have a black-box reduction which shows the hardness of a total NP search problems based on an OWF as long as the reduction queries the adversary only once and the OWF only afterwards.

Construction h without ColFind queries. The oracle ColFind only takes candidate CRH constructions h^F as input rather than constructions $h^{F, \text{ColFind}}$ (which would require a recursive definition

of ColFind). Disallowing ColFind queries in h is, in fact, simply consistent with the previous restriction of disallowing the reduction to make F-queries before its ColFind query since else, the reduction \mathcal{A} could obtain a similar effect by asking a (non-restricted) $h^{\text{F,ColFind}}$ to ColFind.

Disallowing constructions to call adversaries is a common restriction which was pioneered by Hsiao and Reyzin [HR04] and the resulting impossibility result is often referred to as *two-oracle-style* separation. In the language of Reingold, Trevisan and Vadhan [RTV04], a two-oracle style separation only rules out *fully black-box* constructions rather than all *relativizing* reductions.

Polynomial honesty and fixed query-length. We require constructions h^{F} and adversaries \mathcal{A} to be *polynomially honest*, i.e., the input to any oracle call is required to be polynomially related to the security parameter which disallows tiny queries, e.g., of constant or logarithmic size in the security parameter. Moreover, we demand that the construction h^{F} only makes F queries of a *single* input-length. The latter restriction to a fixed query-length is merely for the sake of convenience; our techniques could handle queries with mixed input-lengths as well at the cost of a more technical presentation. The restriction to polynomially honest constructions additionally simplifies the proof, but here, we cannot argue that our techniques still apply. Polynomial honesty is a notion which has recently been used by Saks and Santhanam [SS22], analysing randomized reductions to the problem of approximating Kolmogorov complexity. The assumption of polynomial honesty means that complexity leveraging is not excluded by our separation.

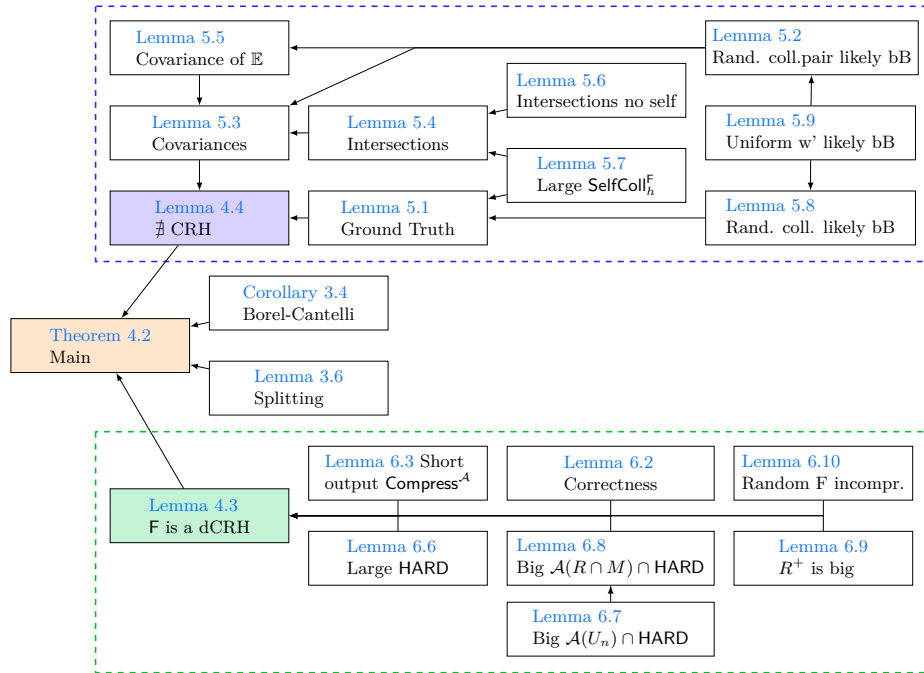
Definition 1.3 (Polynomial Honesty). A uniform or non-uniform sequence of oracle circuits h_λ is *polynomially honest* if there exists a constant c such that for all inputs $w \in \{0, 1\}^\lambda$, h_λ only makes oracle queries of size at most $|w|^c$.

Definition 1.4 (Fixed Query-Length). We say that a construction h^{F} with oracle access to F is *fixed query-length* if there is a single length n such that all queries to F are of length n .

Theorem 1.5 (dCRH $\not\approx$ CRH). *Let $p \geq 4$ be a polynomial. There is no fully black-box, polynomially honest, fixed query-length construction of (salted) CRH from (unsalted) $\frac{1}{p}$ -dCRH with a black-box reduction which makes a single query to the CRH adversary, followed by a polynomial number of queries to the dCRH.*

Interpretation. We can interpret Theorem 1.5 in two ways. Theorem 1.5 covers *arbitrary* black-box constructions, which can make any polynomial number of queries with any polynomial depth of adaptivity, as long as the queries are of a fixed, sufficiently large input length. Moreover, the class of reductions we rule out is rather large, too: Reductions for collision-resistance such as proving the collision-resistance of Merkle trees from the collision-resistance of the underlying compression function tend to call the adversary only once and do not call the compression function before calling the adversary—they might call the compression function *after* calling the adversary though, e.g., to evaluate the construction on the two obtained inputs (w, w_{coll}) . Thus, in terms of black-box constructions with commonly used styles of reductions, Theorem 1.5 can be viewed as a rather strong separation result.

At the same time, our result does not rule out constructions which use the adversary itself such as the construction by Rothblum and Vasudevan (RV [RV22]), which builds CRHs from c -MCRHs for $c \in \{3, 4\}$. Thus, an alternative, more optimistic interpretation of our result could be that it provides

FIGURE 2.4 — Proof tree for [Theorem 4.2](#)

guidance on how to obtain a *positive* result: The reduction for the adversary-based construction by RV succeeds to obtain 4 values mapping to the same output y from an adversary only returning 2 colliding values by ensuring that each of the 2 values induces 2 colliding values so that (a) all 4 map to the same y and (b) all of them are distinct with noticeable probability. The coding theory techniques which RV employ essentially enforce “well-distributedness” of the collisions. Thus, one might optimistically suspect that analogous techniques or perhaps even the RV construction itself works to build CRH from dCRH. Unfortunately, the latter hope seems too optimistic: The RV construction only makes a constant number of queries to the underlying hash-function F , and such constructions seems subject to separation arguments as well, see the beginning of [Section 2.2](#) (Example II) for a discussion of constant-query constructions.

On *negl*-dCRH. As a second minor contribution, we show that indeed, OWFs cannot be built from *negligible* distributional collision-resistance, justifying BHKY’s (and our) decision to focus on *1/poly* dCRH.

Theorem 1.6 (*negl*-dCRH $\not\Rightarrow$ OWF). *There exists no fully black-box construction of OWF from *negl*-dCRH to OWF.*

See [Appendix B](#) for details.

2 Technical overview

In order to show our main separation $\text{dCRH} \not\Rightarrow \text{CRH}$, we design two oracles F and ColFind such that relative to them, a dCRH exists, but CRHs do not. We define $F : \{0, 1\}^n \rightarrow \{0, 1\}^{n-1}$ as a random

2-regular function, i.e., every image has exactly two pre-images. We now add an oracle ColFind which finds *some* collision for every hash-function h^F , but without enabling an adversary to find close to *Simon-random* collisions for F . Concretely, we label one half of the outputs by F as HARD and define ColFind to return the smallest collision (w, w_{coll}) on h^F (according to a random order) such that (w, w_{coll}) does not *induce* a hard collision in F , i.e., the union of the set of queries $\text{qry}_h^F(w)$ which h makes to F on input w and the analogous set $\text{qry}_h^F(w_{\text{coll}})$ together do not contain $x \neq x_{\text{coll}}$ such that $F(x) = F(x_{\text{coll}}) = y \in \text{HARD}$. See [Figure 4.5](#) for a formal description of this distribution.

Our oracle separation now consists of two main parts: Relative to the oracles, we show that first, F is a dCRH, and second, that there exists no construction h^F of an (existential) collision-resistant hash function. In [Section 2.1](#), we outline our proof that F is a dCRH relative to ColFind using a compression argument, and in [Section 2.2](#), we provide an overview over our rather involved proof that for most hash-functions h^F , ColFind can indeed return (w, w_{coll}) without inducing an F -collision in the hard set. [Figure 2.4](#) gives a graphical outline of the full proof.

Note that we will follow the usual pattern for oracle separations: We will first prove our result over a random choice of oracles, and then use Borel-Cantelli ([Lemma 3.3](#)) to fix one oracle. As we consider the latter step rather less interesting from a technical point of view, we will not focus on it in this technical overview and refer the reader to [Section 4](#) (as well as [Appendix A](#) for the proof of [Theorem 4.2](#)) for more details.

2.1 F is a dCRH

To show that F is a dCRH, relative to F and ColFind , we show that PPT adversaries $\mathcal{A}^{F, \text{ColFind}}$ are indeed unlikely to find collisions $x \neq x_{\text{coll}}$ such that $F(x) = F(x_{\text{coll}}) = y \in \text{HARD}$ and thus have statistical distance at least $\frac{1}{4}$ from $\text{COL}_{F,n}$ (cf. [Figure 1.2](#)). The implication that we can find an F such that this holds for *all* uniform PPT adversaries then follows by a standard Borel-Cantelli argument (cf. [Appendix A](#)).

Lemma 2.1 ([Lemma 4.3](#), informal). *For all 1-ColFind-poly- F adversaries and large enough n , the statistical distance between the adversary and Simon-random collisions is more than $1/4$ with high probability.*

$$\Pr[\text{SD}(\mathcal{A}^{F, \text{ColFind}}(1^n), \text{COL}_{F,n}) \geq 1/4] \geq 1 - \frac{1}{n^2},$$

Our proof proceeds via the Gennaro-Trevisan compression paradigm [[GT00](#)]. We assume towards contradiction that there is a uniform PPT adversary $\mathcal{A}^{F, \text{ColFind}}$ which finds sufficiently random F -collisions (x, x_{coll}) in HARD with good probability. For all collisions (x, x_{coll}) which \mathcal{A} finds, it suffices to store $x \mapsto y$ while $x_{\text{coll}} \mapsto y$ can be restored by re-running \mathcal{A} on the same randomness. Thereby, \mathcal{A} allows us to encode F more succinctly than is possible information-theoretically, and we obtain a contradiction.

Encoding randomness. The main technical challenge is to identify and encode the random tapes on which the adversary returns a *fresh* collision (x, x_{coll}) which allows us to compress. Unlike Gennaro and Trevisan, we cannot derandomize the adversary, since our adversary approximates the high-entropy *distribution* $\text{COL}_{F,n}$, and we can also not encode the randomness explicitly, since it might be longer than the n bits we are trying to save. Instead, compressor and decompressor share a *large, random string*, independently of F , and representing a list of 2^n random tapes. We can then encode the *index set* S within these 2^n tapes where the adversary compresses instances which

requires $\log \binom{2^n}{|S|} \approx |S|(n - \log|S|)$ many bits and thus, if $|S|$ is sufficiently large, much less than $n|S|$.

Large set S . In order to show that a large enough set S exists, we use that the adversary returns almost random collisions from $\text{COL}_{F,n}$ and that with inverse polynomial probability $1/\text{poly}(n)$ a random collision is one that we have not seen before. A Chernoff bound then establishes that after $n^2 \text{poly}(n)$ many randomly sampled tapes, it is very likely that one of the tapes hits a collision amongst the polynomial fraction we have not covered yet.

Using the restrictions. Our compression argument uses the restriction that \mathcal{A} cannot make F queries before the single ColFind query. Concretely, if on input r , the adversary \mathcal{A} returns an F -collision (x, x_{coll}) with $F(x) \in \text{HARD}$, then we need to argue that we can emulate the run of \mathcal{A} without knowing *both* x and x_{coll} (Knowing one of them is OK, we can still compress in this case.). If, say, x is induced by the ColFind 's answer (w, w_{coll}) and a later F query “hits” x_{coll} , we can still compress by halting the emulation when the answer to a query is not known (since this gives x_{coll}): In this case, we would encode the index of the previous F query which hit x (possibly pointing to the index in the query sets $\text{qry}_h^F(w)$ and $\text{qry}_h^F(w_{\text{coll}})$ or simply by assuming w.l.o.g. that the reduction makes all queries in $\text{qry}_h^F(w)$ and $\text{qry}_h^F(w_{\text{coll}})$ also to F directly after the ColFind query) and thereby making the mapping $x_{\text{coll}} \mapsto f(x)$ recoverable. However, when a hit on x_{coll} is first made in the F query and then x is queried within ColFind , i.e., in $\text{qry}_h^F(w)$ and $\text{qry}_h^F(w_{\text{coll}})$, then the decoder might also notice that it does not know a valid answer to the ColFind query, but it might have multiple choices for x_{coll} which would all induce valid ColFind answers. See Appendix D for such an example. Analogously, as mentioned before, excluding recursive calls to ColFind is necessary for the same reason.

Our compression argument uses neither polynomial honesty nor the fact that h^F only makes queries on a fixed query-length: Since \mathcal{A} is efficient, it cannot ask functions h^F with an input length $|w|$ that is more than polynomially larger than the input length n of F for which \mathcal{A} is supposed to find collisions. Moreover, giving a description of F for many other values $n' \neq n$ just constitutes more shared randomness between compressor and decompressor which does not count towards the length of the encoding.

See Section 4.3 for a full proof of the compression argument. We now turn to the somewhat involved proof that ColFind can indeed return an h^F -collision (w, w_{coll}) without inducing a hard collision on F .

2.2 No CRH relative to F , ColFind

The second—and much more difficult—lemma establishes that ColFind returns a collision for any construction h^F with sufficiently high probability.

Lemma 2.2 (Lemma 4.4, informal). *For all non-uniform, polynomial-time, polynomially honest and fixed query-length functions $h^F : \{0, 1\}^* \rightarrow \{0, 1\}^{*-1}$ and all large enough input lengths λ , ColFind returns a collision with high probability.*

Technically, we here prove that there exists an h^F collision (w, w_{coll}) such that their query set $\text{qry}_h^F(w)$ and $\text{qry}_h^F(w_{\text{coll}})$ does not contain a hard collision on F . Note that we consider h^F as a circuit (for a specific input length), i.e., we are in a non-uniform setting here and therefore don't need to

salt the hash functions. We will later show in the proof of the main theorem (Appendix A) that we can use Lemma 4.4 to break any candidate *uniform* but *salted* hash function h .

We will now start with two examples of h where it is easy to find a collision ColFind can return, and then turn to general constructions of h .

Example I. Consider, for example, the construction h^F that maps $w = x_1 || \dots || x_t$ to $F(x_1) || \dots || F(x_t)$. As F is 2-regular, a random collision w_{coll} will induce a collision on half of the x_i , and likely, for one of them, $F(x_i) \in \text{HARD}$. However, ColFind can choose w_{coll} such that it induces an F -collision on only one of the $x_i \notin \text{HARD}$ or even choose (w, w_{coll}) such that h^F only makes queries outside of HARD.

Example II. An even easier example are constructions h^F which only make a *constant* c number of queries to F (cf. the discussion of RV in Section 1.4). If we sample a random w and a random collision w_{coll} with w , then there are $2c$ queries which h^F makes to F on inputs w and w_{coll} . Over the choice of the hard set, the probability of all of them being easy is 2^{-2c} , which is still a constant. Since the hard set is chosen independently for each input length, ColFind would break h^F on a 2^{-2c} fraction of all input lengths in expectation (and with good tail bounds) and thus for infinitely many input length. (This argument would even apply to recursive constructions with a constant number of queries, such as RV).

Arbitrary Constructions h^F . A variant of the argument for Example I also applies to *arbitrary* constructions h^F . Unfortunately, we cannot argue that there are w on which h^F makes only queries outside the hard set, since h^F might make many more queries to F than there is entropy in w , say $2|w|$ many and the probability that all of them are easy is $2^{-2|w|}$. A 1-bit-compressing CRH h^F might only have $2^{|w|}$ many collisions and thus, the expected number of h^F collisions which only make easy F queries is $2^{|w|} 2^{-2|w|} = 2^{-|w|}$, which is far away from 1 and, by Borel-Cantelli (cf. Lemma 3.3), the probability of this happening infinitely many times is 0.

Thus, we need to live with the fact that our construction h^F makes F -queries in the hard set on all inputs, but does this also mean that an h^F -collision (w, w_{coll}) necessarily induces F -collisions in the hard set? How many F -collisions does (w, w_{coll}) induce, on the average? Example I induces $t/2$ collisions on the average, meaning that a *random* collision (w, w_{coll}) for h^F induces only easy F collisions with probability $2^{-\frac{t}{2}}$. In turn, the entropy of w is much higher, namely $tn = |w|$. And since there must be at least $2^{|w|} = 2^{tn}$ many h^F -collisions², in expectation, $2^{-\frac{t}{2}} 2^{tn} = 2^{t(n-\frac{1}{2})}$ of the h^F collisions (w, w_{coll}) do not induce F -collisions (x, x_{coll}) in the hard set!

Therefore, if we can argue that the expected number of induced F -collisions is much smaller than $|w|$, it seems plausible that there exists one (or even many) collisions (w, w_{coll}) which do not induce hard collisions in F —at least in expectation. A back-of-the-envelope calculus suggests that conditioning on $h^F(w) = z$ should make at most $\approx \frac{|w|}{n}$ many queries likely, since z has at most as much entropy as $|w|$ (actually even 1 bit less). Indeed, Lemma 5.9 establishes that having $|w| \gg \frac{|z|}{|y|}$ collisions is extremely unlikely, allowing us to derive our Ground Truth Lemma (Lemma 5.1) which states that a random collision pair (w, w_{coll}) is “good” with probability at least $2^{-|w|}$.³

²In fact, Example I induces even more collisions since it compresses more than by 1 bit, but we want to lose this other lower bound so that this discussion prepares our readers for the analysis of the general case.

³The actual lower bound stated in Lemma 5.1 is lower by a factor $\frac{1}{4}$ which we lose in the proof by excluding

Expectation. The ground truth lemma implies that when sampling $2^{2|w|}$ collision pairs (w, w_{coll}) , in expectation, $2^{2|w|} \cdot 2^{-|w|}/4 = 2^{|w|}/4$ are good. We want to establish a tail bound which upper bounds the possibility that, despite this promising high expectation, none of the $2^{2|w|}$ randomly sampled collision pairs are good. Let us introduce random variables X_i , corresponding to collisions (w, w_{coll}) , which is 1 if the collision is good, i.e., it does not induce an F-collision in the hard set, and otherwise 0. Now, we want to show that it is quite unlikely that the sum over all X_i is 0. Unfortunately, strong tail bounds for the sum of random variables, such as the Chernoff bound, require these $2^{2|w|}$ random variables to be independent which they are not necessarily, as the following counterexample shows:

Counterexample. We construct a concrete h^F such that the (random) collisions are dependent. Let $w = z||b$ denote the input to h^F , where b is the last bit of w . Now, h^F queries $b^{|w|}$ to F , ignores the answer and then returns z , i.e., all bits of w except for the last bit of w . h^F -collisions are now of the form $(z||0, z||1)$. They will typically be good—unless $F(0^{|w|}) = F(1^{|w|}) \in \text{HARD}$. While this is unlikely, it shows that the random variables are dependent. If $(z||0, z||1)$ is good then so will be any other h^F -collision $(z'||0, z'||1)$. If $(z||0, z||1)$ is bad then so will be any other h^F -collision $(z'||0, z'||1)$.

Chebychev Bound. As we cannot use a Chernoff bound here, we will use a *Chebychev bound* instead. In cryptography, Chebychev bound is typically applied with respect to *pairwise* independent random variables, and the above counterexample shows that our random variables are not even pairwise independent. However, we can, in fact, use Chebychev bound for general random variables and in this case, we obtain an additional corrective term, the *co-variance* between two random variables (cf. Section 5.1). The co-variance $\text{Cov}(X_i, X_j)$ measures how much the random variables X_i and X_j are dependent. If the random variables X_i and X_j are independent, the co-variance is 0, but if $X_i = X_j$, then the co-variance equals the variance $\text{Var}(X_i)$. In summary, the general Chebychev bound is

$$\Pr\left[\sum X_i = 0\right] \leq \frac{1}{a^2} \left(\sum_i \text{Var}(X_i) + \sum_{i \neq j} \text{Cov}(X_i, X_j) \right), \quad (2.1)$$

where a is the expectation of $\sum_i X_i$. The term $\frac{1}{a^2} \sum_i \text{Var}(X_i)$ is the Chebychev bound that we'd obtain if X_i and X_j were actually pairwise independent (cf. Equation (5.4) in Section 5.1). We return to this term shortly, but first focus on how large the term $\frac{1}{a^2} \sum_{i \neq j} \text{Cov}(X_i, X_j)$ can become. Unfortunately, $\text{Cov}(X_i, X_j)$ is rather large when two collision pairs (w, w_{coll}) and (w', w'_{coll}) both induce one or more shared collisions (x, x_{coll}) on F (as in our above counterexample where the random variables become fully dependent). On the other hand, it is also quite unlikely that (w, w_{coll}) and (w', w'_{coll}) induce the *same* collision (x, x_{coll}) ; and the probability that the number of shared collisions is α decreases with α . Thus, our analysis splits $\text{Cov}(X_i, X_j)$ into multiple disjoint cases, depending on the intersection size α . We obtain a product between the probability of intersection size α and the covariance of X_i and X_j assuming that the intersection size is α , and we want to show that the (decreasing) probability and (increasing) co-variance counterbalance one another to yield an overall sufficiently small term.

the cases (1) $w = w_{\text{coll}}$ as well as (2) w and w_{coll} which induce “self-collisions”, i.e., the case where $\text{qry}_h^F(w)$ alone already contains an F collision (which is rather unlikely, yet possible). Handling collisions within $\text{qry}_h^F(w)$ and the case $w = w_{\text{coll}}$ are a technicalities which need to be carefully resolved, but are—in our opinion—not conceptually interesting. We thus omit details from the technical overview.

Padding X_i . In our analysis, it will be useful if the *expectation* of X_i is independent of the number of collisions α which X_i and X_j share. In general, this is not the case, because if X_i and X_j have an intersection size α , they will each have at least α many collisions (meaning that their expectation is at most $2^{-\alpha}$), while in principle, they could also have none (meaning that their expectation could be 1). Since we already know that each X_i has at most $|w|$ many collisions (except with tiny probability, cf. Lemma 5.2), we can make their expectation independent by adding “padding bits” so that if the number of collisions in X_i is t , we flip $|w| - t$ many padding bits and ask for these padding bits to be 1. Since F-collisions are in the hard set with probability $1/2$ and padding bits are 1 with probability $1/2$, we obtain an expectation of $2^{-|w|}$, regardless of how many collisions are in X_i and regardless of the intersection size between X_i and X_j —except for the event where one of them has more than $|w|$ many collisions, but this case is extremely unlikely (Lemma 5.2); the bound $|w|$ was chosen such that this bad event does not affect the analysis significantly.

Note the the above analysis treats co-variances as if they could just be split based on disjoint events, similar to the law of total probability. The analogous law of total covariance (cf. Equation 5.10), however, requires one more corrective term, which describes the co-variance of the expectations of the random variables *conditioned on the intersection size*. This now lets us split the covariance into three parts: The case where there is no intersection between the collisions, the case where there is an intersection and the corrective term.

$$\begin{aligned} \text{Cov}(X_i^F, X_j^F) &= \Pr[\#\text{Int}(i, j) = 0] \cdot \text{Cov}(X_i^F, X_j^F \mid \#\text{Int}(i, j) = 0) \\ &\quad + \sum_{\alpha=1}^q \Pr[\#\text{Int}(i, j) = \alpha] \cdot \text{Cov}(X_i^F, X_j^F \mid \#\text{Int}(i, j) = \alpha) \\ &\quad + \text{Cov}(\mathbb{E}[X_i^F \mid \#\text{INT}(i, j)], \mathbb{E}[X_j^F \mid \#\text{INT}(i, j)]) \end{aligned}$$

We can show that the first term is 0, as the two random variables are independent if the intersection is empty. See Section 5.1 for bounds on all terms – we will here bound them very roughly by

$$\lesssim \frac{1}{2^{n/2}} \mathbb{E}[X_1]^2. \quad (2.2)$$

A bound for ColFind. We can now combine these ideas to give a sufficiently strong tail bound for the probability that ColFind cannot find a good collision: We will first look at the variance term in (2.1). Here, we divide the sum over the variances by a^2 , where a is the sum over the expectations of X_i :

$$\frac{1}{a^2} \sum_i \text{Var}(X_i) = \frac{\sum_i \text{Var}(X_i)}{(\sum_i \mathbb{E}[X_i])^2}$$

Now, we can bound the variance by the expectation of X_i and cancel out one of the sums:

$$\leq \frac{1}{\sum_i \mathbb{E}[X_i]}$$

Further, all X_i are equally distributed, and by the Ground Truth lemma, we get:

$$= \frac{1}{2^{2|w|} \mathbb{E}[X_1]} \lesssim \frac{1}{2^{|w|}}$$

For the covariance, we have a quadratic number of terms, namely, $(2^{2|w|})^2 = 2^{4|w|}$, which is why it is essential that we established a very low bound on them in (2.2):

$$\frac{1}{a^2} \sum_{i \neq j} \text{Cov}(X_i, X_j) \lesssim \frac{1}{2^{4|w|} \mathbb{E}[X_1]^2} 2^{4|w|} \frac{1}{2^{n/2}} \mathbb{E}[X_1]^2 = \frac{1}{2^{n/2}}$$

Note again that this is a very rough approximation to give an intuition, the correct bound can be found in Lemma 5.3. Together, we get a bound showing that the probability of no good collision existing is exponentially small, i.e., with very high probability, for any construction h^F , there exists a collision ColFind can return—and therefore, h cannot be collision resistant (relative to the oracles).

Using the restrictions. Our analysis uses that h^F does not make ColFind queries since we do not know how to prove the lower bound in the Ground Truth Lemma (Lemma 5.1) in this case. Concretely, the proof of Lemma 5.1 (and also the analysis of the variances and co-variances) relies on the *independence* of the sampling of HARD from the sampling of (w, w_{coll}) . However, if h alternates F-queries and ColFind queries, then the answers of ColFind depend on HARD in complex ways, and measuring this interdependence will require new techniques.

Moreover, we use polynomial honesty, since $h^F(w)$ cannot make many queries on an input size of F which is more than polynomially smaller than $|w|$. Such small collisions would make collisions *within* the set $\text{qry}_h^F(w)$ much more likely and thus could increase the plausible number of collisions to much higher than $|w|$. We also use the fixed query-length restriction in the sense that ColFind does not need to try to avoid the HARD set for several input length at the same time. We do not find removing this restriction particularly interesting or important (since the usefulness for a reduction is rather unclear) so no effort was made towards exploring this generalization.

Note that we do not use the 1-ColFind-poly-F restriction here, as this is a restriction on the *reduction*.

3 Preliminaries

Definition 3.1 (Statistical Distance).

$$SD(X, Y) = \frac{1}{2} \sum_{z \in \text{Supp}(X) \cup \text{Supp}(Y)} |\Pr[X = z] - \Pr[Y = z]|$$

Lemma 3.2 (Chernoff Bound). *Let X_1, \dots, X_k be independent random variables with*

$$\forall i : \Pr[X_i = 1] = p, \Pr[X_i = 0] = 1 - p$$

where $0 < p < 1$. Now

$$\Pr \left[\sum_{i=1}^k X_i \leq (1 - \epsilon)kp \right] \leq 2^{-kp\epsilon^2/2}$$

Lemma 3.3 (Borel-Cantelli). *Let E_1, E_2, \dots be a sequence of events on the same probability space. If $\sum_{n \in \mathbb{N}} \Pr[E_n] < \infty$, then the probability that infinitely many of the events occur is 0.*

Corollary 3.4. *Let E_1, E_2, \dots be a sequence of events on the same probability space, and let c be some constant. If there is $N \in \mathbb{N}$ s.t. for all $n > N$ the probability of E_n is less than $\frac{c}{n^2}$, then the probability that infinitely many of the events occur is 0.*

Proof. Firstly, since $\Pr[E_n] \leq \frac{c}{n^2}$ for $n > N$ and $\Pr[E_n] \leq 1$ for all $n \in \mathbb{N}$, we have that

$$\sum_{n \in \mathbb{N}} \Pr[E_n] = \sum_{n \leq N} \Pr[E_n] + \sum_{n > N} \Pr[E_n] \leq N + \sum_{n \in \mathbb{N}} \frac{c}{n^2}.$$

Since N is finite and $\sum_{n \in \mathbb{N}} \frac{c}{n^2} < \infty$, the precondition of the Borel-Cantelli Lemma 3.3 is satisfied and Corollary 3.4 follows.

Lemma 3.5 (Hoeffding's Inequality). *Let X_i , where $i \in \{1, \dots, N\}$, be independent random variables that get the value 1 with probability $1/p$ and 0 else. Then for all $k \geq 0$, we have*

$$\Pr \left[\left| \frac{\sum_{i=1}^N X_i - \mathbb{E} \left[\sum_{i=1}^N X_i \right]}{N} \right| \geq k \right] \leq 2e^{-2Nk^2}$$

Lemma 3.6 (Splitting Lemma). *Let $A \subset X \times Y$ such that $\Pr_{X \times Y}[(x, y) \in A] \geq \varepsilon$. For any $\varepsilon' < \varepsilon$, define B as*

$$B = \{(x, y) \in X \times Y \mid \Pr_{y' \in Y}[(x, y') \in A] \geq \varepsilon - \varepsilon'\}$$

Then, we have $\Pr_{X \times Y}[(x, y) \in B] \geq \varepsilon'$.

Notation. We denote by $\text{qry}_h^F(w)$ the ordered set of F-query-answer pairs that h^F makes on input w . E.g. let $h^f(\cdot) := f(\cdot) \text{ xor } f(\cdot + 1)$ and let $F(\cdot) := 2 \cdot$, now $\text{qry}_h^F(3) = [(3, 6), (4, 8)]$.

4 Main Theorem (dCRH $\not\equiv$ CRH)

In this section, we state our main result and its core lemmata. Our result holds for a restricted class of reductions (see also Section 1.4 for a discussion) which we call 1-ColFind-poly-F and define as follows:

Definition 4.1 (1-ColFind-poly-F Reductions). We restrict our reductions' access to oracles F and ColFind as follows:

1st stage. \mathcal{A} may ask *one* query to ColFind(\cdot), but no queries to F.

2nd stage. \mathcal{A} has (adaptive) access to F, but no queries to ColFind.

We note that we do not restrict the information that is passed from the first to the second stage in any way.

Our main theorem implies that (salted or unsalted) dCRH cannot be used to build a (salted) CRH⁴ in a black-box way using a 1-ColFind-poly-F reduction. We only state our theorem for *unsalted* dCRH, since the existence of an unsalted dCRH trivially implies existence of a salted dCRH.

Theorem 4.2 (Main). *There exist oracles ColFind, F s.t. for all polynomially bounded 1-ColFind-poly-F $\mathcal{A}^{F, \text{ColFind}}$, for all polynomially honest, fixed query-length $h_{\text{slt}}^F : \{0, 1\}^\lambda \rightarrow \{0, 1\}^t$ both of the following hold for all sufficiently large $n \in \mathbb{N}$:*

(\exists dCRH) *F is an unsalted $\frac{1}{4}$ -dCRH, i.e. $\text{SD}(\mathcal{A}^{F, \text{ColFind}}(1^n), \text{COL}_{F, n}) \geq \frac{1}{4}$*

(no CRH) *There is no salted CRH, i.e.,*

$$\Pr_{\text{slt} \leftarrow \{0, 1\}^n} \left[\begin{array}{l} \text{ColFind}(h_{\text{slt}}) \rightarrow (w, w') : \\ w \neq w', h_{\text{slt}}(w) = h_{\text{slt}}(w') \end{array} \right] \geq \frac{1}{\lambda^2}$$

⁴and hence also no *unsalted* CRH, since the existence of an unsalted CRH directly implies existence of a salted CRH.

\mathcal{D} <hr/> for $n \in \mathbb{N} : F'_n \leftarrow \text{\$} \text{Permutations}(\{0, 1\}^n)$ for $x \in \{0, 1\}^n : \quad // \text{ remove last bit}$ $F_n(x) := F'_n(x)_{1, \dots, x -1}$ $F \leftarrow \bigcup F_n \quad // \text{ 2-to-1 function } \forall n \in \mathbb{N}$ $\Pi \leftarrow \text{\$} (\text{Permutations}(\{0, 1\}^{2^m}))_{m \in \mathbb{N}}$ for $y \in \{0, 1\}^* : b_y \leftarrow \text{\$} \{0, 1\}$ HARD $\leftarrow \{y : b_y = 1\}$ return $\left(\begin{array}{l} F, \Pi, \text{HARD}, \\ \text{ColFind}[F, \text{HARD}, \Pi] \end{array} \right)$	$\text{ColFind}[F, \text{HARD}, \Pi](h)$ <hr/> $\lambda \leftarrow \text{input length of } h$ for $i \in [1 \dots 2^{2\lambda}] : \quad // \text{ Go over all pairs}$ $w, w' \leftarrow \Pi_\lambda(i) \quad // \text{ in random order.}$ if $h^F(w) = h^F(w')$ and $\forall x, x' \in (\text{qry}_h^F(w) \cup \text{qry}_h^F(w'))$ $\cap \text{HARD} :$ $F(x) = F(x') \implies x = x'$ then return w, w' return \perp
<p>(A) Oracle Distributions for Lemma 4.4 and Lemma 4.3.</p>	<p>(B) ColFind oracle, h is interpreted as a circuit description.</p>

FIGURE 4.5 — Oracle (Distribution). For conciseness of notation, we use **HARD** for the hard output values as well as for the input values $F^{-1}(\text{HARD})$.

Note that the existential result for dCRH in this oracle world is quite strong: A $\frac{1}{4}$ -dCRH is also a $\frac{1}{p}$ -dCRH for any $p \geq 4$. Recall we do not allow for recursive calls to ColFind. Theorem 4.2 immediately yields our main conceptual result which we re-state here for convenience.

Theorem 1.5 (dCRH $\not\Rightarrow$ CRH). *Let $p \geq 4$ be a polynomial. There is no fully black-box, polynomially honest, fixed query-length construction of (salted) CRH from (unsalted) $\frac{1}{p}$ -dCRH with a black-box reduction which makes a single query to the CRH adversary, followed by a polynomial number of queries to the dCRH.*

Oracle Distribution. A first step in the proof of Theorem 4.2 is to work with an *oracle distribution* and then extract a single oracle from the oracle distribution via a standard Borel-Cantelli argument ([Corollary 3.4](#)) which we defer to [Section A](#). The oracle distribution we consider (cf. [Figure 4.5](#)) samples F as random 2-regular functions for any input size n . In addition, it samples a hard subset **HARD** of size $2^n/2$ (in expectation) for any input size n and an auxiliary sequence of permutations Π which are used in the breaker oracle ColFind. Concretely, we use Π to define a random order in which ColFind iterates over candidate collisions while **HARD** defines the subset where the dCRH-adversary should be unable to produce collisions. To that end ColFind will never return a collision w, w' where queries of $h(w)$ and $h(w')$ within **HARD** produce a collision on F . While **HARD** is defined on the output set, we occasionally also use **HARD** to refer to their pre-images under F or the set of *pairs* of all input-output values in F , where $x \in \text{HARD}$.

Proof Outline. The core of Theorem 4.2 is to show [Lemma 4.3](#) (F is a dCRH) and [Lemma 4.4](#) (no CRH) relative to our oracle distribution over $(F, \text{ColFind})$. For [Lemma 4.3](#) (F is a dCRH), we rely on a compression argument (cf. [Section 6](#)) to show that an adversary \mathcal{A} can not break distributional collision resistance.

Lemma 4.3 (\exists dCRH). *For all 1-ColFind-poly- F PPT \mathcal{A} : $\exists N \in \mathbb{N} \forall n > N :$*

$$\Pr_{(F, \Pi, \text{HARD}) \leftarrow \mathcal{D}} [\text{SD}(\mathcal{A}^{F, \text{ColFind}}(1^n), \text{COL}_{F, n}) \geq 1/4] \geq 1 - \frac{1}{n^2},$$

For [Lemma 4.4](#) (no CRH), the challenge is to prove that ColFind can return some collision for any function h^F without inducing collisions in HARD. We denote by $\text{FColl}_h^F(w, w_{\text{coll}})$ the set of y such that $\text{qry}_h^F(w) \cup \text{qry}_h^F(w_{\text{coll}})$ contains two distinct $x \neq x'$ such that $F(x) = F(x') = y$.

Lemma 4.4 (No CRH). *For all non-uniform, polynomial-time, polynomially honest and fixed query-length functions $h^F : \{0, 1\}^* \rightarrow \{0, 1\}^{* - 1}$ and all large enough λ , ColFind returns a collision with high probability, i.e.*

$$\Pr_{F, \text{HARD} \leftarrow \mathcal{D}} \left[\begin{array}{l} \exists w \neq w' \in \{0, 1\}^\lambda : \\ h^F(w) = h^F(w'), \\ \text{FColl}_h^F(w, w_{\text{coll}}) \cap \text{HARD} = \emptyset \end{array} \right] \geq 1 - \frac{1}{\lambda^2}$$

Note that we prove [Lemma 4.4](#) (no CRH) with respect to *non-uniform*, but *unsalted* hash-functions. In [Appendix A](#), we will then use that [Lemma 4.4](#) implies that ColFind breaks every hash-function with *every* sequence of hard-coded salts (the non-uniformity is needed to discuss hard-coded salts) and thus, especially, also breaks every hash-function with a *random* salt. See [Section A](#) for details. We prove [Lemma 4.4](#) in [Section 5](#).

5 No CRH relative to F , ColFind

In this section, we prove [Lemma 4.4](#), see top of [Fig. 2.4](#) for a proof overview. In order to show that no CRH exists relative to oracles F and ColFind, we need to show that for every candidate hash-function h , ColFind can indeed return an h -collision (w, w_{coll}) without inducing an F -collision on HARD values (with probability $1 - \frac{1}{\lambda^2}$ over the choice of F and ColFind, where λ denotes the input length of h). We occasionally refer to such a collision (w, w_{coll}) as a *legal* collision. Let $h^F : \{0, 1\}^* \rightarrow \{0, 1\}^{* - 1}$ be a polynomially honest, fixed query-length function which, given an input w , makes F -queries in $\{0, 1\}^n$, where n is a polynomial in $\lambda = |w|$. Let $\lambda \in \mathbb{N}$ be an arbitrary, large enough λ . In order to prove that ColFind can choose a legal pair (w, w_{coll}) , given any *fixed* F , we define random variables $X_1^F, \dots, X_{2^{2\lambda}}^F$ such that the event $\sum_{i=1}^{2^{2\lambda}} X_i^F \geq 1$ implies the existence of $w \neq w_{\text{coll}} \in \{0, 1\}^\lambda$ such that $h^F(w) = h^F(w_{\text{coll}})$ and $\text{FColl}_h^F(w, w_{\text{coll}}) \cap \text{HARD} = \emptyset$ (i.e., the existence of a collision ColFind would return).

Let SelfColl^F be the set of $w \in \{0, 1\}^\lambda$ such that $\text{qry}_h^F(w)$ contains F -collisions, and let NoSelfColl^F be the set of $w \in \{0, 1\}^\lambda$ such that $\text{qry}_h^F(w)$ does not contain F -collisions. Since for *typical* F , the set NoSelfColl^F contains the vast majority of w (cf. [Lemma 5.7](#)) and since collisions within $\text{qry}_h^F(w)$ are inconvenient to analyse, we simply discard all w in SelfColl^F (which makes our statement stronger). Moreover, we add a form of “padding” to the random variables X_i^F : Denote by b_y the bit which indicates whether $y \in \text{HARD}^{\text{out}}$ and for each $1 \leq i \leq 2^{2\lambda}$ and each $1 \leq j \leq |w|$, flip an independent bit $b_{i,j}$. With this in mind, we define X_i^F as follows:

$$X_i^F$$

if NoSelfColl^F = \emptyset **then return 0.**
 $w \leftarrow \text{NoSelfColl}^F$; $w_{\text{coll}} \leftarrow \text{NoSelfColl}^F \cap (h_{\text{st}}^F)^{-1}(h_{\text{st}}^F(w))$
if $w \neq w_{\text{coll}}$ **and** $|\text{FColl}_h^F(w, w_{\text{coll}})| \leq \lambda$
 and $\forall y \in \text{FColl}_h^F(w, w_{\text{coll}}) : b_y = 0$
 and $\forall 1 \leq j \leq \lambda - |\text{FColl}_h^F(w, w_{\text{coll}})| : b_{i,j} = 0$
 then return 1
return 0

FIGURE 5.6 — The random variable X_i samples an h-collision Simon-style, while excluding all w which have internal F-collisions. Now, the random variable returns 1, if a set of conditions are fulfilled: The collision is no pseudo-collision, the number of F-collisions between w and w_{coll} is less than λ , all the collisions are contained in the easy set, and all the padding bits are 0.

While the “padding bits” $b_{i,j}$ make us discard pairs (w, w_{coll}) which would form a valid collision, the use of the $b_{i,j}$ ensures that once w and w_{coll} are fixed and distinct and there are not too many F-collisions, then the probability that $X_i^F = 1$ over the choice of the b_y and the $b_{i,j}$ is always $2^{-\lambda}$. This property makes X_i^F independent on the number of F-collisions between w and w_{coll} , which will come in handy in Lemma 5.5. Now, we indeed obtain the desired property that

$$\Pr_{F, \text{HARD} \leftarrow \mathcal{D}, \text{st}} \left[\sum_{i=1}^{2^{2\lambda}} X_i^F \geq 1 \right] \geq 1 - \frac{1}{\lambda^2} \quad (5.3)$$

directly implies Lemma 4.4. Now, what do we know about the random variables X_i^F ? One of our core lemmas establishes a lower bound on the *expectation* of the X_i^F (note that for all X_i^F , the expectation is identical).

Lemma 5.1 (Ground Truth).

$$\Pr_F \left[\Pr_{\text{HARD} \leftarrow \mathcal{D}, b_{i,j} \leftarrow \{0,1\}} [X_i^F = 1] \geq \frac{2^{-\lambda}}{4} \right] \geq 1 - \frac{1}{4\lambda^2}$$

We call the set of all F for which the inner probability holds \mathcal{F}_{GT} . Along the way, we also prove a bound on the probability of the number of F-collisions being above our bound, which is useful on its own:

Lemma 5.2. *Let (w, w_{coll}) be sampled by X_i as described in Figure 5.6 and q the maximal number of queries made by h . Then, the probability that w and w_{coll} have many F-collisions is small:*

$$\Pr_{F, \text{HARD}, (w, w_{\text{coll}}) \leftarrow X_i} \left[|\text{FColl}_h^F(w, w_{\text{coll}})| > \lambda \right] \leq \left(\frac{2q^2}{2^n - q} \right)^\lambda$$

Due to space restrictions, we have moved the proofs of Lemma 5.1 and Lemma 5.2 to Appendix 5.4. If the X_i^F were all *independent*, then a simple Chernoff bound would suffice to show that Lemma 5.1 implies Inequality (5.3). In fact, even *pairwise independence* would suffice using a Chebychev bound, since we have so many copies of the X_i^F . Unfortunately, the X_i^F are not necessarily pairwise independent. If, for example, all h^F collisions (w, w_{coll}) induce on their query set an f -collision on the same y (which is unlikely, but possible with non-zero probability), then the bit b_y can make all

X_i^F to be equal to 0 simultaneously. However, the aforementioned example is somewhat unlikely, i.e., for typical F, typical pairs of collisions (w, w_{coll}) and (w', w'_{coll}) will induce *disjoint* F-collisions. Nevertheless, a small interdependence between the X_i^F remains even for *typical* F. However, we can use a *generalized* Chebychev bound which allows us to handle the case that the X_i^F are *almost* pairwise independent, but not fully.

5.1 Using Chebychev

Chebychev's inequality gives us a way to bound the distance of a variable from its expectation: For a random variable Z and $a > 0$,

$$\Pr[|Z - \mathbb{E}[Z]| \geq a] \leq \frac{\text{Var}(Z)}{a^2}.$$

The helpful property here, especially compared to a Markov bound, is that the bound shrinks quadratically in the distance a . This inequality is often used to bound the sum of *pairwise independent* random variables Z_i :

$$\Pr\left[\left|\sum Z_i - \sum \mathbb{E}[Z_i]\right| \geq a\right] \leq \frac{\text{Var}(\sum Z_i)}{a^2} \stackrel{\text{p.i.}}{=} \frac{\sum \text{Var}(Z_i)}{a^2}, \quad (5.4)$$

where the last step (p.i.) only holds when the Z_i are pairwise independent, as for two independent random variables A and B , $\text{Var}(A + B) = \text{Var}(A) + \text{Var}(B)$. However, as discussed above, our random variables X_i are only *almost* pairwise independent. In this case, instead of the last step (p.i.), we can use Bienaymé's Identity to compute the variance over the sum of X_i s as

$$\text{Var}\left(\sum_{i=1}^{2^{2\lambda}} X_i\right) = \sum_{i=1}^{2^{2\lambda}} \text{Var}(X_i) + \sum_{i \neq j} \text{Cov}(X_i, X_j), \quad (5.5)$$

where the covariance $\text{Cov}(X_i, X_j)$ is a measure of *dependency* between X_i and X_j and can be bounded well when X_i and X_j are quite independent, see Section 5.2 for details, including a definition of covariance. We can now upper-bound the probability that all $X_i^F = 0$. For this, we will now fix a "good" F, i.e., one for which the inner equality of Lemma 5.1 holds, and any salt slt for h and then bound the probability that all of the X_i are 0:

$$\begin{aligned} \Pr\left[\sum_{i=1}^{2^{2\lambda}} X_i^F = 0\right] &\leq \Pr\left[\left|\sum_{i=1}^{2^{2\lambda}} X_i^F - \mathbb{E}[X_i^F]\right| \geq 2^{2\lambda} \mathbb{E}[X_1^F]\right] \\ &\leq \frac{1}{2^{4\lambda} \mathbb{E}[X_1^F]^2} \left(\sum_i \text{Var}(X_i^F) + \sum_{i \neq j} \text{Cov}(X_i^F, X_j^F) \right) \\ &\leq \frac{1}{2^{2\lambda} \mathbb{E}[X_1^F]} + \frac{\left(\sum_{i \neq j} \text{Cov}(X_i^F, X_j^F)\right)}{2^{4\lambda} \mathbb{E}[X_1^F]^2}, \end{aligned}$$

where the first inequality follows from $\sum_{i=1}^{2^{2\lambda}} \mathbb{E}[X_i^F] = 2^{2\lambda} \mathbb{E}[X_1^F]$, the 2nd inequality is an application of Chebychev (cf. 5.4) where the last step is replaced by Bienaymé (cf. 5.5). Finally, the 3rd inequality follows since all X_i^F are identically distributed Bernoulli variables and thus, $\text{Var}(X_i^F) = \text{Var}(X_1^F)$ and

$$\text{Var}(X_i^F) = \mathbb{E}[X_1^F](1 - \mathbb{E}[X_1^F]) \leq \mathbb{E}[X_1^F],$$

so that

$$\frac{\sum_i \text{Var}(X_i^F)}{2^{4\lambda} \mathbb{E}[X_1^F]^2} \leq \frac{2^{2\lambda} \mathbb{E}[X_1^F]}{2^{4\lambda} \mathbb{E}[X_1^F]^2} \leq \frac{1}{2^{2\lambda} \mathbb{E}[X_1^F]}.$$

In Section 5.2, we then prove the following statement.

Lemma 5.3. *If we sample F from \mathcal{F}_{GT} , i.e., the set of F that fulfill the inner probability of Lemma 5.1, then, with high probability, we can bound the covariance:*

$$\Pr_{F \leftarrow \mathcal{F}_{\text{GT}}} \left[\frac{\left(\sum_{i \neq j} \text{Cov}(X_i^F, X_j^F) \right)}{2^{4\lambda} \mathbb{E}[X_1^F]^2} \leq 12\lambda^2 c \sqrt{\frac{q^3}{2^{n-1}}} + 4 \left(\frac{2q^2}{2^n - q} \right)^\lambda \right] \geq 1 - \frac{1}{4\lambda^2}$$

We define the set $\mathcal{F}_{\text{good}}$ as the F fulfilling the inner inequality.

Using Lemma 5.3 for a large fraction of F , we obtain:

$$\begin{aligned} \Pr \left[\sum_{i=1}^{2^{2\lambda}} X_i^F = 0 \right] &\leq \frac{1}{2^{2\lambda} \mathbb{E}[X_1^F]} + \frac{1}{2^{4\lambda} \mathbb{E}[X_1^F]^2} \left(\sum_{i \neq j} \text{Cov}(X_i^F, X_j^F) \right) \\ &\leq \frac{1}{2^{2\lambda} \mathbb{E}[X_1^F]} + 12\lambda^2 c \sqrt{\frac{q^3}{2^{n-1}}} + 4 \left(\frac{2q^2}{2^n - q} \right)^\lambda \end{aligned} \quad (5.6)$$

This allows us to prove the main lemma of this section which we re-state here for convenience.

Lemma 4.4 (No CRH). *For all non-uniform, polynomial-time, polynomially honest and fixed query-length functions $h^F : \{0, 1\}^* \rightarrow \{0, 1\}^{* - 1}$ and all large enough λ , ColFind returns a collision with high probability, i.e.*

$$\Pr_{F, \text{HARD} \leftarrow \mathcal{D}} \left[\begin{array}{l} \exists w \neq w' \in \{0, 1\}^\lambda : \\ h^F(w) = h^F(w'), \\ \text{FColl}_h^F(w, w_{\text{coll}}) \cap \text{HARD} = \emptyset \end{array} \right] \geq 1 - \frac{1}{\lambda^2}$$

Proof (Proof of Lemma 4.4). We first fix an F from $\mathcal{F}_{\text{good}}$ as defined in Lemma 5.3. Now, we bound the event of finding a legal collision (for this fixed F) using X_i^F s:

$$\begin{aligned} &\Pr_{\text{HARD}} \left[\begin{array}{l} \exists w \neq w' \in \{0, 1\}^\lambda : \\ h^F(w) = h^F(w'), \\ \text{FColl}_h^F(w, w_{\text{coll}}) \cap \text{HARD} = \emptyset \end{array} \right] \\ &\geq \Pr_{\text{HARD}} \left[\sum_{i=1}^{2^{2\lambda}} X_i^F \geq 1 \right] = 1 - \Pr_{\text{HARD}} \left[\sum_{i=1}^{2^{2\lambda}} X_i^F = 0 \right] \end{aligned}$$

The inequality holds because X_i^F will only be 1 if we have found a legal collision, and this collision will be included if we sum over all possible collisions. Next, we can use the Chebychev bound (Eq 5.6):

$$\Pr_{\text{HARD}} \left[\sum_{i=1}^{2^{2\lambda}} X_i^F = 0 \right] \leq \frac{1}{2^{2\lambda} \mathbb{E}[X_1^F]} + 12\lambda^2 c \sqrt{\frac{q^3}{2^{n-1}}} + 4 \left(\frac{2q^2}{2^n - q} \right)^\lambda$$

Using Lemma 5.1 (and the fact that $\mathcal{F}_{\text{GT}} \subset \mathcal{F}_{\text{good}}$), we can bound $\mathbb{E}[X_1^F] \geq \frac{1}{4} 2^{-\lambda}$:

$$\leq \frac{4}{2^\lambda} + 12\lambda^2 c \sqrt{\frac{q^3}{2^{n-1}}} + 4 \left(\frac{2q^2}{2^n - q} \right)^\lambda$$

$$\leq 33\lambda^2 \frac{3q^3}{2^{n/2}}$$

Now, this holds for a fixed F from $\mathcal{F}_{\text{good}}$. The set \mathcal{F}_{GT} contains at least a $(1 - \frac{1}{4\lambda^2})$ fraction of all F , and $\mathcal{F}_{\text{good}}$ contains at least a $(1 - \frac{1}{4\lambda^2})$ fraction of all \mathcal{F}_{GT} . The probability of choosing a good F is therefore

$$\Pr_F[F \in \mathcal{F}_{\text{good}}] \geq \left(1 - \frac{1}{4\lambda^2}\right)^2 \geq 1 - \frac{1}{2\lambda^2}$$

Therefore, we can bound the probability for a random F by

$$\Pr_{F, \text{HARD} \leftarrow \text{sD}, \text{st}} \left[\sum_{i=1}^{2^{2\lambda}} X_i^F = 0 \right] \leq \left(1 - \frac{1}{2\lambda^2}\right) 33\lambda^2 \frac{3q^3}{2^{n/2}} + \frac{1}{2\lambda^2} \leq \frac{1}{\lambda^2}$$

5.2 Bounding Co-Variance (Lemma 5.3)

We start by the definition and properties of co-variance.

Covariance. The covariance of two random variables A, B denotes the dependence between the two random variables, and is defined as

$$\text{Cov}(A, B) = \mathbb{E}[(A - \mathbb{E}[A])(B - \mathbb{E}[B])] \quad (5.7)$$

$$= \mathbb{E}[AB] - \mathbb{E}[A]\mathbb{E}[B]. \quad (5.8)$$

Note that if A and B are independent, the covariance is 0, and if $A = B$, the covariance equals the variance of A . We will further make use of the **law of total (co)-variance**:

$$\text{Var}(A) = \mathbb{E}[\text{Var}(A | C)] + \text{Var}(\mathbb{E}[A | C]) \quad (5.9)$$

$$\text{Cov}(A, B) = \mathbb{E}[\text{Cov}(A, B | C)] + \text{Cov}(\mathbb{E}[A | C], \mathbb{E}[B | C]) \quad (5.10)$$

We will further make use of the following inequality:

$$\text{Cov}(A, B) \leq \mathbb{E}[AB] \quad (5.11)$$

Proof of Lemma 5.3. Let us fix some $F \in \mathcal{F}_{\text{good}}$ (note that we haven't defined $\mathcal{F}_{\text{good}}$ yet – we will do this later in Lemma 5.4). For the covariance terms $\text{Cov}(X_i^F, X_j^F)$, we are interested in the intersection of F -collisions of X_i^F and X_j^F : For this, we define two events, the event $\text{Int}(i, j)$ where the intersection is not empty as

$$\text{Int}(i, j) = \text{FColl}(w^i, w_{\text{coll}}^i) \cap \text{FColl}(w^j, w_{\text{coll}}^j) \neq \emptyset.$$

and the size of the intersection as

$$\#\text{Int}(i, j) = \left| \text{FColl}(w^i, w_{\text{coll}}^i) \cap \text{FColl}(w^j, w_{\text{coll}}^j) \right|.$$

Further, we denote by $\#\text{INT}(i, j)$ the random variable over the size of the intersection. We now want to split the covariance based on the number of intersections. The idea here is that, with high probability, the intersection should be empty. However, if it is empty, we can show that the two random variables X_i^F, X_j^F are independent. When the intersection grows, the dependence of X_i^F and

X_j^F grows as well and therefore also their covariance — but larger intersections are very unlikely to happen.

We will now split the covariance accordingly. By the law of total covariance (see Eq. 5.10), we then also have to add a term which considers the covariance over the expectation of X_i^F, X_j^F conditioned on the random variable $\#\text{INT}(i, j)$. Let q be the maximal number of queries by h , then

$$\begin{aligned} \text{Cov}(X_i^F, X_j^F) &= \Pr[\#\text{Int}(i, j) = 0] \cdot \text{Cov}(X_i^F, X_j^F \mid \#\text{Int}(i, j) = 0) \\ &\quad + \sum_{\alpha=1}^q \Pr[\#\text{Int}(i, j) = \alpha] \cdot \text{Cov}(X_i^F, X_j^F \mid \#\text{Int}(i, j) = \alpha) \\ &\quad + \text{Cov}(\mathbb{E}[X_i^F \mid \#\text{INT}(i, j)], \mathbb{E}[X_j^F \mid \#\text{INT}(i, j)]). \end{aligned}$$

We will now bound these three terms separately.

No Intersection

We will first look into the case where there is no intersection between the F-collisions. For this, we are interested in whether the probability of X_i^F is influenced by X_j^F , i.e.,

$$\begin{aligned} &\Pr[X_i^F X_j^F = 1 \mid \#\text{Int}(i, j) = 0] \\ &= \Pr[X_i^F = 1 \mid \#\text{Int}(i, j) = 0 \wedge X_j^F = 1] \Pr[X_j^F = 1 \mid \#\text{Int}(i, j) = 0] \end{aligned}$$

Let FColl_i^F denote the set of F-collisions for X_i^F . We now focus on the probability of $X_i^F = 1$ conditioned on the empty intersection and X_j^F :

$$\begin{aligned} &\Pr[X_i^F = 1 \mid \#\text{Int}(i, j) = 0 \wedge X_j^F = 1] \\ &= \Pr \left[\begin{array}{l} |\text{FColl}_i^F| \leq |w| \\ \forall y \in \text{FColl}_i^F : b_y \\ \forall 1 \leq j \leq |w| - |\text{FColl}_i^F| : b_{i,j} \end{array} \middle| \#\text{Int}(i, j) = 0 \wedge X_j^F = 1 \right] \end{aligned}$$

We first note that the number of F-collisions of X_i^F and X_j^F are independent, as F and h are fixed. Further, as the intersection of F-collisions is empty, all b_y are independent from X_j^F . Therefore, X_i^F is independent from X_j^F given that the intersection is empty:

$$= \Pr[X_i^F = 1 \mid \#\text{Int}(i, j) = 0]$$

As X_i^F and X_j^F are independent in the case that the intersection of F-collisions is empty, their covariance is 0 in this case:

$$\begin{aligned} \text{Cov}(X_i^F, X_j^F) &= \Pr[\#\text{Int}(i, j) = 0] \cdot 0 \\ &\quad + \sum_{\alpha=1}^q \Pr[\#\text{Int}(i, j) = \alpha] \cdot \text{Cov}(X_i^F, X_j^F \mid \#\text{Int}(i, j) = \alpha) \\ &\quad + \text{Cov}(\mathbb{E}[X_i^F \mid \#\text{INT}(i, j)], \mathbb{E}[X_j^F \mid \#\text{INT}(i, j)]). \end{aligned}$$

Intersections

Next, we have to bound the case where we have a non-empty intersection between the F-collisions of X_i^F and X_j^F :

$$\sum_{\alpha=1}^q \Pr[\#\text{Int}(i, j) = \alpha] \cdot \text{Cov}(X_i^F, X_j^F \mid \#\text{Int}(i, j) = \alpha)$$

First, we need that for most “good” F, the probability of a large intersection between the F-collisions is small:

Lemma 5.4 (Intersections of F-Collisions). *Let \mathcal{F}_{GT} be the F for which the inner equality in Lemma 5.1 holds. Further, let (w^i, w_{coll}^i) denote the collision sampled by X_i . Then for most F, the probability for a large intersection is small*

$$\Pr_{F \leftarrow \mathcal{F}_{\text{GT}}} \left[\Pr_{X_i^F, X_j^F} \left[|\text{FColl}(w^i, w_{\text{coll}}^i) \cap \text{FColl}(w^j, w_{\text{coll}}^j)| \geq \alpha \right] \leq \frac{1}{4} \lambda^2 \left(\frac{9q^3}{2^{n-1}} \right)^{\alpha/2} \right] \\ \geq 1 - \frac{1}{4\lambda^2}$$

We will prove this lemma in Appendix 5.3. We will now define the Fs for which the inner inequality of Lemma 5.4 holds as $\mathcal{F}_{\text{good}}$ – therefore, for our fixed F, the inequality holds as well.

Now, let us first concentrate on the covariance term. We can upper bound the covariance by the expected value of the product of X_i^F and X_j^F :

$$\text{Cov}(X_i^F, X_j^F \mid \#\text{Int}(i, j) = \alpha) \\ \leq \mathbb{E}[X_i^F X_j^F \mid \#\text{Int}(i, j) = \alpha]$$

As X_i^F and X_j^F are boolean random variables, their expectation equals the probability of them being one:

$$= \Pr[X_i^F = X_j^F = 1 \mid \#\text{Int}(i, j) = \alpha] \\ = \Pr[X_i^F = 1 \mid X_j^F = 1 \wedge \#\text{Int}(i, j) = \alpha] \Pr[X_j^F = 1 \mid \#\text{Int}(i, j) = \alpha]$$

The first probability is the probability of X_i^F being one, conditioned on X_j^F being one and an intersection of size α between the two random variables, i.e., there are $y_{i_1}, \dots, y_{i_\alpha}$ such that they are in the set of F-collisions for both random variables. Without loss of generality, let us assume these are the lexicographically first y s in the set of F-collisions of X_i^F , and denote them by y_1, \dots, y_α . As $X_j^F = 1$, also all $b_{y_1}, \dots, b_{y_\alpha}$ have to be one, and we can rewrite this probability as

$$\Pr[X_i^F = 1 \mid X_j^F = 1 \wedge \#\text{Int}(i, j) = \alpha] \\ = \Pr[X_i^F = 1 \mid b_{y_1} = \dots = b_{y_\alpha} = 1 \wedge \#\text{Int}(i, j) = \alpha] \\ = \frac{\Pr[X_i^F = 1 \wedge b_{y_1} = \dots = b_{y_\alpha} = 1 \mid \#\text{Int}(i, j) = \alpha]}{\Pr[b_{y_1} = \dots = b_{y_\alpha} = 1]} \\ = \frac{\Pr[X_i^F = 1 \mid \#\text{Int}(i, j) = \alpha]}{2^{-\alpha}} = 2^\alpha \Pr[X_i^F = 1 \mid \#\text{Int}(i, j) = \alpha]$$

Now, X_i^F alone is nearly independent of $\#\text{Int}(i, j)$ — the only dependence stems from the fact that $|\text{FColl}|$ could be larger than $|w|$, in which case X_i^F would be 0. Therefore, we can bound $\mathbb{E}[X_i^F \mid \#\text{Int}(i, j) = \alpha]$ by

$$\Pr[X_i^F = 1 \mid \#\text{Int}(i, j) = \alpha] \leq \Pr[X_i^F = 1 \mid |\text{FColl}| \leq |w|] \leq 2 \Pr[X_1^F = 1]$$

where the last inequality is due to $\Pr[|\text{FColl}| \leq |w|] \geq \frac{1}{2}$ (see Lemma 5.2). Therefore, we get the following bound:

$$\begin{aligned} \text{Cov}(X_i^F, X_j^F \mid \#\text{Int}(i, j) = \alpha) &\leq 2^\alpha 4 \Pr[X_i^F = 1] \Pr[X_j^F = 1] \\ &= 2^\alpha 4 \mathbb{E}[X_i^F]^2 \end{aligned}$$

Lemma 5.4 lets us bound the probability of having an intersection of size α :

$$\Pr[\#\text{Int}(i, j) = \alpha] \leq \frac{1}{4} \lambda^2 \left(\frac{9q^3}{2^{n-1}} \right)^{\alpha/2}$$

Putting both these bounds together, we get

$$\begin{aligned} &\sum_{\alpha=1}^q \Pr[\#\text{Int}(i, j) = \alpha] \cdot \text{Cov}(X_i^F, X_j^F \mid \#\text{Int}(i, j) = \alpha) \\ &\leq \sum_{\alpha=1}^q \frac{1}{4} \lambda^2 \left(\frac{9q^3}{2^{n-1}} \right)^{\alpha/2} \cdot 2^\alpha 4 \mathbb{E}[X_1^F]^2 \end{aligned}$$

and, for sufficiently large n , we can bound the sum by taking its value for $\alpha = 1$ twice:

$$\leq 12\lambda^2 c \sqrt{\frac{q^3}{2^{n-1}}} \mathbb{E}[X_1^F]^2$$

Therefore, our covariance term simplifies to

$$\begin{aligned} \text{Cov}(X_i^F, X_j^F) &= 0 + 12\lambda^2 c \sqrt{\frac{q^3}{2^{n-1}}} \mathbb{E}[X_1^F]^2 \\ &\quad + \text{Cov}(\mathbb{E}[X_i^F \mid \#\text{INT}(i, j)], \mathbb{E}[X_j^F \mid \#\text{INT}(i, j)]). \end{aligned}$$

Covariance of Expectations

Last, we will bound the last term, which computes the covariance of expectations conditioned on the size of the intersection:

$$\text{Cov}(\mathbb{E}[X_i^F \mid \#\text{INT}(i, j)], \mathbb{E}[X_j^F \mid \#\text{INT}(i, j)])$$

Bounding it will again require Lemma 5.4. We will just state the bound here and give a proof in Appendix 5.5.

Lemma 5.5 (Bounding the Covariance of Expectations). *Let $F \in \mathcal{F}_{\text{good}}$. Then,*

$$\text{Cov}(\mathbb{E}[X_i^F \mid \#\text{INT}(i, j)], \mathbb{E}[X_j^F \mid \#\text{INT}(i, j)]) \leq 4 \left(\frac{2q^2}{2^n - q} \right)^\lambda \mathbb{E}[X_i^F]^2$$

This Lemma now allows us to bound the whole covariance term, finishing the proof of Lemma 5.3:

$$\frac{\left(\sum_{i \neq j} \text{Cov}(X_i^F, X_j^F) \right)}{2^{4\lambda} \mathbb{E}[X_1^F]^2} \leq 12\lambda^2 c \sqrt{\frac{q^3}{2^{n-1}}} + 4 \left(\frac{2q^2}{2^n - q} \right)^\lambda$$

□

5.3 Proof of Lemma 5.4 (Intersections of F-Collisions)

Lemma 5.4 (Intersections of F-Collisions). *Let \mathcal{F}_{GT} be the F for which the inner equality in Lemma 5.1 holds. Further, let (w^i, w_{coll}^i) denote the collision sampled by X_i . Then for most F , the probability for a large intersection is small*

$$\Pr_{F \leftarrow \mathcal{F}_{\text{GT}}} \left[\Pr_{X_i^F, X_j^F} \left[\left| \text{FColl}(w^i, w_{\text{coll}}^i) \cap \text{FColl}(w^j, w_{\text{coll}}^j) \right| \geq \alpha \right] \leq \frac{1}{4} \lambda^2 \left(\frac{9q^3}{2^{n-1}} \right)^{\alpha/2} \right] \\ \geq 1 - \frac{1}{4\lambda^2}$$

To prove this lemma, we will first prove a slightly different lemma: This lemma shows, in a distributional way, that for a random F and random collisions (w, w_{coll}) , (w', w'_{coll}) , having a large intersection of non-self F-collisions is small. For this, let $\text{FColl}_{\text{nsc}}(w, w_{\text{coll}})$ define the F-collisions between w and w_{coll} *without* self-collisions: We define $y \in \text{FColl}_{\text{nsc}}$ if there exists an F-collision $x \neq x'$, $F(x) = F(x') = y$ such that $F(x)$, but not $F(x')$, appears in the path of $h^F(w)$ and $F(x')$, but not $F(x)$, appears in the path of $h^F(w_{\text{coll}})$. We write $\bar{y} = (y_1, \dots, y_n) \in \text{FColl}_{\text{nsc}}(w, w_{\text{coll}})$ if every $y_i \in \text{FColl}_{\text{nsc}}(w, w_{\text{coll}})$. Now, Lemma 5.6 states that if w, w_{coll} is a random collision for h^F and w', w'_{coll} is a second random collision for h^F , then the probability that the intersection of F-collisions *without* self-collisions between the two h -collisions is larger than α is bound by $2 \left(\frac{9q^3}{2^{n-3}} \right)^{\alpha/2}$.

Lemma 5.6.

$$\Pr_{F, w, w_{\text{coll}}, w', w'_{\text{coll}}} \left[\left| \text{FColl}_{\text{nsc}}(w, w_{\text{coll}}) \cap \text{FColl}_{\text{nsc}}(w', w'_{\text{coll}}) \right| \geq \alpha \right] \leq 2 \left(\frac{9q^3}{2^{n-1}} \right)^{\alpha/2},$$

where $w \leftarrow_{\$} \{0, 1\}^\lambda$, $w_{\text{coll}} \leftarrow_{\$} (h^F)^{-1}(h^F(w))$, $w' \leftarrow_{\$} \{0, 1\}^\lambda$, and $w'_{\text{coll}} \leftarrow_{\$} (h^F)^{-1}(h^F(w'))$.

Proof. We are now interested in the probability that there exists a set of α pairwise distinct $y_1, \dots, y_\alpha \in \{0, 1\}^{n-1}$ that are in the intersection of the F-collisions. We use \bar{y}_d as a short-hand for y_1, \dots, y_α , where $y_i \neq y_j \forall i \neq j$.

$$\Pr_{F, w, w_{\text{coll}}, w', w'_{\text{coll}}} \left[\left| \text{FColl}_{\text{nsc}}(w, w_{\text{coll}}) \cap \text{FColl}_{\text{nsc}}(w', w'_{\text{coll}}) \right| \geq \alpha \right] \\ \leq \sum_{\bar{y}_d} \Pr_{F, w, w_{\text{coll}}, w', w'_{\text{coll}}} \left[\forall 1 \leq i \leq \alpha : y_i \in \text{FColl}_{\text{nsc}}(w, w_{\text{coll}}) \cap \text{FColl}_{\text{nsc}}(w', w'_{\text{coll}}) \right]$$

We will now look at the probability for one fixed but arbitrary \bar{y}_d

$$\Pr_{F, w, w_{\text{coll}}, w', w'_{\text{coll}}} \left[\forall 1 \leq i \leq \alpha : y_i \in \text{FColl}_{\text{nsc}}(w, w_{\text{coll}}) \cap \text{FColl}_{\text{nsc}}(w', w'_{\text{coll}}) \right]$$

If a y_i appears as collision in $\text{FColl}_{\text{nsc}}(w, w_{\text{coll}})$ as well as in $\text{FColl}_{\text{nsc}}(w', w'_{\text{coll}})$, then it must also appear as collision in either $\text{FColl}_{\text{nsc}}(w, w')$ or in $\text{FColl}_{\text{nsc}}(w, w'_{\text{coll}})$. Further, note that the distribution of (w', w'_{coll}) is symmetric in the sense that it doesn't matter whether one first samples w' and then w'_{coll} conditioned on being a collision with w' , or the other way around. Therefore, without loss of generality (and by losing a factor of two), we can assume that at least half of all y_i are contained in $\text{FColl}_{\text{nsc}}(w, w')$ – we will denote the set of indices of these y_i by I , where $|I| = \lceil \frac{\alpha}{2} \rceil$:

$$\leq 2 \sum_{\substack{I \subseteq \{1, \dots, \alpha\} \\ |I| = \lceil \frac{\alpha}{2} \rceil}} \Pr_{F, w} \left[\begin{array}{l} \forall 1 \leq i \leq \alpha : y_i \in \text{FColl}_{\text{nsc}}(w, w_{\text{coll}}) \wedge \\ \forall j \in I : y_j \in \text{FColl}_{\text{nsc}}(w, w') \end{array} \right],$$

where \mathbf{w} denotes the tuple $(w, w_{\text{coll}}, w', w'_{\text{coll}})$. Now, for every i , we know that exactly one preimage x_i of y_i is queried to F when calculating $h^F(w)$. Further, for every $j \in I$, as y_j is contained in $\text{FColl}_{\text{nsc}}(w, w')$, we know that the second preimage $x'_j \neq x_j$ is queried to F when calculating $h^F(w')$.

$$\leq 2 \sum_{\substack{I \subseteq \{1, \dots, \alpha\} \\ |I| = \lceil \frac{\alpha}{2} \rceil}} \Pr_{F, w, w'} \left[\begin{array}{l} \forall 1 \leq i \leq \alpha \exists! x_i : x_i \in F^{-1}(y_i) \cap \text{qry}_h^F(w) \wedge \\ \forall j \in I \exists x'_j \neq x_j : x'_j \in F^{-1}(y_j) \cap \text{qry}_h^F(w') \end{array} \right]$$

We now upper bound the following probability for any fixed, but arbitrary I , and then later take a union bound over I .

$$\Pr_{F, w, w'} \left[\begin{array}{l} \forall 1 \leq i \leq \alpha \exists! x_i : x_i \in F^{-1}(y_i) \cap \text{qry}_h^F(w) \wedge \\ \forall j \in I \exists x'_j \neq x_j : x'_j \in F^{-1}(y_j) \cap \text{qry}_h^F(w') \end{array} \right]$$

We now want to argue that we can bound the above with the product of the probabilities for individual y_i s. For this, we will first introduce the event $E_{\text{hit}}^I(F, w, w', y, i)$ which denotes above event for a single y_i :

$$E_{\text{hit}}^I(F, w, w', y, i) = \begin{array}{l} \exists! x : x \in F^{-1}(y) \cap \text{qry}_h^F(w) \wedge \\ (i \notin I \vee \exists x' \neq x : x' \in F^{-1}(y) \cap \text{qry}_h^F(w')) \end{array}$$

Now, we can rewrite the probability above using E_{hit}^I :

$$\begin{aligned} & \Pr_{F, w, w'} \left[\begin{array}{l} \forall 1 \leq i \leq \alpha \exists! x_i : x_i \in F^{-1}(y_i) \cap \text{qry}_h^F(w) \wedge \\ \forall j \in I \exists x'_j \neq x_j : x'_j \in F^{-1}(y_j) \cap \text{qry}_h^F(w') \end{array} \right] \\ &= \Pr_{F, w, w'} [\forall 1 \leq i \leq \alpha : E_{\text{hit}}^I(F, w, w', y_i, i)] \\ &= \prod_{1 \leq i \leq \alpha} \Pr_{F, w, w'} [E_{\text{hit}}^I(F, w, w', y_i, i) \mid \forall 1 \leq k < i : E_{\text{hit}}^I(F, w, w', y_k, k)] \end{aligned} \quad (5.11)$$

We will now focus on a fixed but arbitrary i :

$$\Pr_{F, w, w'} [E_{\text{hit}}^I(F, w, w', y_i, i) \mid \forall 1 \leq k < i : E_{\text{hit}}^I(F, w, w', y_k, k)]$$

We want to split the event E_{hit}^I further now to talk about $\text{qry}_h^F(w)$ and $\text{qry}_h^F(w')$ separately, using conditional probabilities:

$$\begin{aligned} &= \Pr_{F, w, w'} [\exists! x : x \in F^{-1}(y) \cap \text{qry}_h^F(w) \mid \forall 1 \leq k < i : E_{\text{hit}}^I(F, w, w', y_k, k)] \\ &\quad \cdot \Pr_{F, w, w'} \left[\left(i \notin I \vee \exists x' \neq x : x' \in F^{-1}(y) \cap \text{qry}_h^F(w') \right) \mid \begin{array}{l} \exists! x : x \in F^{-1}(y) \cap \text{qry}_h^F(w) \wedge \\ \forall 1 \leq k < i : E_{\text{hit}}^I(F, w, w', y_k, k) \end{array} \right] \end{aligned}$$

Let us focus on the first probability. Note that w is chosen uniformly and therefore especially independent of F . Therefore, h does not have any information about F (except what it learns from the queries itself), and for every query $x \in \text{qry}_h^F(w)$, the image $F(x)$ can be seen as randomly chosen from the remaining ys – meaning that the *images* of $\text{qry}_h^F(w)$ are identically distributed to choosing a random subset of size q of all possible ys , and we are looking for the probability that y_i is contained in this set. Now, conditioning on previous hits means that some of the queries map to the previous

y_k s and, as all y_k s are distinct, these queries therefore cannot map to y_i – so conditioning here only decreases the probability. As h makes at most q queries, we can bound the first probability as follows:

$$\Pr_{\mathbb{F}, w, w'} [\exists! x : x \in \mathbb{F}^{-1}(y) \cap \text{qry}_h^{\mathbb{F}}(w) \mid \forall 1 \leq k < i : E_{\text{hit}}^I(\mathbb{F}, w, w', y_k, k)] \leq \frac{q}{2^{(n-1)}}$$

We will now look at the second probability. If i is not included in I , then the probability is 1, let us therefore now assume that $i \in I$. Note again that w' is chosen uniformly and independently of \mathbb{F} and w . We are looking at the probability that some query in $\text{qry}_h^{\mathbb{F}}(w')$ that is not contained in $\text{qry}_h^{\mathbb{F}}(w)$ maps to y_i . Similar to before, we can consider the images of the new queries in $\text{qry}_h^{\mathbb{F}}(w') \setminus \text{qry}_h^{\mathbb{F}}(w)$ as a random subset of all y s, with a slightly smaller probability to choose one of the y s that appeared already as an image of $\text{qry}_h^{\mathbb{F}}(w)$ (which includes y_i). With a similar argument as before, the previous hits only decrease the probability, and therefore, we can give an upper bound to the probability (still assuming $i \in I$):

$$\Pr_{\mathbb{F}, w, w'} \left[\begin{array}{l} i \notin I \vee \exists x' \neq x : \\ x' \in \mathbb{F}^{-1}(y) \cap \text{qry}_h^{\mathbb{F}}(w') \end{array} \mid \begin{array}{l} \exists! x : x \in \mathbb{F}^{-1}(y) \cap \text{qry}_h^{\mathbb{F}}(w) \wedge \\ \forall 1 \leq k < i : E_{\text{hit}}^I(\mathbb{F}, w, w', y_k, k) \end{array} \right] \leq \frac{q}{2^{(n-1)}}$$

Now we can plug this result into Equation (5.11): Note that as I contains at least $\frac{\alpha}{2}$ many i s, we get

$$\begin{aligned} \Pr_{\mathbb{F}, w, w'} & \left[\begin{array}{l} \forall 1 \leq i \leq \alpha \exists x_i : x_i \in \mathbb{F}^{-1}(y_i) \cap \text{qry}_h^{\mathbb{F}}(w) \wedge \\ \forall j \in I \exists x'_j \neq x_j : x'_j \in \mathbb{F}^{-1}(y_j) \cap \text{qry}_h^{\mathbb{F}}(w') \end{array} \right] \\ & \leq \left(\frac{q}{2^{n-1}} \right)^\alpha \left(\frac{q}{2^{n-1}} \right)^{\frac{\alpha}{2}} \end{aligned}$$

Summing over all possible I s:

$$\begin{aligned} \Pr_{\mathbb{F}, w, w_{\text{coll}}, w', w'_{\text{coll}}} & [\forall 1 \leq i \leq \alpha : y_i \in \text{FColl}_{\text{nsc}}(w, w_{\text{coll}}) \cap \text{FColl}_{\text{nsc}}(w', w'_{\text{coll}})] \\ & \leq 2 \sum_{\substack{I \subseteq \{1, \dots, \alpha\} \\ |I| = \lceil \frac{\alpha}{2} \rceil}} \left(\frac{q}{2^{n-1}} \right)^\alpha \left(\frac{q}{2^{n-1}} \right)^{\frac{\alpha}{2}} \end{aligned}$$

Note that there are $\binom{\alpha}{\lceil \frac{\alpha}{2} \rceil} < \left(\frac{\alpha e}{\alpha/2} \right)^{\alpha/2} \leq 3^\alpha$ many combinations for I :

$$\begin{aligned} & \leq 2 \cdot 3^\alpha \cdot \left(\frac{q}{2^{n-1}} \right)^\alpha \left(\frac{q}{2^{n-1}} \right)^{\frac{\alpha}{2}} \\ & \leq 2 \left(\frac{9q^3}{2^{n-1}} \right)^{\frac{\alpha}{2}} 2^{-\alpha(n-1)} \end{aligned}$$

If we now sum over all possible combinations of y_i s, we get

$$\begin{aligned} \Pr_{\mathbb{F}, w, w_{\text{coll}}, w', w'_{\text{coll}}} & [|\text{FColl}_{\text{nsc}}(w, w_{\text{coll}}) \cap \text{FColl}_{\text{nsc}}(w', w'_{\text{coll}})| \geq \alpha] \\ & \leq \sum_{y_1, \dots, y_\alpha} 2 \left(\frac{9q^3}{2^{n-1}} \right)^{\frac{\alpha}{2}} 2^{-\alpha(n-1)} \\ & = 2 \left(\frac{9q^3}{2^{n-1}} \right)^{\frac{\alpha}{2}} \end{aligned}$$

In the next step, we want to adapt this result to the setting of Lemma 5.4: We show that for most “good” F as defined in Lemma 5.1, the result holds if we sample the w s as it is done in X_i , i.e., without self-collisions. Note that in that case FColl equals $\text{FColl}_{\text{nsc}}$.

Proof (Proof of Lemma 5.4). For the proof, we will first use Markov’s inequality to split off the probability of F and restrict it to the “good” F s. Afterwards, we will use the fact that for “good” F , the probability of self-collisions are small to exclude them.

Let b be the expectation over F of the probability in Lemma 5.6:

$$\mathbb{E}_F[\Pr[|\text{FColl}_{\text{nsc}}(w, w_{\text{coll}}) \cap \text{FColl}_{\text{nsc}}(w', w'_{\text{coll}})| \geq \alpha]] = b \leq 2 \left(\frac{9q^3}{2^{n-1}} \right)^{\alpha/2}$$

Markov’s inequality gives us

$$\Pr_F \left[\Pr[|\text{FColl}_{\text{nsc}}(w, w_{\text{coll}}) \cap \text{FColl}_{\text{nsc}}(w', w'_{\text{coll}})| \geq \alpha] \geq \frac{\lambda^2}{8} b \right] \leq \frac{8}{\lambda^2}$$

Conditioning on $F \in \mathcal{F}_{\text{GT}}$:

$$\begin{aligned} & \Pr_{F \leftarrow \mathcal{F}} \left[\Pr[|\text{FColl}_{\text{nsc}}(w, w_{\text{coll}}) \cap \text{FColl}_{\text{nsc}}(w', w'_{\text{coll}})| \geq \alpha] \geq \frac{\lambda^2}{8} b \right] \\ &= \Pr_F \left[\Pr[|\text{FColl}_{\text{nsc}}(w, w_{\text{coll}}) \cap \text{FColl}_{\text{nsc}}(w', w'_{\text{coll}})| \geq \frac{\lambda^2}{8} b \mid F \in \mathcal{F}_{\text{GT}}] \right] \\ &\leq \frac{1}{\Pr[F \in \mathcal{F}_{\text{GT}}]} \Pr_F \left[\Pr[|\text{FColl}_{\text{nsc}}(w, w_{\text{coll}}) \cap \text{FColl}_{\text{nsc}}(w', w'_{\text{coll}})| \geq \alpha] \geq \frac{\lambda^2}{8} b \right] \\ &\leq \frac{1}{4\lambda^2}, \end{aligned}$$

where the last inequality is due to $\Pr[F \in \mathcal{F}_{\text{GT}}] \geq \frac{1}{2}$. We will now take the counter-probability:

$$\Pr_{F \leftarrow \mathcal{F}_{\text{GT}}} \left[\Pr[|\text{FColl}_{\text{nsc}}(w, w_{\text{coll}}) \cap \text{FColl}_{\text{nsc}}(w', w'_{\text{coll}})| \geq \alpha] \leq \frac{\lambda^2}{8} b \right] \geq 1 - \frac{1}{4\lambda^2}$$

Let us fix a “good” F for which the inner event holds. We now want to switch back to sampling from X_i and X_j , i.e., sampling all the w s without self-collisions:

$$\begin{aligned} & \Pr_{X_i, X_j} \left[|\text{FColl}(w^i, w^i_{\text{coll}}) \cap \text{FColl}(w^j, w^j_{\text{coll}})| \geq \alpha \right] \\ &= \Pr[|\text{FColl}_{\text{nsc}}(w, w_{\text{coll}}) \cap \text{FColl}_{\text{nsc}}(w', w'_{\text{coll}})| \geq \alpha \mid \text{NoSelfColl}(w, w_{\text{coll}}, w', w'_{\text{coll}})] \\ &\leq \frac{1}{\Pr[\text{NoSelfColl}(w, w_{\text{coll}}, w', w'_{\text{coll}})]} \Pr[|\text{FColl}_{\text{nsc}}(w, w_{\text{coll}}) \cap \text{FColl}_{\text{nsc}}(w', w'_{\text{coll}})| \geq \alpha] \\ &\leq 2 \frac{\lambda^2}{8} b = \frac{1}{4} \lambda^2 \left(\frac{9q^3}{2^{n-1}} \right)^{\alpha/2} \end{aligned}$$

The last inequality here uses that the probability for self collisions is less than $\frac{1}{2}$ (see Lemma 5.7). Plugging everything together, we get the statement of the lemma.

5.4 Proof of Lemma 5.1 (Ground Truth)

In this section, we prove Lemma 5.1, i.e.,

$$\Pr_F \left[\Pr_{\text{HARD} \leftarrow \mathcal{D}, b_{i,j} \leftarrow \mathbb{S}\{0,1\}} [X_i^F = 1] \geq \frac{2^{-\lambda}}{4}] \geq 1 - \frac{1}{4\lambda^2} \right] \quad (5.12)$$

For the proof of Inequality 5.12, we will consider all the different components of X_i (which we recall shortly). For our analysis, however, it will be useful to define the set $\overline{\text{FColl}}(w, w_{\text{coll}})$ slightly different from $\text{FColl}(w, w_{\text{coll}})$, we highlight the difference between the two definitions in blue:

$$\begin{aligned} \overline{\text{FColl}}(w, w_{\text{coll}}) &:= \{y : \exists x \in \text{qry}_h^F(w), \exists x' \in \text{qry}_h^F(w_{\text{coll}}) \setminus \text{qry}_h^F(w) \wedge x \neq x' \wedge F(x) = F(x') = y\} \\ \text{FColl}(w, w_{\text{coll}}) &:= \{y : \exists x, x' \in \text{qry}_h^F(w) \cup \text{qry}_h^F(w_{\text{coll}}) \wedge x \neq x' \wedge F(x) = F(x') = y\} \end{aligned}$$

Concretely, $\overline{\text{FColl}}(w, w_{\text{coll}})$ demands that x' is *only* in $\text{qry}_h^F(w_{\text{coll}})$ and not also in $\text{qry}_h^F(w)$. For Inequality 5.12, it does not matter whether X_i uses $\overline{\text{FColl}}(w, w_{\text{coll}})$ or $\text{FColl}(w, w_{\text{coll}})$ since in the case where the two sets are different, there is a self-collision on w (since both x and x' are in $\text{qry}_h^F(w)$) and X_i samples w from NoSelfColl_h^F only (cf. Fig. 5.6). Thus, as long as w and w_{coll} are sampled from NoSelfColl_h^F , the sets $\text{FColl}(w, w_{\text{coll}})$ and $\overline{\text{FColl}}(w, w_{\text{coll}})$ are equivalent. Using this adapted definition of $\overline{\text{FColl}}(w, w_{\text{coll}})$, we now recall the definition of X_i (cf. Figure 5.6) and introduce notation. X_i samples w from NoSelfColl_h^F and a collision w_{coll} from $\text{NoSelfColl}_h^F \cap (h^F)^{-1}(h^F(w))$. X_i is 1 if and only if all of the following three events happen:

$$\begin{aligned} E_{\neq} &: w \neq w_{\text{coll}} \\ E_{\text{bB}} &: |\overline{\text{FColl}}(w, w_{\text{coll}})| \leq \lambda \\ E_{\text{easy}} &: \forall y \in \overline{\text{FColl}}(w, w_{\text{coll}}) : b_y = 0 \text{ and} \\ &\quad \forall 1 \leq j \leq \lambda - |\overline{\text{FColl}}(w, w_{\text{coll}})| : b_{i,j} = 0. \end{aligned}$$

Here, E_{\neq} denotes that X_i has sampled a collision (w, w_{coll}) such that $w \neq w_{\text{coll}}$. Further, E_{bB} (*below bound*) denotes that the number of induced F -collisions is below the bound of λ many collisions. Finally, E_{easy} denotes that all the induced F -collisions are in the easy set, as well as that all padding bit flips are 0. Now, note that X_i is 1 if and only if all these three events are true. Therefore, we can now re-write Inequality 5.12 in this notation as follows as follows:

$$\Pr_F \left[\Pr_{\text{HARD} \leftarrow \mathcal{D}, X_i} [E_{\neq} \wedge E_{\text{bB}} \wedge E_{\text{easy}}] \geq \frac{2^\lambda}{4} \right] \geq 1 - \frac{1}{4\lambda^2} \quad (5.13)$$

Let F be arbitrary, but fixed. We have that

$$\begin{aligned} &\Pr_{\text{HARD} \leftarrow \mathcal{D}, X_i} [E_{\neq} \wedge E_{\text{bB}} \wedge E_{\text{easy}}] \\ &= \Pr_{\text{HARD} \leftarrow \mathcal{D}, X_i} [E_{\neq} \wedge E_{\text{easy}} \mid E_{\text{bB}}] \cdot \Pr[E_{\text{bB}}] \end{aligned}$$

Conditioned on E_{bB} , the events E_{\neq} and E_{easy} are independent and, in particular, $\Pr[E_{\text{easy}}] = 2^{-\lambda}$. Thus, the previous term is equal to

$$\begin{aligned} &2^{-\lambda} \cdot \Pr_{\text{HARD} \leftarrow \mathcal{D}, X_i} [E_{\neq} \mid E_{\text{bB}}] \cdot \Pr[E_{\text{bB}}] \\ &= 2^{-\lambda} \cdot \Pr_{\text{HARD} \leftarrow \mathcal{D}, X_i} [E_{\neq} \wedge E_{\text{bB}}] \end{aligned}$$

and Inequality 5.13 simplifies to proving

$$\Pr_F \left[\Pr \left[\begin{array}{c} w \leftarrow \text{NoSelfColl}_h^F \\ w_{\text{coll}} \leftarrow \text{NoSelfColl}_h^F \cap (h^F)^{-1}(h^F(w)) \end{array} \mid E_{\neq} \wedge E_{\text{bB}} \right] \geq \frac{1}{4} \right] \geq 1 - \frac{1}{4\lambda^2} \quad (5.14)$$

In order to prove Inequality 5.14, we want to move to a setting where w and w_{coll} are sampled from the entire space $\{0, 1\}^\lambda$ rather than only from NoSelfColl_h^F . For most F , the set NoSelfColl_h^F covers most of $\{0, 1\}^\lambda$ and thus, this does not affect probabilities significantly. We now first prove that NoSelfColl_h^F covers most of $\{0, 1\}^\lambda$ by a simple birthday bound and application of Markov inequality.

Lemma 5.7 (Large SelfColl_h^F).

$$\Pr_F \left[\left| \text{NoSelfColl}_h^F \right| \leq \left(1 - 8\lambda^2 \frac{q^2}{2^n} \right) 2^\lambda \right] \leq \frac{1}{8\lambda^2}$$

Proof of Lemma 5.7. We first determine an upper bound of the expected size of SelfColl_h^F and then apply Markov inequality.

$$\begin{aligned} \mathbb{E}_F \left[\left| \text{SelfColl}_h^F \right| \right] &= \sum_w \Pr_F \left[w \in \text{SelfColl}_h^F \right] \quad (\text{by linearity of expectation}) \\ &\leq 2^\lambda \frac{q^2}{2^n} \quad (\text{by a birthday bound}) \end{aligned}$$

Markov inequality then implies that $\Pr_F \left[\left| \text{SelfColl}_h^F \right| > 8\lambda^2 2^\lambda \frac{q^2}{2^n} \right] \leq \frac{1}{8\lambda^2}$. \square

We will now next introduce a lemma that does not depend on sampling w and w_{coll} without collisions, and show that this new lemma together with the Markov bound yields Inequality 5.14.

Lemma 5.8.

$$\Pr_F \left[\Pr_{\substack{w \leftarrow \{0, 1\}^\lambda \\ w_{\text{coll}} \leftarrow (h^F)^{-1}(h^F(w))}} \left[E_{\neq} \wedge E_{\text{bB}} \right] \geq \frac{1}{4} + 8\lambda^2 \frac{2q^2}{2^n} \right] \geq 1 - \frac{1}{8\lambda^2}$$

Lemma 5.7 and Lemma 5.8 implies Inequality 5.14. Let F be such that

$$\left| \text{NoSelfColl}_h^F \right| > \left(1 - 8\lambda^2 \frac{q^2}{2^n} \right) 2^\lambda.$$

Let $E_{\text{NoSelfColl}}$ be the event that $w, w_{\text{coll}} \in \text{NoSelfColl}_h^F$ and E_{SelfColl} be the event that one of them is not. Since the distribution of w and w_{coll} is Simon-random, a union bound yields that

$$\Pr[E_{\text{SelfColl}}] \leq 8\lambda^2 \frac{q^2}{2^n} + 8\lambda^2 \frac{q^2}{2^n}.$$

$$\begin{aligned} &\Pr_{\substack{w \leftarrow \text{NoSelfColl}_h^F \\ w_{\text{coll}} \leftarrow \text{NoSelfColl}_h^F \cap (h^F)^{-1}(h^F(w))}} \left[E_{\neq} \wedge E_{\text{bB}} \right] \\ &= \Pr[E_{\neq} \wedge E_{\text{bB}} \mid E_{\text{NoSelfColl}}], \text{ where } w \leftarrow \{0, 1\}^\lambda, w_{\text{coll}} \leftarrow (h^F)^{-1}(h^F(w)) \\ &\geq \Pr[E_{\neq} \wedge E_{\text{bB}} \wedge E_{\text{NoSelfColl}}] \\ &\geq \Pr[E_{\neq} \wedge E_{\text{bB}}] - \Pr[E_{\text{SelfColl}}] \\ &\geq \Pr[E_{\neq} \wedge E_{\text{bB}}] - 8\lambda^2 \frac{2q^2}{2^n} \geq \frac{1}{4} \end{aligned}$$

Since Lemma 5.7 establishes that all except for a $\frac{1}{8\lambda^2}$ fraction of all F satisfy the property that $\left| \text{NoSelfColl}_h^F \right| > \left(1 - 8\lambda^2 \frac{q^2}{2^n}\right) 2^\lambda$, Inequality 5.14 follows. \square

Now, we are just left with proving Lemma 5.8, which we will do next. For this, note that we have

$$\begin{aligned} & \Pr[E_{\neq} \wedge E_{\text{bB}}] \\ & \geq \Pr[E_{\neq}] - \Pr[\neg E_{\text{bB}}] \geq \frac{1}{2} - \Pr[\neg E_{\text{bB}}], \end{aligned} \quad (5.16)$$

where $\Pr[E_{\neq}] \geq \frac{1}{2}$ follows from a counting argument since w is one bit longer than the output of z , and therefore, every w has at least one (true) collision on average. Thus, it remains to upper bound $\Pr[\neg E_{\text{bB}}]$, the event that (w, w_{coll}) induce too many F -collisions, and Lemma 5.8 then follows from Inequality 5.16 and Lemma 5.9 which we state next.

Towards this goal, we split the sampling into the sampling of $F_w := \text{qry}_h^F(w)$ and \overline{F}_w which is F on the rest of the domain.

Lemma 5.9 (Below bound is likely). *For any w and F_w , we have that*

$$\Pr_{\overline{F}_w, w'} [|\overline{FColl}(w, w')| > \lambda] \leq \left(\frac{q^2}{2^n - q} \right)^\lambda$$

Note that we sample a uniform w' here, which is not necessarily a collision with w .

Inequality 5.16 and Lemma 5.9 imply Lemma 5.8 (as well as Lemma 5.2). Since the event $h^F(w) = h^F(w')$ has probability at least $2^{-\lambda}$ for a uniformly random w' , Lemma 5.9 implies that

$$\Pr_{\overline{F}_w, w'} [|\overline{FColl}(w, w')| > \lambda \mid h^F(w) = h^F(w')] \leq \left(\frac{q^2}{2^n - q} \right)^\lambda \cdot 2^\lambda.$$

and since this holds for *every* w and F_w , it holds, in particular, also for random w and F_w and we obtain

$$\Pr_{F, w, w_{\text{coll}}} [\neg E_{\text{bB}}] \leq \left(\frac{q^2}{2^n - q} \right)^\lambda \cdot 2^\lambda, \quad (5.17)$$

This is the *expected* probability over F , which already yields the proof for Lemma 5.2. Using Markov inequality, we obtain that

$$\Pr_F \left[\Pr_{w, w_{\text{coll}}} [\neg E_{\text{bB}}] > 8\lambda^2 \left(\frac{q^2}{2^n - q} \right)^\lambda \cdot 2^\lambda \right] \leq \frac{1}{8\lambda^2}. \quad (5.18)$$

Finally, to prove Lemma 5.8, we have that

$$\begin{aligned} & \Pr_F \left[\Pr_{w, w_{\text{coll}}} [E_{\neq} \wedge E_{\text{bB}}] \geq \frac{1}{4} + 8\lambda^2 \frac{2q^2}{2^n} \right] \\ & \geq \Pr_F \left[\Pr_{w, w_{\text{coll}}} [E_{\neq}] - \Pr_{w, w_{\text{coll}}} [E_{\text{bB}}] \geq \frac{1}{4} + 8\lambda^2 \frac{2q^2}{2^n} \right] \\ & \geq \Pr_F \left[\frac{1}{2} - \Pr_{w, w_{\text{coll}}} [\neg E_{\text{bB}}] \geq \frac{1}{4} + 8\lambda^2 \frac{2q^2}{2^n} \right] \text{ by Inequality 5.16} \\ & \geq \Pr_F \left[\frac{1}{4} - 8\lambda^2 \frac{2q^2}{2^n} \geq \Pr_{w, w_{\text{coll}}} [\neg E_{\text{bB}}] \right] \\ & \geq \Pr_F \left[8\lambda^2 \left(\frac{q^2}{2^n - q} \right)^\lambda > \Pr_{w, w_{\text{coll}}} [\neg E_{\text{bB}}] \right] \text{ by polynomial honesty} \end{aligned}$$

$$\geq 1 - \frac{1}{8\lambda^2} \text{ by Inequality 5.18}$$

□

Proof of Lemma 5.9. For proving Lemma 5.9, we want to bound the probability that for any w , if we choose a second w' uniformly, there are many F-collisions between the two:

$$\Pr_{\overline{F}_w, w'} [|\overline{\text{FColl}}(w, w')| > \lambda] \leq \left(\frac{q^2}{2^n - q} \right)^\lambda$$

By definition of $\overline{\text{FColl}}(w, w')$, we can think of h only making *new* F-queries on w' , i.e., queries not in $\text{qry}_h^F(w)$. Since F is a 2-to-1 function, we know that each of the 2^{n-1} output values should be reached exactly twice and the answers to queries of h^F on input w' are chosen uniformly amongst the still available “slots”, i.e., q out of $2 \cdot 2^{n-1}$ slots have been taken by $\text{qry}_h^F(w)$ and there are $2^n - q$ remaining slots. Since h makes at most q queries on w' , we can assume w.l.o.g. that h makes exactly q queries by adding new queries until reaching q , because the number of F-collisions can only increase by adding queries. Denote by y_1, \dots, y_q the answers to these queries. Denote by Y_w all the y -values reached in $\text{qry}_h^F(w)$ which are at most q .

Now, to bound the probability that $|\overline{\text{FColl}}(w, w')| > \lambda$, there needs to be a size $|\lambda|$ subset $\{y_{i_1}, \dots, y_{i_{|\lambda|}}\} \subseteq \{y_1, \dots, y_q\}$ such that $\{y_{i_1}, \dots, y_{i_{|\lambda|}}\} \subseteq Y_w$. Thus, we take a union bound over all size $|\lambda|$ subsets of $\{y_{i_1}, \dots, y_{i_{|\lambda|}}\}$ and bound the probability that for one of these subsets, all answers to the queries $y_{i_1}, \dots, y_{i_\lambda}$ are in Y_w :

$$\begin{aligned} & \Pr_{\overline{F}_w, w' \leftarrow \mathfrak{s}\{0,1\}^\lambda} [|\overline{\text{FColl}}(w, w')| > \lambda] \\ & \leq \sum_{i_1 < \dots < i_\lambda \leq q} \Pr[y_{i_1}, \dots, y_{i_\lambda} \in Y_w] \\ & = \sum_{i_1 < \dots < i_\lambda \leq q} \Pr[y_{i_1} \in Y_w] \cdot \Pr[y_{i_2} \in Y_w \mid y_{i_1} \in Y_w] \cdots \Pr[y_{i_\lambda} \in Y_w \mid y_{i_1}, \dots, y_{i_{\lambda-1}} \in Y_w] \end{aligned}$$

Now, for the first y_{i_1} , we have $2^n - q$ “slots”, q of which are in Y_w (due to F being 2-regular) – therefore, the probability of y_{i_1} being in Y_w is $q/(2^n - q)$. For the probability, $y_{i_2} \in Y_w$ conditioned on $y_{i_1} \in Y_w$, we have one less slot, but also only $q - 1$ many possibilities to land in Y_w – therefore, the probability is $(q-1)/(2^n - q - 1) \leq q/(2^n - q)$. Continuing this argument, we can bound the probabilities by

$$\begin{aligned} & \leq \sum_{i_1 < \dots < i_\lambda \leq q} \left(\frac{q}{2^n - q} \right)^\lambda = \binom{q}{\lambda} \left(\frac{q}{2^n - q} \right)^\lambda \\ & \leq q^\lambda \left(\frac{q}{2^n - q} \right)^\lambda = \left(\frac{q^2}{2^n - q} \right)^\lambda \end{aligned}$$

□

5.5 Proof of Lemma 5.5

In this section, we want to bound the last term that appeared when we split the covariance based on the size of the intersection:

$$\text{Cov}(X_i^F, X_j^F) = \Pr[\#\text{Int}(i, j) = 0] \cdot \text{Cov}(X_i^F, X_j^F \mid \#\text{Int}(i, j) = 0)$$

$$\begin{aligned}
& + \sum_{\alpha=1}^q \Pr[\#\text{Int}(i, j) = \alpha] \cdot \text{Cov}(X_i^F, X_j^F \mid \#\text{Int}(i, j) = \alpha) \\
& + \text{Cov}(\mathbb{E}[X_i^F \mid \#\text{INT}(i, j)], \mathbb{E}[X_j^F \mid \#\text{INT}(i, j)]).
\end{aligned}$$

The first terms are intuitive — we take the covariances conditioned on a intersection size and multiply it with the probability of such an intersection size. Now, the last term takes into account how the two random variables X_i and X_j relate to each other for a randomly chosen intersection size. The expectation of X_i conditioned on $\#\text{INT}(i, j)$ is now a random variable over the randomness of $\#\text{INT}(i, j)$ and not over the randomness of X_i . With this, let us now restate the lemma and give the proof.

Lemma 5.5 (Bounding the Covariance of Expectations). *Let $F \in \mathcal{F}_{\text{good}}$. Then,*

$$\text{Cov}(\mathbb{E}[X_i^F \mid \#\text{INT}(i, j)], \mathbb{E}[X_j^F \mid \#\text{INT}(i, j)]) \leq 4 \left(\frac{2q^2}{2^n - q} \right)^\lambda \mathbb{E}[X_i^F]^2$$

Proof.

First note that the two conditional expectations are identically distributed, and therefore, we can replace the covariance with a variance:

$$\text{Cov}(\mathbb{E}[X_i^F \mid \#\text{INT}(i, j)], \mathbb{E}[X_j^F \mid \#\text{INT}(i, j)]) = \text{Var}(\mathbb{E}[X_i^F \mid \#\text{INT}(i, j)])$$

We will now start by looking at the expectation conditioned on the random variable $\#\text{INT}(i, j)$. First, note that as long as the number of F -collisions is below $|w|$, X_i^F is independent from $\text{INT}(i, j)$: Due to the padding, the exact number of F -collisions does not influence the random variable. Therefore, we will now split the expectation based on whether the number of F -collisions is above or below the bound. For this, let bB_i (below bound) denote the event that for X_i , $|\text{FColl}_i^F| \leq |w|$. Then,

$$\begin{aligned}
\mathbb{E}[X_i^F \mid \#\text{INT}(i, j)] &= \Pr[\neg \text{bB}_i \mid \#\text{INT}(i, j)] \mathbb{E}[X_i^F \mid \#\text{INT}(i, j) \wedge \neg \text{bB}_i] \\
&+ \Pr[\text{bB}_i \mid \#\text{INT}(i, j)] \mathbb{E}[X_i^F \mid \#\text{INT}(i, j) \wedge \text{bB}_i]
\end{aligned}$$

If bB_i does not happen, X_i is always 0, and therefore the first term is 0

$$= 0 + \Pr[\text{bB}_i \mid \#\text{INT}(i, j)] \mathbb{E}[X_i^F \mid \#\text{INT}(i, j) \wedge \text{bB}_i]$$

Now, however, due to the independence if $|\text{FColl}_i^F| \leq |w|$, the expectation does not depend on $\#\text{INT}$ anymore:

$$= \Pr[\text{bB}_i \mid \#\text{INT}(i, j)] \mathbb{E}[X_i^F \mid \text{bB}_i]$$

We will now apply this to the variance:

$$\text{Var}(\mathbb{E}[X_i^F \mid \#\text{INT}(i, j)]) = \text{Var}(\Pr[\text{bB}_i \mid \#\text{INT}(i, j)] \mathbb{E}[X_i^F \mid \text{bB}_i])$$

Note that the variance is computed over the random variable $\#\text{INT}(i, j)$, but the expected value is now a constant and can be moved out of the variance:

$$= \mathbb{E}[X_i^F \mid \text{bB}_i]^2 \text{Var}(\Pr[\text{bB}_i \mid \#\text{INT}(i, j)])$$

and as the probability of \mathbf{bB}_i is small, we can upper bound $\mathbb{E}[X_i^F \mid \mathbf{bB}_i]^2$ by $2\mathbb{E}[X_i^F]$:

$$\leq 4\mathbb{E}[X_i^F]^2 \text{Var}(\text{Pr}[\mathbf{bB}_i \mid \#\text{INT}(i, j)])$$

Next, we want to split the variance based on whether we are below our collision bound, i.e., whether \mathbf{bB}_i holds for X_i^F . For this, let \mathbf{BB}_i denote the random variable over \mathbf{bB}_i , and we will use the law of total variance (see Eq. 5.9) to get

$$\begin{aligned} \text{Var}(\text{Pr}[\mathbf{bB}_i \mid \#\text{INT}(i, j)]) &= \text{Pr}[\mathbf{bB}_i] \text{Var}(\text{Pr}[\mathbf{bB}_i \mid \#\text{INT}(i, j)] \mid \mathbf{bB}_i) \\ &\quad + \text{Pr}[\neg \mathbf{bB}_i] \text{Var}(\text{Pr}[\mathbf{bB}_i \mid \#\text{INT}(i, j)] \mid \neg \mathbf{bB}_i) \\ &\quad + \text{Var}(\mathbb{E}[\text{Pr}[\mathbf{bB}_i \mid \#\text{INT}(i, j)] \mid \mathbf{BB}]) \end{aligned}$$

Now, clearly, the first two terms are 0, as the probability of \mathbf{bB}_i is constant conditioned on both \mathbf{bB}_i and $\neg \mathbf{bB}_i$. Therefore, only the last term remains, which is now also independent of INT:

$$\begin{aligned} &= 0 + 0 + \text{Var}(\mathbb{E}[\text{Pr}[\mathbf{bB}_i] \mid \mathbf{BB}]) \\ &= \text{Var}(\text{Pr}[\mathbf{bB}_i]) \\ &= \text{Pr}[\mathbf{bB}_i] \text{Pr}[\neg \mathbf{bB}_i] \leq \text{Pr}[\neg \mathbf{bB}_i] \end{aligned}$$

Putting everything together, we get

$$\begin{aligned} &\text{Cov}(\mathbb{E}[X_i^F \mid \#\text{INT}(i, j)], \mathbb{E}[X_j^F \mid \#\text{INT}(i, j)]) \\ &\leq 4\mathbb{E}[X_i^F]^2 \text{Pr}[\neg \mathbf{bB}_i] \end{aligned}$$

and, by Lemma 5.2, we can bound this by

$$\leq 4 \left(\frac{2q^2}{2^n - q} \right)^\lambda \mathbb{E}[X_i^F]^2$$

6 F is a dCRH

In this section, we prove Lemma 4.3, which we re-state here for convenience, using a *compression argument*.

Lemma 4.3 (\exists dCRH). *For all 1-ColFind-poly-F PPT \mathcal{A} : $\exists N \in \mathbb{N} \forall n > N$:*

$$\Pr_{(\mathbf{F}, \Pi, \text{HARD}) \leftarrow \mathcal{D}} [\text{SD}(\mathcal{A}^{\mathbf{F}, \text{ColFind}}(1^n), \text{COL}_{\mathbf{F}, n}) \geq 1/4] \geq 1 - \frac{1}{n^2},$$

Firstly, we will provide the algorithms $\text{Compress}^{\mathcal{A}}$ and $\text{Decompress}^{\mathcal{A}}$ (in Section 6.1) that compress \mathbf{F}_n for a fixed n . The algorithms are proven to be compressing only assuming the adversary for that \mathbf{F}_n provides enough many true collisions in the hard set. Hence, to obtain the full proof of Lemma 4.3, we need to prove that if an adversary against dCRH \mathbf{F} exists, then there will be infinitely many n s.t. the adversary will return many hard values for significant fraction of the possible $\mathbf{F}_n \leftarrow \mathcal{D}$.

A small technicality arises from the amount of randomness the adversary uses: in $\text{Compress}^{\mathcal{A}}$, we need to save which random strings of the adversary allow to compress, as this randomness may depend on \mathbf{F} . However, as the adversary might use any polynomial amount of randomness, saving this randomness would take too much space in the encoding. Therefore, we can only consider a *subset* of all the possible adversary's randomnesses and hence we argue that a random subset of the adversary's runs will also cover a big enough set of the hard elements.

Detailed Roadmap. In Section 6.1 we discuss the algorithms $\text{Compress}^{\mathcal{A}}$ and $\text{Decompress}^{\mathcal{A}}$ (cf. Figure 6.9 for a pseudocode description). While we describe how $\text{Compress}^{\mathcal{A}}$ and $\text{Decompress}^{\mathcal{A}}$ work, we do not talk about the encoding of the output of $\text{Compress}^{\mathcal{A}}$ (and the input of $\text{Decompress}^{\mathcal{A}}$) yet, but just what information they output, namely, $\text{Compress}^{\mathcal{A}}$ simply returns a set of good adversary’s random strings, denoted R^+ , (i.e. those that allow us to compress some input x), description of the partial function F_n without the compressed inputs, denoted G , and a set of indices, denoted I , that tell which of the adversary’s queries was the hitting one. In Section 6.2 we show correctness of this algorithm.

In Section 6.3, we show how to encode the $\text{Compress}^{\mathcal{A}}$ outputs in a compressed way. That is, we show that if R^+ contains a polynomial fraction of all possible inputs to F_n , then a short encoding of (R^+, G, I) exists (Lemma 6.3). Section 6.4 is dedicated to how many collisions returned by \mathcal{A} we can use to compress. We start by showing that the hard set is large with high probability (Lemma 6.6) and therefore, a dCRH adversary finds many hard collisions (Lemma 6.8 and Lemma 6.7). Next, we prove that when the adversary finds many collisions, then the set R^+ that the encoder returns, which is the different random strings which we want \mathcal{A} to run on, will indeed be large (Lemma 6.9). Finally, in Section 6.5 we tie everything together and prove Lemma 4.3. In Section 6.5 we also prove the helper lemma that a random function is incompressible (Lemma 6.10) that is needed for the compression argument in Lemma 4.3. The proof-tree in Figure 6.7 provides an overview of the structure.

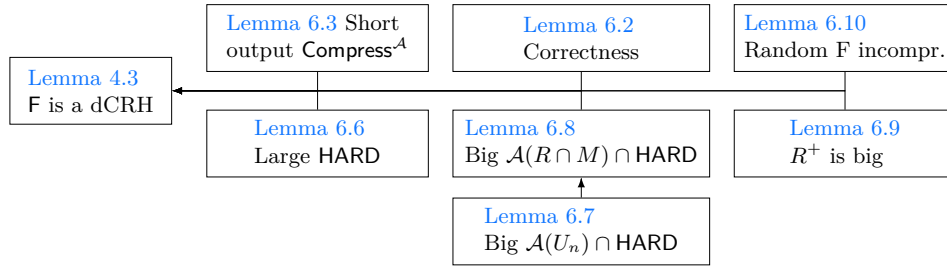


FIGURE 6.7 — Proof tree for Lemma 4.3

Terminology. To simplify arguments, we sometimes use \mathcal{A} in *normal form*.

Definition 6.1 (Normal-Form Adversaries). We say that a 1-ColFind-poly- F^5 adversary \mathcal{A} is in *normal-form* if it returns two values (x, x') such that:

1. $|x| = |x'| = n$. and $F(x) = F(x')$
2. After the call to $\text{ColFind}(h)$, \mathcal{A} will make all queries to F appearing in the path of $h(w)$ and $h(w')$.
3. Both x and x' are queried to F by \mathcal{A} exactly once.
4. \mathcal{A} will query x to F before it queries x' to F .

We can turn every PPT adversary \mathcal{A} into a normal form PPT adversary \mathcal{A}' as follows: If \mathcal{A} does not output a valid collision (1), \mathcal{A}' returns a pseudo-collision instead, e.g. $(0^n, 0^n)$. Moreover, \mathcal{A}'

⁵see Definition 4.1

additionally queries $F(x)$ and $F(x')$ before returning (x, x') . Finally, if x' is queried to F before x , \mathcal{A}' switches the two values. While the first two changes do not deteriorate the success probability, the switching of (x, x') might make (x, x') more likely than (x', x) is amongst random collisions. However, when considering (unordered) sets $\{x, x'\}$ rather than ordered pairs, the statistical distance from $\text{Col}_{h,n}$ is preserved (when also switching $\text{Col}_{h,n}$ to sets), and this property is all we need in our analysis.

6.1 The Compression and Decompression Algorithms

We start by assuming that there exists a 1-ColFind-poly-F adversary \mathcal{A} against the distributional collision resistance of (an arbitrary, fixed) F_n . We now describe two deterministic algorithms with exponential runtime and exponentially large inputs and outputs. The first one, $\text{Compress}^{\mathcal{A}}$, gets access to F_n (and ColFind) as well other parameters that do not depend on F_n (including some pre-sampled randomness, the adversary, the set HARD and the rest of the description of F for other input lengths than n), and it gives an encoding of F_n . The second algorithm, $\text{Decompress}^{\mathcal{A}}$, gets access to this encoding, and the F_n -independent parameters, but no access to F_n itself, yet it outputs the full description of F_n . We will now describe these algorithms while Section 6.2 proves correctness and Section 6.3 describes efficient representation of the compressor output.

Representation of Randomness.

The algorithms $\text{Compress}^{\mathcal{A}}$ and $\text{Decompress}^{\mathcal{A}}$ need to run \mathcal{A} on the same sequence of random tapes and therefore, $\text{Compress}^{\mathcal{A}}$ needs to communicate them to $\text{Decompress}^{\mathcal{A}}$, but including them *explicitly* requires too much space. Therefore, we use a setup which *pre-samples* a sequence of random tapes R *independently* of F_n (cf. Figure 6.8), and then $\text{Compress}^{\mathcal{A}}$ encodes the *subset of indices* in R (denoted by R^+) of random tapes (rather than the random tapes themselves) which yield collisions that we compress. The sequence R should be long enough to ensure that with high probability, sufficiently many collisions are compressed, but at the same time as short as possible to minimize the cost of encoding R^+ .

As one further refinement (which simplifies the proof of Lemma 6.8), $\text{Compress}^{\mathcal{A}}$ first finds a subset M of RAND (all possible adversary's randomness) such that $\mathcal{A}(r)$ for $r \leftarrow M$ produces (marginally) *uniform* collisions in a set COLS^+ (to be specified later) and then, $\text{Compress}^{\mathcal{A}}$ only attempts to compress values in $\mathcal{A}(R \cap M)$.

Compression Algorithm (Fig 6.9).

The compression algorithm identifies random tapes leading to new true hard collisions and, for each such tape, stores the necessary information to later run \mathcal{A} on the same tape and recover the compressed information. In addition to any points \hat{x} which $(x, x') \leftarrow \mathcal{A}(r)$ has queried to F the compression algorithm stores the *query index* of x (we assume \mathcal{A} is in normal form) and marks the randomness r . On the re-run in the $\text{Decompress}^{\mathcal{A}}$ this allows to recover y and allows answering queries for x' (as x is called before x'). In the end, further mappings in HARD_n which have not been compressed or stored as additional queries are added to the encoding (G). Importantly, storing r allows $\text{Decompress}^{\mathcal{A}}$ to only run on useful random tapes and we thus only need to store additional queries \mathcal{A} makes for these tapes (instead of all tapes in R). The encoded

```

Setup( $\mathcal{A}$ )
-----
 $a \leftarrow$  Max. random tape for  $\mathcal{A}$ 
 $R \leftarrow$   $\{S \subseteq \{0, 1\}^a : |S| = 2^n\}$ 
return  $R$ 

```

FIGURE 6.8 — Compressor setup

$\text{Compress}^{\mathcal{A}} \left(\begin{array}{l} \text{inst} = (F_n), \\ \text{aux} = (R, \text{HARD}, \Pi, F \setminus F_n) \end{array} \right)$ <p> $R^+ \leftarrow \emptyset$ // set of good randomness $C \leftarrow \emptyset$ // set of F_n-inputs we compress $I \leftarrow \varepsilon$ // list of indices $G \leftarrow \emptyset$ // in-out mappings of things not in C $G \leftarrow G \cup \{(x, F_n(x)) \mid x \in \{0, 1\}^n \setminus \text{HARD}_n^{\text{in}}\}$ compute COLS^+ // cf Lemma 6.7 $M \stackrel{\text{max}}{\leftarrow} \text{SD}(\mathcal{A}(U_M), \text{COL}_{h,n} \text{COLS}^+) = 0$ for $r \in R \cap M$ $x, x' \leftarrow \mathcal{A}^{\text{F}, \text{ColFind}[F]}(r)$ if $x = x'$ or $x, x' \notin \text{HARD}_n^{\text{in}}$ or $x' \in G$ then continue $R^+ \leftarrow R^+ \cup \{r\}; \quad C \leftarrow C \cup \{x'\}$ $I \leftarrow I \parallel \text{qry_index}(x)$ $G \leftarrow G \cup (\text{qry}_{\mathcal{A}}^{\text{F}_n}(r) \setminus \{x'\})$ $G \leftarrow G \cup (\text{HARD} \setminus C)$ return $\text{encode}_{(\text{Section 6.3})}(R^+, G, I)$ </p>	$\text{Decompress}^{\mathcal{A}} \left(\begin{array}{l} \text{desc} \\ \text{aux} = (R, \text{HARD}, \Pi, F \setminus F_n) \end{array} \right)$ <p> $(R^+, G, I) \leftarrow \text{decode}_{(\text{Section 6.3})}(\text{desc})$ for $(r, i) \in (R^+, I)$ Run $x, x' \leftarrow \mathcal{A}^{\text{SimOracles}[G]}(r)$: </p> <div style="border: 1px dashed black; padding: 10px; margin: 10px 0;"> <p style="text-align: center; margin: 0;">SimOracles$[G]$</p> <hr style="border: 0.5px solid black; margin: 0 0 5px 0;"/> $l_q \leftarrow \varepsilon$ // list of queries in \mathcal{A} On ColFind-query h: $(w, w') \leftarrow \text{ColFind}[G \cup (F \setminus F_n)](h)$ return (w, w') On F-query \hat{x}: $l_q \leftarrow l_q \parallel \hat{x}$ if $\hat{x} \in G \cup F \setminus F_n$: return $(G \cup F \setminus F_n)[\hat{x}]$ else : $x \leftarrow l_q[i]$ $G[\hat{x}] \leftarrow G[x]$ abort this run of \mathcal{A} </div> <p>return G</p>
--	--

FIGURE 6.9 — Compression Algorithms⁶

output then consists of the subset of successful random tapes R^+ , the sequence of query indices I and the partial mapping G .

Decompression Algorithm (Fig 6.9). $\text{Decompress}^{\mathcal{A}}$ recovers the sequence of random tapes which $\text{Compress}^{\mathcal{A}}$ used from the set R^+ and the order of them from R . It then runs \mathcal{A} on $r \in R^+$ to restore the mapping $x' \mapsto y$ compressed in this run. As $\text{Decompress}^{\mathcal{A}}$ does not have the full description of F it needs to emulate the oracles for \mathcal{A} . As the compression algorithm included all F queries made by \mathcal{A} apart from x' in the encoded information, the first F -query which $\text{Decompress}^{\mathcal{A}}$ can not answer will be $F(x')$ at which point \mathcal{A} can be aborted: As \mathcal{A} queries x first, we can recover x' by keeping a list of all queries l_q and using the query index i and set $F(x') := F(x)$. Emulating $(w, w') \leftarrow \text{ColFind}$ requires some care: while normal form adversaries will have evaluated h on w and w' and thus all F queries related to them are available to $\text{Decompress}^{\mathcal{A}}$, this is not necessarily true for all values \hat{w} considered by ColFind internally. However, none of the \hat{w} can have been returned by ColFind and thus it is correct to *skip* any candidate (w, w') where F on either value cannot be evaluated using the partial description of F . If no candidate remains, ColFind must have returned \perp .

6.2 Correctness of Compression

Lemma 6.2 (Correctness of $\text{Compress}^{\mathcal{A}}$ and $\text{Decompress}^{\mathcal{A}}$). *For all adversaries \mathcal{A} , randomness (R, HARD, Π) , 2-regular functions F and all n :*

$$\text{Decompress}^{\mathcal{A}}(\text{Compress}^{\mathcal{A}}(F_n, \text{aux}), \text{aux}) = F_n$$

where $\text{aux} = (R, \text{HARD}, \Pi, F \setminus F_n)$.

Proof. First note that for all $x \notin C$ (and especially for all $x \notin \{0, 1\}^n$), $F(x) = G[x]$ is given directly to Decompress^A , therefore, the reconstructed F is correct for all of them. Therefore, all that remains is to argue that for all $\hat{x} \in C$, $G[\hat{x}]$ is correctly restored. Let us fix some \hat{x} .

As we save a randomness r in R^+ as well as a query index i for each x added to C , such a pair $(r, i) \in (R^+, I)$ must exist. As we go through all of these pairs in the same order as in Compress^A , at some point, \mathcal{A} is run on r . The initial call to ColFind is correctly simulated: Based on our partial information, we will take the first set (w, w') such that ColFind would output those and the path of both w and w' is defined in G (or return \perp , if no such pair exists). As we save the paths of w and w' in G and that is all the information ColFind needs to decide which collision to return, this indeed emulates ColFind perfectly.

For the further calls to the F oracle, we have two cases: Either the answer for the query x saved in G , and we can therefore return $G[x]$, or the answer to the query is not saved in G . In the latter case, this query must be \hat{x} : With the fixed randomness, \mathcal{A} is deterministic and we have answered all previous queries as the real oracles. Therefore, also this new query must have happened in Compress^A . However, as Compress^A saves the answers to all queries except for \hat{x} in G , this new query must be \hat{x} .

Now, we just have to complete the mapping from $G[\hat{x}]$ to $F(\hat{x})$. Using the fact that \mathcal{A} is in normal-form, we know that there has been a previous query x to F such that (x, \hat{x}) will be the collision later returned by \mathcal{A} . Further, we have the query index i of x , i.e., we know which of the previous queries x was (it is saved in our list l_q at position i). Therefore, we can now read out $y \leftarrow G[x]$, and, as x is a collision with \hat{x} , set $G[\hat{x}] \leftarrow y$.

As we have now shown that for each \hat{x} , we recover the mapping $\hat{x} \mapsto F(\hat{x})$, and as this was the only information missing in the encoding of G , we have now shown that Decompress^A completely recovers F .

6.3 Compressed Encoding

Now we will show how to encode the output of Compress^A , i.e. R^+, G, I , compactly.

Lemma 6.3 (Output length of Compress^A is short). *Let $\frac{1}{\text{com}}$ be the fraction of compressed inputs, i.e. $|R^+| = 2^n \frac{1}{\text{com}}$ and let $|\mathcal{A}|$ be an upper bound on the number of oracle queries \mathcal{A} makes. We assume both com and $|\mathcal{A}|$ to be polynomial. Then, the output length of Compress^A is bounded by*

$$\leq 2^n(n - \log e) - \frac{1}{2\text{com}} 2^n n.$$

Proof. Using (R^+, G, I) we encode the function F_n .

Representation of R^+ As R^+ is a subset of R we can encode it using subset encoding⁷ resulting in

$$\log \left(\binom{2^n}{\frac{1}{\text{com}} 2^n} \right)$$

description size for R^+ . Note that R^+ inherits the order from R , so it is enough to encode R^+ as a *set* rather than *ordered set* or list.

⁶for simplicity, we write for example Π as input to the algorithms, even though Π is infinite size. However, only a finite part of Π is needed, so w.l.o.g. one could only write a finite subset of Π (and of other infinite-size inputs) as the input.

⁷Create a list of all subsets and encode the subset by its index into this list

Representation of I Then we need $\log|\mathcal{A}|$ bits to encode the query offset for all the compressed inputs, resulting in

$$\frac{1}{\text{com}} \cdot 2^n \cdot \log|\mathcal{A}|$$

description size for I .

Representation of G A 2-regular function $F : \{0, 1\}^n \rightarrow \{0, 1\}^{n-1}$ can be represented as a sequence of $y \leftarrow x_1, x_2$ sorted by y . One can represent this function as a sequence of (x_1, x_2) pairs for all y 's in order. Now, since G is a partial function, that is, some y 's have only one corresponding x_i . We can represent such G as follows:

Encode the set of missing x_i 's in the table as a set, that takes $\log\left(\frac{2^n}{\text{com}}\right)$

Encode similarly the set of y 's that are missing an input. We encode the subset of such y -values using $\log\left(\frac{2^{n-1}}{\text{com}}\right)$ bits.⁸

Then, we only need to encode a permutation of the remaining x_i values that are in the table. This requires $\log\left(\left(1 - \frac{1}{\text{com}}\right)2^n\right)!$ bits.

All together G requires:

$$\begin{aligned} & \underbrace{\log\left(\frac{2^{n-1}}{\text{com}}\right)}_{\text{set of compressed } y\text{'s}} + \underbrace{\log\left(\frac{2^n}{\text{com}}\right)}_{\text{set of compressed } x\text{'s}} + \underbrace{\log\left(\left(1 - \frac{1}{\text{com}}\right)2^n\right)!}_{\text{permutation of } x\text{'s}} \\ & \leq 2 \log\left(\frac{2^n}{\text{com}}\right) + \log\left(\left(1 - \frac{1}{\text{com}}\right)2^n\right)! \end{aligned}$$

Taking all together, the length of the output of the compressor is (in bits):

$$\begin{aligned} & \underbrace{\log\left(\frac{2^n}{\text{com}}\right)}_{R^+} + \underbrace{\frac{1}{\text{com}} \cdot 2^n \log|\mathcal{A}|}_{I} + \underbrace{2 \log\left(\frac{2^n}{\text{com}}\right) + \log\left(\left(1 - \frac{1}{\text{com}}\right)2^n\right)!}_{G} \\ & \leq 3 \log\left(\frac{2^n}{\text{com}}\right) + 2^n \frac{\log|\mathcal{A}|}{\text{com}} + \underbrace{\log\left(\frac{\left(\left(1 - \frac{1}{\text{com}}\right)2^n\right)^{\left(1 - \frac{1}{\text{com}}\right)2^n}}{e^{\left(1 - \frac{1}{\text{com}}\right)2^n}}\right)}_{\text{by Lemma 6.4 (Lower Bound. Factorial)}} \\ & \leq \underbrace{\frac{3}{\text{com}} 2^n (n - n + \log \text{com} + \log e)}_{\text{by Lemma 6.5 (Binomial Coeff.)}} + \frac{2^n}{\text{com}} \log|\mathcal{A}| + \left(1 - \frac{1}{\text{com}}\right) 2^n (n - \log e) \\ & = 2^n (n - \log e) - \frac{1}{\text{com}} 2^n \underbrace{(n - 3 \log \text{com} - \log|\mathcal{A}| - 4 \log e)}_{> n/2 \text{ for big enough } n, \text{ if } |\mathcal{A}|, \text{com polynomials}} \\ & \leq 2^n (n - \log e) - \frac{1}{2\text{com}} 2^n n \tag{6.19} \end{aligned}$$

Lemma 6.4 (Lower Bounding Factorial). $\forall k \in \mathbb{N} : k! \geq k^k / e^k$

Proof. By definition of the e we get $\forall x : e^x = 1 + \frac{x^1}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots$, consequently we have $\forall x, k : e^x \geq \frac{x^k}{k!}$, in particular, let $x = k$ and now $e^k \geq \frac{k^k}{k!}$, which proves the statement.

⁸since fraction of compressed inputs is $1/\text{com}$, and the function F is two-regular, it follows that fraction of outputs for which an input is compressed is exactly $2/\text{com}$

Lemma 6.5 (Binomial Coefficient). $\forall a, b \in \mathbb{N} : \log \binom{a}{b} \leq b(\log a - \log b + \log e)$

Proof. $\log \binom{a}{b} = \log \frac{a!}{(a-b)!b!} \leq \log \frac{a^b}{b!} = b \log a - \log b! \leq b \log a - \log b^b / e^b = b \log a - b \log b + b \log e$, where the last inequality is by Lemma 6.4.

6.4 Adversary Finds Many Hard Collisions

In this section, we will prove the heart piece of our compression argument: Showing that we compress a polynomial fraction $\frac{1}{\text{com}}$ of all inputs. A technical challenge here is to show that even if we only look at a subset of all possible randomness, with high probability, our adversary still returns enough *distinct* collisions.

We start by showing that, with high probability, we have a sufficiently large HARD_n set. In the definition of the hard set, we flip a coin for every value whether it is in the hard set or not, hence the expected size of the hard set is $2^{n-1}/2$. We show that w.h.p. the hard set covers at least half the expected space.

Lemma 6.6 (Hard Set is Large w.h.p.).

$$\Pr_{\text{HARD}_n \leftarrow \mathcal{D}} \left[|\text{HARD}_n^{\text{out}}| \leq \frac{1}{4} 2^{n-1} \right] \leq 2^{-n}$$

Proof. By definition of \mathcal{D} , for every n , we go through all possible 2^{n-1} outputs y of F_n and toss a coin to decide whether that y is in $\text{HARD}_n^{\text{out}}$ or not. The probability that y is hard is $1/2$.

Now

$$\begin{aligned} & \Pr_{\text{HARD}_n \leftarrow \mathcal{D}} \left[|\text{HARD}_n^{\text{out}}| \leq \frac{1}{4} 2^{n-1} \right] \\ &= \Pr \left[\frac{1}{2} 2^{n-1} - |\text{HARD}_n^{\text{out}}| \geq \frac{1}{4} 2^{n-1} \right] \\ &= \frac{1}{2} \Pr \left[\left| \frac{1}{2} - \frac{|\text{HARD}_n^{\text{out}}|}{2^{n-1}} \right| \geq \frac{1}{4} \right] \\ &\leq e^{-2 \cdot 2^{n-1} (\frac{1}{4})^2} \quad \text{|by Lemma 3.5 (Hoeffding)} \\ &\leq 2^{-n} \end{aligned}$$

Now, we define a set COLS^+ of true, i.e. non-pseudo, hard collisions that the adversary returns with reasonably high probability and we show that the set must be large, if the adversary successfully approximates $\text{COL}_{F_n, n}$.

Lemma 6.7 (Many Hard Collisions). *Let $|\text{HARD}_n^{\text{out}}| \geq \frac{1}{4}$ and let \mathcal{A} be s.t. $\text{SD}[\mathcal{A}^{F_n}(1^n), \text{COL}_{F_n, n}] \leq \frac{1}{4}$. Denote by COLS^+ the set of unordered collisions (i.e., $(x, x') = (x', x)$) in $\text{COL}_{F_n, n}$ that are true and hard and are returned by the adversary with probability at least 2^{-n-1} . Then, COLS^+ is at least a $\frac{1}{16}$ fraction of all collisions in $\text{COL}_{F_n, n}$.*

Proof. The adversary has a statistical distance from $\text{COL}_{F_n, n}$ of at most $\frac{1}{4}$. The probability of a collision from $\text{COL}_{F_n, n}$ to be a true collision in the HARD set is $\frac{1}{2} \cdot \frac{1}{4} = \frac{1}{16}$ (as half of the collisions are pseudocollisions).

In the worst case (i.e. adversary returns as few true hard collisions as possible), the adversary returns all the collisions in $\text{COL}_{F_n, n}$ that are easy or pseudo collisions with higher or equal probability as $\text{COL}_{F_n, n}$.

Let's consider the remaining (i.e. hard, true) collisions as sets (i.e. order of the colliding pair does not matter). Now $\text{COL}_{F_n, n}$ returns each collision with probability 2^{-n} .

Adversary on the other hand, returns some of these collisions with probability $< 2^{-n-1}$ (let's call these COLS^-) and the rest with probability $\geq 2^{-n-1}$, called COLS^+ .

In the worst case, COLS^- is as large as possible. However, since the statistical distance is $1/4$, we know that

$$\begin{aligned} \sum_{(x, x') \in \text{COLS}^-} |\Pr[\mathcal{A} = (x, x')] - 2^{-n}| &\leq 1/4 \\ \sum_{(x, x') \in \text{COLS}^-} \underbrace{2^{-n} - \Pr[\mathcal{A} = (x, x')]}_{> 2^{-n} - 2^{-n-1} = 2^{-n-1}} &\leq 1/4 \\ |\text{COLS}^-| 2^{-n-1} &\leq 1/4 \\ |\text{COLS}^-| &\leq \frac{1}{4} 2^{n+1} \end{aligned}$$

Hence

$$\begin{aligned} |\text{COLS}^+| &= \frac{1}{8} 2^{n-1} - |\text{COLS}^-| \\ |\text{COLS}^+| &\geq \frac{1}{8} 2^{n-1} - \frac{1}{16} 2^{n-1} = \frac{1}{16} 2^{n-1} \end{aligned}$$

which finishes the proof.

Next, we want to show that our adversary returns enough *distinct* collisions in our set COLS^+ , even when we only run it on the random strings in R . This requires a few technical tricks, as we want to bound the size by a Chernoff bound, which requires *independence* between the different random variables. To achieve this, we first define a maximal set $M \subset \text{RAND}$ on which the adversary returns all elements from COLS^+ *uniformly*, and then show that our adversary, that runs on R , uses randomness from M often enough. Next, we want to count how many *distinct* collisions the adversary returns, and again, we have to make sure that the drawings are independent of each other. We achieve this by making the probability that a collision in COLS^+ is not counted constant. Of course, if we have already seen a collision, then we don't count it. Additionally, we don't count a collision if it is one of the first values of COLS^+ , where the size of this first segment is determined in a way such that its size plus the size of the already counted collisions is always constant, making the probability of finding a new collision constant as well.

Note that for both, we slightly deviate from what Compress^A does, but in both cases, Compress^A will potentially just find more collisions, so it is fine for a lower bound.

Lemma 6.8 (Many True Hard Collisions in $\mathcal{A}(R \cap M)$). *With overwhelming probability over the choice of R (a uniformly random subset of size 2^n of RAND , all possible adversary's random strings), the number of true collisions in the hard set returned by the adversary \mathcal{A} on all random strings in $R \cap M$ is large:*

$$\Pr_{R \leftarrow \text{RAND}} \left[|\mathcal{F}(\mathcal{A}(R \cap M)) \cap \text{HARD}_n^{cl}| \geq \underbrace{\frac{1}{320} 2^n}_{:= \frac{1}{\text{poly}} 2^{n-1}} \right] \geq 1 - 2^{-\Omega(n)}$$

where HARD_n^{cl} is the set of true hard collisions.

Proof. By Lemma 6.7, we know that there exists a set of true hard collision COLS^+ such that for each of these collisions, we have a probability $\geq 2^{-n-1}$ that \mathcal{A} returns them. We now define a maximal set $M \subseteq \text{RAND}$ (where RAND is the set of all possible adversary's randomness) of the randomness of \mathcal{A} such that \mathcal{A} on M *uniformly* outputs values from COLS^+ .

Recall that by Lemma 6.7:

$$|\text{COLS}^+| \geq \frac{1}{16} 2^{n-1}$$

Next, as in the $\text{Compress}^{\mathcal{A}}$ algorithm, we define $R' = R \cap M$ as the intersection of R and M . We want to argue that R' is large. For each $r \in R$, the probability that it is in M is some probability bounded by

$$\Pr_{r \leftarrow \text{RAND}}[r \in M] \geq \frac{1}{16} 2^{n-1} 2^{-n-1} = \frac{1}{64},$$

where the first inequality follows from taking a sum over all collisions in COLS^+ of the minimum probability that the adversary returns that collision.

Now, w.l.o.g. $|\text{RAND}| > 2^{4n}$ and R can be thought of as sampling 2^n values independently and uniformly from RAND .⁹ Consider a random variable X_r that is 1 if $r \in M$ and 0 else. Now we can use a Chernoff bound to show that

$$\Pr_R \left[|R'| = \sum_{r \in R} X_r \leq \frac{1}{2} 2^n \frac{1}{64} \right] \leq 2^{-\frac{1}{8} 2^n \frac{1}{64}}. \quad (6.20)$$

Now, we want to argue that the number of true collisions in the hard set the $\text{Compress}^{\mathcal{A}}$ algorithm sees is large enough.

Let us consider the set COLS^+ as an ordered list of at least $m := \frac{1}{16} 2^{n-1}$ many elements (collisions).

Define sets $K_i, i \in 0, \dots, |R'|$ as follows:

- $K_0 = \emptyset$
- for each i , take the i 'th $r \in R'$ and if $\mathcal{A}(r)$ is in K_{i-1} or in the first $\frac{1}{2}m - |K_{i-1}|$ values in COLS^+ then $K_i = K_{i-1}$. Else $K_i = K_{i-1} \cup \mathcal{A}(r)$.

Define binary random variables $X_i, i \in 1, \dots, m' := \min(|R'|, m/2)$ as follows $X_i = 1$ exactly when $|K_i| > |K_{i-1}|$. Note that these X_i are independent and $\Pr[X_i = 1] = 1/2$.

For each iteration, we draw a uniform sample from M as randomness r , as R is a random function, for a uniform $r \in M$, the adversary will create a uniform (w, w') in COLS^+ . Further, note that the expected size of $K = K_{m'}$ is:

$$\mathbb{E} \left[\sum_{i=1}^{m'} X_i \right] = \mathbb{E}[|K|] = \frac{1}{2} m'.$$

As the X_i are independent, we can use Chernoff to bound the probability that K is more than a small fraction smaller than expected:

$$\Pr \left[\sum_{i=1}^{m'} X_i < \left(1 - \frac{1}{5}\right) \frac{1}{2} m' \right] < 2^{-\frac{1}{50} m'}$$

⁹even though we don't allow repetitions in practice, the difference to uniform is negligible as the probability of repetitions is exponentially small.

By (6.20) R' has exponential size, hence m' is also exponential and hence with overwhelming probability,

$$|K| \geq \frac{2}{5}m' \geq \frac{2}{5} \frac{1}{2} 2^n \frac{1}{64} = \frac{1}{320} 2^n.$$

Note that our analysis here does not quite match what the $\text{Compress}^{\mathcal{A}}$ algorithm does: It does not discard values from the beginning of COLS^+ and it continues until $i = |R'|$, not only until $i = m'$. However, both differences mean that $\text{Compress}^{\mathcal{A}}$ potentially finds even more collisions, and therefore $|K|$ holds as a lower bound for $|\mathcal{F}(\mathcal{A}(R \cap M)) \cap \text{HARD}_n^{cl}|$.

Now we have argued why the set of collisions that $\text{Compress}^{\mathcal{A}}$ goes through in the for-loop is large. However, one iteration in the loop only compresses a collision *provided that a previous iteration did not add it to G* . Next, we show, via a simple counting argument, that a big fraction of the iterations successfully compress nonetheless.

Lemma 6.9 (Large $|\mathcal{A}(R \cap M)| \Rightarrow$ significant compression). *Let $R \cap M$ be s.t. $\mathcal{A}(R \cap M)$ contains $2^{n-1}/\text{ply}$ distinct true collisions in HARD_n . Then $|R^+| \geq 2^{n-1}/(\text{ply}|\mathcal{A}|)$.*

Proof. Let $S \leftarrow R \cap M$ be s.t. $\mathcal{A}(R \cap M)$ contains $2^{n-1}/\text{ply}$ distinct true collisions in HARD_n (which is implied by Lemma 6.8). When we run the compressor, we add new bitstrings from S to R^+ t times. How large must t be?

Every time we add a new element to R^+ , we add at most $|\mathcal{A}|$ many elements from S to G . How many times (at least) we can add values to R^+ before S has been exhausted? In the worst case, we have $|S| = t|\mathcal{A}| \Leftrightarrow t = |S|/|\mathcal{A}| = 2^{n-1}/(\text{ply}|\mathcal{A}|)$. Thus, at least $2^{n-1}/(\text{ply}|\mathcal{A}|)$ many values are in R^+ .

6.5 Proof of Lemma 4.3

Now we are ready to prove Lemma 4.3, restated here for convenience:

Lemma 4.3 (\exists dCRH). *For all 1-ColFind-poly-F PPT \mathcal{A} : $\exists N \in \mathbb{N} \forall n > N$:*

$$\Pr_{(\mathcal{F}, \Pi, \text{HARD}) \leftarrow \mathcal{D}} [\text{SD}(\mathcal{A}^{\mathcal{F}, \text{ColFind}}(1^n), \text{COL}_{\mathcal{F}, n}) \geq 1/4] \geq 1 - \frac{1}{n^2},$$

Proof. Suppose for contradiction that there is an adversary \mathcal{A} s.t. for infinitely many n :

$$\Pr_{(\text{HARD}, \Pi, \mathcal{F}) \leftarrow \mathcal{D}} [\text{SD}(\mathcal{A}^{\mathcal{F}, \text{ColFind}}(1^n), \text{COL}_{\mathcal{F}, n}) < 1/4] > \frac{1}{n^2},$$

By Lemma 6.6, the hard set covers at least $\frac{1}{4}$ fraction of the output space with probability $> 1 - 2^{-n}$. Hence, we can fix the hard set to such a large set without altering the above probability more than by an additive negligible fraction.

Now fix $\text{HARD}_n, \Pi, \mathcal{F}/\mathcal{F}_n$ s.t. the above probability is as large as possible, now for infinitely many n :

$$\Pr_{\mathcal{F}_n \leftarrow \mathcal{D}} [\text{SD}(\mathcal{A}^{\mathcal{F}, \text{ColFind}}(1^n), \text{COL}_{\mathcal{F}, n}) < 1/4] > \frac{1}{n^2} - \text{negl}(\lambda),$$

That is, at least $\frac{1}{n^2} - \text{negl}(\lambda)$ fraction of all possible two-regular \mathcal{F}_n are broken by \mathcal{A} . Let's call these $\mathcal{F}_{\text{broken}}$.

Now we pre-sample the randomness R for the adversary and define $\text{Compress}^{\mathcal{A}}, \text{Decompress}^{\mathcal{A}}$ as in Figure 6.9, and with probability $1 - \text{negl}$ (by Lemma 6.8) that randomness R is such that

$|\mathbb{F}_n(\mathcal{A}(R \cap M)) \cap \text{HARD}_n^{cl}| \geq \frac{1}{\text{ply}} 2^{n-1}$, where $\text{ply} = 160$. Now with that randomness, the $\text{Compress}^{\mathcal{A}}$ has output length at most

$$2^n(n - \log e) - \frac{1}{2\text{com}} 2^n n \quad (6.21)$$

by [Lemma 6.3](#). Note that $|\mathcal{A}|$, the number queries that the adversary makes, is polynomial by assumption and additionally com is defined as $|R^+| = 2^n \frac{1}{\text{com}}$, i.e. the fraction of inputs to \mathbb{F}_n that we compress, so by [Lemma 6.9](#) $\text{com} = \text{ply}|\mathcal{A}|$ which is also a polynomial. That is, all functions in $\mathcal{F}_{\text{broken}}$ have a short encoding.

However, $\mathcal{F}_{\text{broken}}$ covers a non-negligible fraction of 2-regular functions, while [Lemma 6.10](#) says that only neglig. fraction can have short description.

Below we show that [\(6.21\)](#) is indeed shorter than $\log(2^{n!}/2) - \log^2 n$ and get a contradiction with [Lemma 6.10](#).

$$\begin{aligned} \log \frac{2^{n!}}{2} - \log^2 n &\geq \log \frac{(2^n)^{2^n}}{2e^{2^n}} - \log^2 n && \quad | \text{ By Lemma 6.4} \\ &= 2^n n - \log 2 - \log e^{2^n} - \log^2 n \\ &= 2^n n - 1 - 2^n \log e - \log^2 n \\ &= 2^n(n - \log e) - 1 - \log^2 n \\ &> 2^n(n - \log e) - \frac{1}{2\text{com}} 2^n n && \quad | \text{ Same as (6.21)} \end{aligned}$$

where the last inequality clearly holds for big enough n .

Lemma 6.10 (Encoding Length of a 2-Regular Function). *The description of a 2-regular function $f : \{0, 1\}^n \rightarrow \{0, 1\}^{n-1}$ requires at least $\log(2^{n!}/2) - \log^2 n$ bits except with negligible probability over the choice of the random 2-regular function f .*

Proof. Let $\mathcal{F} := \{f = f'_{1, \dots, n-1} | f' : \{0, 1\}^n \rightarrow \{0, 1\}^n \text{ is a bijection}\}$. Fix any encoding $\text{Encode} : \mathcal{F} \rightarrow \{0, 1\}^*$ for the functions in \mathcal{F} , that is, each function $f \in \mathcal{F}$ is mapped to some bitstring $\text{Encode}(f)$ and Encode is an injective function. There are $2^{n!}/2$ many functions in \mathcal{F} and $< 2^m$ bitstrings of length $< m$. Let S_m be the set of functions in \mathcal{F} s.t. $|\text{Encode}(f)| < m, \forall f \in S_m$. Now, let $m := \log(2^{n!}/2) - \log^2 n$. Then, the probability that a random permutation f is in S_m is:

$$\begin{aligned} \Pr_{f \leftarrow \mathcal{F}}[|\text{Encode}(f)| < m] &= \Pr_{f \leftarrow \mathcal{F}}[f \in S_m] \\ &= \frac{|S_m|}{|\mathcal{F}|} \leq \frac{2^m - 1}{2^{n!}/2} < \frac{2^m}{2^{n!}/2} = \frac{1}{2^{\log^2 n}} = \text{negl}(n) \end{aligned}$$

References

- [Bae14] Paul Baecker. *Simon's Circuit*. Cryptology ePrint Archive, Report 2014/476. 2014. URL: <https://eprint.iacr.org/2014/476>.
- [BBKPV16] Nir Bitansky, Zvika Brakerski, Yael Tauman Kalai, Omer Paneth, and Vinod Vaikuntanathan. "3-Message Zero Knowledge Against Human Ignorance". In: *TCC 2016-B, Part I*. Ed. by Martin Hirt and Adam D. Smith. Vol. 9985. LNCS. Springer, Berlin, Heidelberg, Oct. 2016, pp. 57–83. DOI: [10.1007/978-3-662-53641-4_3](https://doi.org/10.1007/978-3-662-53641-4_3).

- [BHKY19] Nir Bitansky, Iftach Haitner, Ilan Komargodski, and Eylon Yogev. “Distributional Collision Resistance Beyond One-Way Functions”. In: *EUROCRYPT 2019, Part III*. Ed. by Yuval Ishai and Vincent Rijmen. Vol. 11478. LNCS. Springer, Cham, May 2019, pp. 667–695. DOI: [10.1007/978-3-030-17659-4_23](https://doi.org/10.1007/978-3-030-17659-4_23).
- [BKP18] Nir Bitansky, Yael Tauman Kalai, and Omer Paneth. “Multi-collision resistance: a paradigm for keyless hash functions”. In: *50th ACM STOC*. Ed. by Ilias Diakonikolas, David Kempe, and Monika Henzinger. ACM Press, June 2018, pp. 671–684. DOI: [10.1145/3188745.3188870](https://doi.org/10.1145/3188745.3188870).
- [BL13] Daniel J. Bernstein and Tanja Lange. “Non-uniform Cracks in the Concrete: The Power of Free Precomputation”. In: *ASIACRYPT 2013, Part II*. Ed. by Kazuo Sako and Palash Sarkar. Vol. 8270. LNCS. Springer, Berlin, Heidelberg, Dec. 2013, pp. 321–340. DOI: [10.1007/978-3-642-42045-0_17](https://doi.org/10.1007/978-3-642-42045-0_17).
- [BT24] Jan Buzek and Stefano Tessaro. “Collision Resistance from Multi-collision Resistance for All Constant Parameters”. In: *CRYPTO 2024, Part V*. Ed. by Leonid Reyzin and Douglas Stebila. Vol. 14924. LNCS. Springer, Cham, Aug. 2024, pp. 429–458. DOI: [10.1007/978-3-031-68388-6_15](https://doi.org/10.1007/978-3-031-68388-6_15).
- [DI06] Bella Dubrov and Yuval Ishai. “On the randomness complexity of efficient sampling”. In: *38th ACM STOC*. Ed. by Jon M. Kleinberg. ACM Press, May 2006, pp. 711–720. DOI: [10.1145/1132516.1132615](https://doi.org/10.1145/1132516.1132615).
- [FGHMY24] Lukás Folwarczný, Mika Göös, Pavel Hubáček, Gilbert Maystre, and Weiqiang Yuan. “One-Way Functions vs. TFNP: Simpler and Improved”. In: *ITCS 2024*. Ed. by Venkatesan Guruswami. Vol. 287. LIPIcs, Jan. 2024, 50:1–50:14. DOI: [10.4230/LIPIcs.ITCS.2024.50](https://doi.org/10.4230/LIPIcs.ITCS.2024.50).
- [GT00] Rosario Gennaro and Luca Trevisan. “Lower Bounds on the Efficiency of Generic Cryptographic Constructions”. In: *41st FOCS*. IEEE Computer Society Press, Nov. 2000, pp. 305–313. DOI: [10.1109/SFCS.2000.892119](https://doi.org/10.1109/SFCS.2000.892119).
- [HHR07] Iftach Haitner, Jonathan J. Hoch, Omer Reingold, and Gil Segev. “Finding Collisions in Interactive Protocols - A Tight Lower Bound on the Round Complexity of Statistically-Hiding Commitments”. In: *48th FOCS*. IEEE Computer Society Press, Oct. 2007, pp. 669–679. DOI: [10.1109/FOCS.2007.27](https://doi.org/10.1109/FOCS.2007.27).
- [HK06] Shai Halevi and Hugo Krawczyk. “Strengthening Digital Signatures Via Randomized Hashing”. In: *CRYPTO 2006*. Ed. by Cynthia Dwork. Vol. 4117. LNCS. Springer, Berlin, Heidelberg, Aug. 2006, pp. 41–59. DOI: [10.1007/11818175_3](https://doi.org/10.1007/11818175_3).
- [HR04] Chun-Yuan Hsiao and Leonid Reyzin. “Finding Collisions on a Public Road, or Do Secure Hash Functions Need Secret Coins?” In: *CRYPTO 2004*. Ed. by Matthew Franklin. Vol. 3152. LNCS. Springer, Berlin, Heidelberg, Aug. 2004, pp. 92–105. DOI: [10.1007/978-3-540-28628-8_6](https://doi.org/10.1007/978-3-540-28628-8_6).
- [HT98] Satoshi Hada and Toshiaki Tanaka. “On the Existence of 3-Round Zero-Knowledge Protocols”. In: *CRYPTO’98*. Ed. by Hugo Krawczyk. Vol. 1462. LNCS. Springer, Berlin, Heidelberg, Aug. 1998, pp. 408–423. DOI: [10.1007/BFb0055744](https://doi.org/10.1007/BFb0055744).

- [IL89] Russell Impagliazzo and Michael Luby. “One-way Functions are Essential for Complexity Based Cryptography (Extended Abstract)”. In: *30th FOCS*. IEEE Computer Society Press, Oct. 1989, pp. 230–235. DOI: [10.1109/SFCS.1989.63483](https://doi.org/10.1109/SFCS.1989.63483).
- [Imp95] Russell Impagliazzo. “A Personal View of Average-Case Complexity”. In: *Proceedings of the Tenth Annual Structure in Complexity Theory Conference, Minneapolis, Minnesota, USA, June 19-22, 1995*. IEEE Computer Society, 1995, pp. 134–147. DOI: [10.1109/SCT.1995.514853](https://doi.org/10.1109/SCT.1995.514853). URL: <https://doi.org/10.1109/SCT.1995.514853>.
- [IR90] Russell Impagliazzo and Steven Rudich. “Limits on the Provable Consequences of One-way Permutations”. In: *CRYPTO’88*. Ed. by Shafi Goldwasser. Vol. 403. LNCS. Springer, New York, Aug. 1990, pp. 8–26. DOI: [10.1007/0-387-34799-2_2](https://doi.org/10.1007/0-387-34799-2_2).
- [KNY18] Ilan Komargodski, Moni Naor, and Eylon Yogev. “Collision Resistant Hashing for Paranoids: Dealing with Multiple Collisions”. In: *EUROCRYPT 2018, Part II*. Ed. by Jesper Buus Nielsen and Vincent Rijmen. Vol. 10821. LNCS. Springer, Cham, Apr. 2018, pp. 162–194. DOI: [10.1007/978-3-319-78375-8_6](https://doi.org/10.1007/978-3-319-78375-8_6).
- [KY18] Ilan Komargodski and Eylon Yogev. “On Distributional Collision Resistant Hashing”. In: *CRYPTO 2018, Part II*. Ed. by Hovav Shacham and Alexandra Boldyreva. Vol. 10992. LNCS. Springer, Cham, Aug. 2018, pp. 303–327. DOI: [10.1007/978-3-319-96881-0_11](https://doi.org/10.1007/978-3-319-96881-0_11).
- [Mir06] Ilya Mironov. “Collision-Resistant No More: Hash-and-Sign Paradigm Revisited”. In: *PKC 2006*. Ed. by Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin. Vol. 3958. LNCS. Springer, Berlin, Heidelberg, Apr. 2006, pp. 140–156. DOI: [10.1007/11745853_10](https://doi.org/10.1007/11745853_10).
- [NY89] Moni Naor and Moti Yung. “Universal One-Way Hash Functions and their Cryptographic Applications”. In: *21st ACM STOC*. ACM Press, May 1989, pp. 33–43. DOI: [10.1145/73007.73011](https://doi.org/10.1145/73007.73011).
- [Oec03] Philippe Oechslin. “Making a Faster Cryptanalytic Time-Memory Trade-Off”. In: *CRYPTO 2003*. Ed. by Dan Boneh. Vol. 2729. LNCS. Springer, Berlin, Heidelberg, Aug. 2003, pp. 617–630. DOI: [10.1007/978-3-540-45146-4_36](https://doi.org/10.1007/978-3-540-45146-4_36).
- [Res18] E. Rescorla. *The Transport Layer Security (TLS) Protocol Version 1.3*. <https://tools.ietf.org/html/rfc8446>. Aug. 2018.
- [Rog06] Phillip Rogaway. “Formalizing Human Ignorance”. In: *Progress in Cryptology - VIETCRYPT 06*. Ed. by Phong Q. Nguyen. Vol. 4341. LNCS. Springer, Berlin, Heidelberg, Sept. 2006, pp. 211–228. DOI: [10.1007/11958239_14](https://doi.org/10.1007/11958239_14).
- [Rom90] John Rompel. “One-Way Functions are Necessary and Sufficient for Secure Signatures”. In: *22nd ACM STOC*. ACM Press, May 1990, pp. 387–394. DOI: [10.1145/100216.100269](https://doi.org/10.1145/100216.100269).
- [RTV04] Omer Reingold, Luca Trevisan, and Salil P. Vadhan. “Notions of Reducibility between Cryptographic Primitives”. In: *TCC 2004*. Ed. by Moni Naor. Vol. 2951. LNCS. Springer, Berlin, Heidelberg, Feb. 2004, pp. 1–20. DOI: [10.1007/978-3-540-24638-1_1](https://doi.org/10.1007/978-3-540-24638-1_1).

- [RV22] Ron D. Rothblum and Prashant Nalini Vasudevan. “Collision-Resistance from Multi-Collision-Resistance”. In: *CRYPTO 2022, Part III*. Ed. by Yevgeniy Dodis and Thomas Shrimpton. Vol. 13509. LNCS. Springer, Cham, Aug. 2022, pp. 503–529. DOI: [10.1007/978-3-031-15982-4_17](https://doi.org/10.1007/978-3-031-15982-4_17).
- [Sim98] Daniel R. Simon. “Finding Collisions on a One-Way Street: Can Secure Hash Functions Be Based on General Assumptions?” In: *EUROCRYPT’98*. Ed. by Kaisa Nyberg. Vol. 1403. LNCS. Springer, Berlin, Heidelberg, May 1998, pp. 334–345. DOI: [10.1007/BFb0054137](https://doi.org/10.1007/BFb0054137).
- [SS22] Michael Saks and Rahul Santhanam. “On Randomized Reductions to the Random Strings”. In: *37th Computational Complexity Conference (CCC 2022)*. Ed. by Shachar Lovett. Vol. 234. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022, 29:1–29:30. ISBN: 978-3-95977-241-9. DOI: [10.4230/LIPIcs.CCC.2022.29](https://doi.org/10.4230/LIPIcs.CCC.2022.29). URL: <https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.CCC.2022.29>.

Appendix A Proof of [Theorem 4.2](#) (Main)

In this section, we want to prove [Theorem 4.2](#), i.e., that there exists a set of oracles relative to which, no (existential) collision-resistant hash functions exist, but F is still distributional collision-resistant.

Theorem 4.2 (Main). *There exist oracles $\text{ColFind}, F$ s.t. for all polynomially bounded 1-ColFind-poly- $\mathcal{A}^{\text{F}, \text{ColFind}}$, for all polynomially honest, fixed query-length $h_{\text{slt}}^{\text{F}} : \{0, 1\}^\lambda \rightarrow \{0, 1\}^t$ both of the following hold for all sufficiently large $n \in \mathbb{N}$:*

(\exists dCRH) F is an unsalted $\frac{1}{4}$ -dCRH, i.e. $\text{SD}(\mathcal{A}^{\text{F}, \text{ColFind}}(1^n), \text{COL}_{\text{F}, n}) \geq \frac{1}{4}$

(no CRH) There is no salted CRH, i.e.,

$$\Pr_{\text{slt} \leftarrow \{0, 1\}^n} \left[\begin{array}{l} \text{ColFind}(h_{\text{slt}}) \rightarrow (w, w') : \\ w \neq w', h_{\text{slt}}(w) = h_{\text{slt}}(w') \end{array} \right] \geq \frac{1}{\lambda^2}$$

We have already proven *distributional* versions of this theorem – in [Lemma 4.4](#), we have shown that relative to a distribution \mathcal{D} of oracles, with high probability, no existential collision-resistant hash function exists, and in [Lemma 4.3](#), we prove that with high probability over \mathcal{D} , F is still a distributional collision-resistant hash function. However, to prove [Theorem 4.2](#), we need to show that there exists a fixed set of oracles for which both properties hold. We will use Borel-Cantelli ([Corollary 3.4](#)) and the Splitting Lemma [3.6](#) to obtain the main theorem.

Proof (Proof of [Theorem 4.2](#)). In order to prove [Theorem 4.2](#), we will not only prove that oracle $(F, \text{ColFind})$ exists, such that the two conditions (\exists dCRH) and (no CRH) are satisfied for *all* adversaries and *all* hash-functions, but actually, that the two conditions (\exists dCRH) and (no CRH) are satisfied *with probability 1* over sampling a *random* pair $(F, \text{ColFind})$ according to distribution \mathcal{D} .

We first prove that $(\exists \text{ dCRH})$ happens with probability 1 when sampling from \mathcal{D} and then prove that (no CRH) happens with probability 1 when sampling from \mathcal{D} . Since both events happen with probability 1, they also happen simultaneously with probability 1.

$(\exists \text{ dCRH})$ happens with probability 1 over \mathcal{D} . We need to prove that no polynomially bounded Turing machine \mathcal{A} can break the dCRH property of F . Lemma 4.3 shows us that for every adversary \mathcal{A} , for every large enough n , the probability of the adversary being successful is smaller than $\frac{1}{n^2}$. Therefore, Borel-Cantelli (Corollary 3.4) states that the probability that \mathcal{A} is successful for infinitely many security parameters is 0. Further, since Turing machines are countable, the probability that there exists an adversary that is successful for infinitely many security parameters is also 0.

(no CRH) happens with probability 1 over \mathcal{D} . We need to prove that no polynomially bounded and honest, fixed query-length compressing hash-function $h(\text{slt}, \cdot)$ is a CRH. Note that this case is a bit more complicated compared to the previous one: While h is still uniform and therefore countable, together with the salt, there are uncountably many combinations. However, it suffices to show that h is not collision-resistant for a *random* salt.

Let $(\text{slt}_\lambda)_{\lambda \in \mathbb{N}}$ denote a set of salts for h for each input length λ . Further, we denote by $g_{\text{slt}_\lambda} : \{0, 1\}^\lambda \rightarrow \{0, 1\}^t$, $g_{\text{slt}_\lambda}(x) = h(\text{slt}_\lambda, x)$ a circuit for length λ which hard-codes the salt slt_λ . Now, Lemma 4.4 gives us that for any such set of salts and any input lengths, the probability that ColFind does not produce a collision is at most $\frac{1}{\lambda^2}$:

$$\forall (\text{slt}_\lambda)_{\lambda \in \mathbb{N}} \forall \lambda : \Pr_{F, \text{HARD} \leftarrow \mathcal{S}\mathcal{D}} \left[\begin{array}{l} (w, w_{\text{coll}}) \leftarrow \text{ColFind}(g_{\text{slt}_\lambda}) \\ w \neq w_{\text{coll}} \wedge g_{\text{slt}_\lambda}(w) = g_{\text{slt}_\lambda}(w_{\text{coll}}) \end{array} \right] < \frac{1}{\lambda^2}$$

As this holds for all salts, it also holds for a random salt:

$$\forall \lambda : \Pr_{F, \text{HARD} \leftarrow \mathcal{S}\mathcal{D}, \text{slt}_\lambda} \left[\begin{array}{l} (w, w_{\text{coll}}) \leftarrow \text{ColFind}(g_{\text{slt}_\lambda}) \\ w \neq w_{\text{coll}} \wedge g_{\text{slt}_\lambda}(w) = g_{\text{slt}_\lambda}(w_{\text{coll}}) \end{array} \right] < \frac{1}{\lambda^2}$$

Next, as we want to fix F and HARD , while keeping slt random, we split the probability using the Splitting Lemma (Lemma 3.6): Set $X = (F, \text{HARD})$ and $Y = \text{slt}$, as well as $\varepsilon = 1 - \frac{1}{\lambda^2}$ and $\varepsilon' = 1 - \frac{2}{\lambda^2}$. Then, the Splitting Lemma gives us

$$\forall \lambda : \Pr_{F, \text{HARD} \leftarrow \mathcal{S}\mathcal{D}} \left[\Pr_{\text{slt}} \left[\begin{array}{l} (w, w_{\text{coll}}) \leftarrow \text{ColFind}(g_{\text{slt}}) \\ w \neq w_{\text{coll}} \wedge g_{\text{slt}}(w) = g_{\text{slt}}(w_{\text{coll}}) \end{array} \right] \geq \frac{1}{\lambda^2} \right] \geq 1 - \frac{2}{\lambda^2}$$

Let E_λ denote the event that the inner probability does not hold:

$$E_\lambda : \Pr_{\text{slt}} \left[\begin{array}{l} (w, w_{\text{coll}}) \leftarrow \text{ColFind}(g_{\text{slt}}) \\ w \neq w_{\text{coll}} \wedge g_{\text{slt}}(w) = g_{\text{slt}}(w_{\text{coll}}) \end{array} \right] < \frac{1}{\lambda^2}$$

Now, we get $\Pr_{\mathcal{D}}[E_\lambda] < \frac{2}{\lambda^2}$ for large enough λ . By Corollary 3.4, the probability that infinitely many E_λ happen is 0. Thus, with probability 1 ColFind breaks h with polynomial probability over the salt, except on finitely many λ , and thus, h is not a CRH. As h is uniform and therefore countable, with probability 1, relative to F and HARD , no h exists that is a CRH.

Appendix B *negl-dCRH* $\not\approx$ *OWF*

For a hash-function H , recall that $\text{COL}_{H,n}$ is the distribution (x, x') where $x \leftarrow_{\$} \{0, 1\}^n$ and $x' \leftarrow_{\$} H^{-1}(H(x))$ (cf. Fig. 1.2). Recall that a *negl-dCRH* H is a *dCRH* where no PPT adversary \mathcal{A} can induce a distribution such that the statistical distance between $\mathcal{A}(1^n)$ and $\text{COL}_{H,n}$ is negligible. Unrolling the definition of a negligible function, we obtain the following.

Definition B.1 (*negl-dCRH*). The function H is a (*n unsalted*) *negl-dCRH* if for all PPT \mathcal{A} there exists a polynomial q such that for all large enough $n \in \mathbb{N}$:

$$\text{SD}(\mathcal{A}(1^n), \text{COL}_{H,n}) \geq q(n)$$

That is, the definition of *negl-dCRH* is the same as that of *dCRH* except for the quantifier order. Namely, in *dCRH* definition the polynomial q is fixed *before* fixing the adversary, while in *negl-dCRH* every adversary must only have *some* polynomial, such that they are that polynomial -far from the distribution $\text{COL}_{H,n}$. Naturally, the latter is easier to satisfy: since the adversary's runtime is bounded by some polynomial, the runtime also gives a natural bound on how close the adversary can get to some hard distribution. Our proof relies exactly on this weakness of the adversary.

In order to show an oracle separation between *negl-dCRH* and a *OWF*, we will define an oracle H which implements a *negl-dCRH* as well as an oracle INV which inverts any *OWF* on a random input with good probability. Similar to the definition of F in Figure 4.5, we define H as a permutation with the last bit chopped off. Also similarly to the *dCRH* $\not\approx$ *CRH* separation, we define a hard set that INV avoids, however, now we can make the size of the hard set to depend on the adversary's runtime, instead of a fixed polynomial (since that is enough for *negl-dCRH* adversary) and that allows us rule out all relativizing proofs, instead of the limited class of relativizing proofs that we consider in the main theorem (*dCRH* $\not\approx$ *CRH*).

Similarly to our main separation result, we define an oracle distribution \mathcal{D} (cf. Figure 2.10) and prove that any polynomial-time computable function f^\square is inverted with probability $\frac{1}{2}$ by INV with high probability over \mathcal{D} .

Lemma B.2 ($\not\exists$ *OWF*). For all polynomially-time computable oracle functions f^\square there exists a constant c such that for infinitely many $\lambda \in \mathbb{N}$:

$$\Pr_{H, \text{INV} \leftarrow_{\$} \mathcal{D}} \left[\left| \left\{ \begin{array}{l} x \in \{0, 1\}^\lambda : \\ f^{\text{H}, \text{INV}}(\text{INV}(f^\square, f^{\text{H}, \text{INV}}(x), 1^\lambda, 1^{\lambda^c})) = f^{\text{H}, \text{INV}}(x) \end{array} \right\} \right| < \frac{1}{2} 2^\lambda \right] \leq \frac{1}{\lambda^2}$$

Moreover, we show that for any PPT adversary \mathcal{A} there exists a polynomial q , namely $q = |\mathcal{A}|^2$ (where $|\mathcal{A}|$ is the adversary's runtime), such that \mathcal{A} is unable to approximate uniform collisions better than with statistical distance $1/q(n)$, a non-negligible gap—with high probability over \mathcal{D} . Note that unlike Lemma 4.3 (F is *dCRH*), here we do *not* limit the adversary to be 1-ColFind-poly- F , but it can query all the oracles ($H, \text{INV}, \text{PSPACE}$) in any order and any polynomial number of times.

Lemma B.3 (\exists *negl-dCRH*). For all PPT oracle adversaries \mathcal{A}^\square there exists a polynomial q such that for all large enough security parameters $n \in \mathbb{N}$:

$$\Pr_{H, \text{INV} \leftarrow_{\$} \mathcal{D}} \left[\text{SD}(\mathcal{A}^{\text{H}, \text{INV}}(1^n), \text{COL}_{H,n}) < \frac{1}{q(n)} \right] \leq 1/n^2.$$

From Lemma B.2 and Lemma B.3, we then obtain the following theorem via a Borel-Cantelli argument, we omit this standard argument (it is analogous to the proof of Theorem 4.2).

Theorem B.4. *There exists an oracle \mathcal{O} relative to which there exists a negl-dCRH , but no OWF .*

We now first define and discuss the oracle distribution \mathcal{D} and then turn to the proofs of Lemma B.2 and Lemma B.3.

B.1 Oracle distribution

Recall that the main idea of \mathcal{D} is that INV inverts any polynomial-time computable OWF candidate f^\square , and H is a negl-dCRH . However, we need to be careful in our choice of INV : if INV yields a uniformly random pre-image for any function, we can use INV to find uniform collisions for H : We first sample x , then compute $y := \text{H}(x)$ and use INV to find a uniformly random pre-image of y .

Therefore, similarly to our $\text{dCRH} \not\approx \text{CRH}$ separation, we define a HARD set for H . Given HARD , we then define $\text{INV}(f^\square, y, 1^n)$ so that it returns only pre-images $x \in (f^{\text{H}, \text{INV}})^{-1}(y)$ which do not have queries in the HARD set on the path. However, now, we need to argue that INV still returns pre-images often enough. In fact, if f^\square makes too many queries, then it might almost always make a query in HARD , making INV useless, since it returns an error for most y . Therefore, we define an infinite sequence of hard sets

$$\text{HARD}_2 \supseteq \text{HARD}_4 \supseteq \text{HARD}_8 \dots$$

such that $|\text{HARD}_2| \approx 2 \cdot |\supseteq \text{HARD}_4|$ and $|\text{HARD}_4| \approx 2 \cdot |\supseteq \text{HARD}_8|$ etc.. Now, when querying INV , the input consists not only of $(f^\square, y, 1^n)$ (where the INV oracle seeks to give us $x \in \{0, 1\}^n$: $f^{\text{H}, \text{INV}}(x) = y$), but also a “budget” 1^m (with $m = 2^\ell$) specifying which hard set shall be used. The higher m , the smaller HARD_m and the more likely it is that INV returns an answer. However, the higher m , the more expensive it is also to specify the query, hence for every adversary attempting to break the negl-dCRH H there will be some small HARD set (whose size depends on the adversary’s runtime) that the adversary will never reach, thus we can show that H is a negl-dCRH .

However, this allows us to invert $f^{\text{H}, \text{INV}}$ on y , because the caller can choose m to be the polynomial $n^2 \cdot q_{f,n}$, where $q_{f,n}$ is an upper bound on the number of H -queries that f makes, and thus, if the

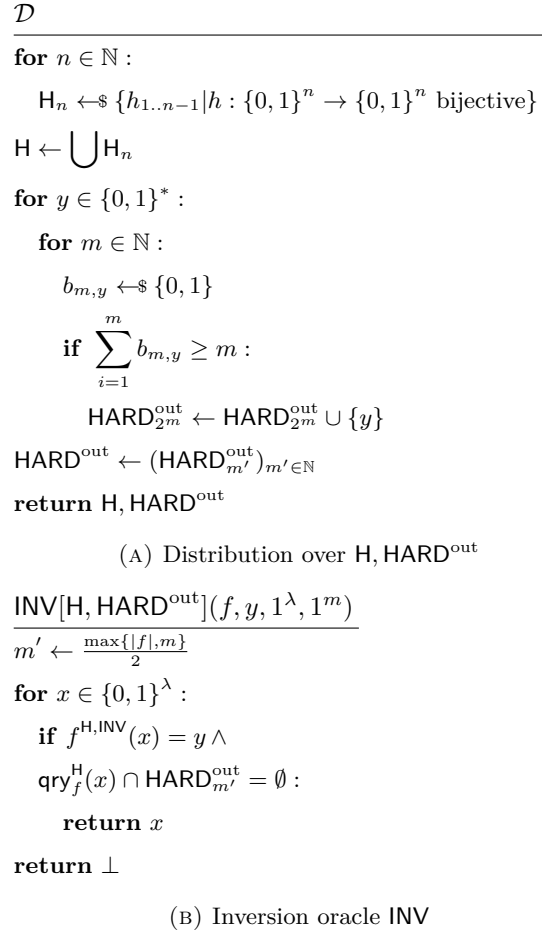


FIGURE 2.10 — Oracle (Distribution). We also give all algorithms access to a PSPACE oracle which we omit from notation for conciseness.

hard set is a smaller fraction than $\frac{1}{n^{2 \cdot q_f, n}}$ of the entire space, then in expectation, on a uniformly random input $f^{\text{H}, \text{INV}}(x)$ makes $\frac{1}{n^2}$ hard queries. By a Markov bound, INV inverts $f^{\text{H}, \text{INV}}$ successfully on all inputs except for a $\frac{1}{n^2}$ fraction. We describe the oracle distribution \mathcal{D} in Fig. 2.10 and then make the above arguments formal.

B.2 Proof of Lemma B.3 (Exists Negl-dCRH)

Now we prove Lemma B.3, restated here for convenience. The proof follows closely that of Lemma 4.3.

Lemma B.3 (\exists negl-dCRH). *For all PPT oracle adversaries \mathcal{A}^\square there exists a polynomial q such that for all large enough security parameters $n \in \mathbb{N}$:*

$$\Pr_{\text{H}, \text{INV} \leftarrow \mathcal{D}} \left[\text{SD}(\mathcal{A}^{\text{H}, \text{INV}}(1^n), \text{COL}_{\text{H}, n}) < \frac{1}{q(n)} \right] \leq 1/n^2.$$

Proof. Suppose for contradiction that there is an adversary \mathcal{A} , with polynomial runtime $|\mathcal{A}|$ s.t. for infinitely many n and for the polynomial $q = |\mathcal{A}|^2$:

$$\Pr_{\text{H}, \text{INV} \leftarrow \mathcal{D}} \left[\text{SD}(\mathcal{A}^{\text{H}, \text{INV}}(1^n), \text{COL}_{\text{H}, n}) < \frac{1}{q(n)} \right] > 1/n^2, \quad (2.22)$$

Now, similar to proof of Lemma 4.3, we use this adversary to compress H , see Figure 2.11 for compressor and decompressor algorithms (analogous to the ones in Figure 4.5). The main difference to the dCRH case is that unlike ColFind , the oracle INV might make recursive INV queries (since the input f might contain INV queries). However, each such recursive query will have m' that is at most half the m' in the higher level of the recursion (suppose f is given to INV as a circuit). Hence, the total number of H queries that one (recursive) INV query can make, is still upper bounded by a polynomial: there can be at most $\log m'$ many recursion layers and each layer can have at most m' many H queries on the path of f that is returned. Since $m' < |\mathcal{A}|$, simulating one INV query requires at most $|\mathcal{A}| \log |\mathcal{A}| < |\mathcal{A}|^2$ many H queries. Also, by definition of HARD , anything that is considered hard in the top level of the recursion, will also be considered hard in the deeper levels (in fact, even more queries are hard in the deeper levels), so we can be sure that INV never returns anything that has $\text{HARD}_{p, n}$, where $p = |\mathcal{A}|$, H_n -query on the path (where path consists of all the recursive queries too). Hence, we know that it is simulatable in the $\text{Decompress}^{\mathcal{A}}$.

By Lemma 6.6 (Hard Set is Large w.h.p)¹⁰, the hard set $\text{HARD}_{p, n}$ covers at least $\frac{1}{2^p}$ fraction of the output space with probability $> 1 - 2^{-n}$. Hence, we can fix the hard set to such a large set without altering the probability in (2.22) more than by an additive negligible fraction.

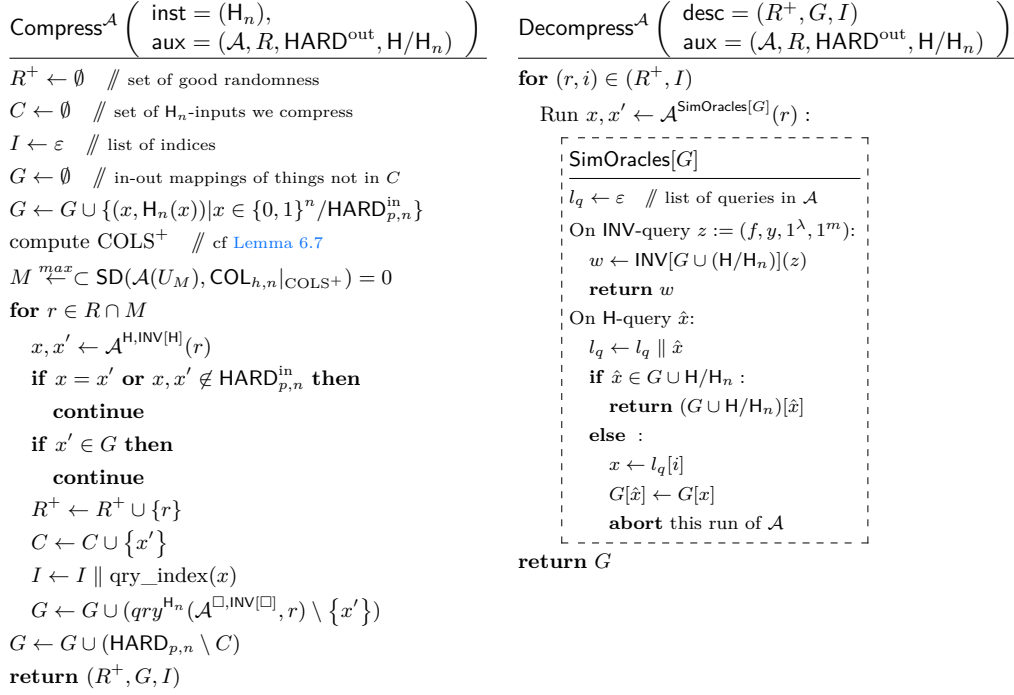
Now fix HARD , H/H_n , INV s.t. the probability in (2.22) is as large as possible, now for infinitely many n :

$$\Pr_{\text{H}_n \leftarrow \mathcal{D}} \left[\text{SD}(\mathcal{A}^{\text{H}, \text{INV}}(1^n), \text{COL}_{\text{H}, n}) < 1/q \right] > \frac{1}{n^2} - \text{negl}(\lambda),$$

That is, at least $\frac{1}{n^2} - \text{negl}(\lambda)$ fraction of all possible two-regular H_n are broken by \mathcal{A} . Let's call these $\mathcal{H}_{\text{broken}}$.

Now we pre-sample the randomness R for the adversary similarly to R in Section 6 and define $\text{Compress}^{\mathcal{A}}$, $\text{Decompress}^{\mathcal{A}}$ as in Figure 2.11, and with probability $1 - \text{negl}$ (by Lemma 6.8 (Many True

¹⁰note that Lemma 6.6 works also for the negl-dCRH definition of hard set $\text{HARD}_{p, n}$, when p, n are fixed, since in that case the definitions of $\text{HARD}_{p, n}$ coincide.


 FIGURE 2.11 — Compression Algorithms. Here $p := |\mathcal{A}|$.

Hard Collisions in $\mathcal{A}(R \cap M)$ ¹¹) that randomness R is such that $|\mathbf{H}_n(\mathcal{A}(R \cap M)) \cap \text{HARD}_{p,n}^{\text{cl}}| \geq \frac{1}{\text{ply}} 2^{n-1}$, where $\frac{1}{\text{ply}} = \frac{1}{40p(n)} - \frac{1}{40q(n)} = \frac{1}{40|\mathcal{A}|(n)} - \frac{1}{40|\mathcal{A}|^2(n)}$. Now with that randomness, the $\text{Compress}^{\mathcal{A}}$ has output length at most

$$2^n(n - \log e) - \frac{1}{2\text{com}} 2^n n \quad (2.23)$$

by Lemma 6.3 (Output length of $\text{Compress}^{\mathcal{A}}$ is short)¹². Note that $|\mathcal{A}|$, the number queries that the adversary makes, is polynomial by assumption and additionally com is defined as $|R^+| = 2^n \frac{1}{\text{com}}$, i.e. the fraction of inputs to \mathbf{F}_n that we compress, so by Lemma 6.9 (If $|\mathcal{A}(R \cap M)|$ big, then $\text{Compress}^{\mathcal{A}}$ compresses by a lot)¹³ $\text{com} = \text{ply}|\mathcal{A}|$ which is also a polynomial. That is, all functions in $\mathcal{H}_{\text{broken}}$ have a short encoding.

However, $\mathcal{H}_{\text{broken}}$ covers a non-negligible fraction of 2-regular functions, while Lemma 6.10 says that only neglig. fraction can have short description. The exact calculation to get the contradiction is exactly the same as in proof of Lemma 4.3.

¹¹This lemma only depends on the adversary's success probability, and since \mathbf{H} is defined similarly to \mathbf{F} , the same lemma applies here, when we just rename \mathbf{F} to \mathbf{H} .

¹²Since the $\text{Compress}^{\mathcal{A}}$ here outputs analogous parameters compared to the $\text{Compress}^{\mathcal{A}}$ in Section 6, this lemma gives also the encoding length of our neglig-dCRH compressor $\text{Compress}^{\mathcal{A}}$.

¹³This lemma also works in our case, since it only counts what is the maximum number of hard \mathbf{F} -queries (\mathbf{H} -queries in our case) that are added to G at each iteration. Since INV does not have hard queries in the path, this number is upper bounded by $|\mathcal{A}|$, exactly as in Lemma 6.9 proof.

B.3 Proof of Lemma B.2 (No OWF)

In this section we prove that for all polynomial-time computable oracle functions f^\square there exists a constant c such that for infinitely many $\lambda \in \mathbb{N}$:

$$\Pr_{\mathbf{H}, \text{INV} \leftarrow \mathcal{D}} \left[\left| \left\{ \begin{array}{c} x \in \{0, 1\}^\lambda : \\ \text{INV}(f^\square, f^{\mathbf{H}, \text{INV}}(x), 1^\lambda, 1^{\lambda^c}) = f^{-1}(f^{\mathbf{H}, \text{INV}}(x)) \end{array} \right\} \right| < \frac{1}{2} 2^\lambda \right] \leq \frac{1}{\lambda^2}$$

Lemma B.5 (Markov's Inequality). *Let X be a non-negative random variable and $a > 0$. Now*

$$\Pr[X \geq a] \leq \frac{\mathbb{E}[X]}{a}$$

Proof (Lemma B.2). Let f^\square be an arbitrary, but fixed polynomial-time computable function and denote by $q(\lambda)$ an upper bound on the number of direct and indirect (through INV) H-queries that $f^{\mathbf{H}, \text{INV}}(x)$ makes on inputs $x \in \{0, 1\}^\lambda$. Let c be an arbitrary but fixed constant such that $\lambda^c > 2\lambda^2 q(\lambda)$ (and λ^c a power of 2 for convenience). We call x *bad* if there happens at least one query in HARD_{λ^c} on the path of $f^{\mathbf{H}, \text{INV}}(x)$. Assuming w.l.o.g. that f only makes *distinct* queries, we have that for all $x \in \{0, 1\}^\lambda$ and for all H, the expected number of hard queries in the path is

$$\mathbb{E}_{\mathbf{H}, \text{HARD}_{\lambda^c}} [|\text{qry}_f^{\mathbf{H}}(x) \cap \text{HARD}_{\lambda^c}|] \leq q(\lambda) \frac{1}{2\lambda^2 q(\lambda)} = \frac{1}{2\lambda^2}$$

We now use Markov's inequality (Lemma B.5) to bound the probability of x being bad:

$$\begin{aligned} \Pr_{\mathbf{H}, \text{HARD}_{\lambda^c}} [x \text{ is bad}] &= \Pr_{\mathbf{H}, \text{HARD}_{\lambda^c}} [|\text{qry}_f^{\mathbf{H}}(x) \cap \text{HARD}_{\lambda^c}| \geq 1] \\ &\leq \frac{\mathbb{E}_{\mathbf{H}, \text{HARD}_{\lambda^c}} [|\text{qry}_f^{\mathbf{H}}(x) \cap \text{HARD}_{\lambda^c}|]}{1} \\ &\leq \frac{1}{2\lambda^2} \end{aligned}$$

Similarly, we can use Markov again to bound the number of x which are bad as

$$\begin{aligned} \Pr_{\mathbf{H}, \text{INV} \leftarrow \mathcal{D}} [|\{x \in \{0, 1\}^\lambda : x \text{ is bad}\}| \geq \frac{1}{2} 2^\lambda] \\ &\leq \frac{1}{\frac{1}{2} 2^\lambda} \mathbb{E}_{\mathbf{H}, \text{INV} \leftarrow \mathcal{D}} [|\{x \in \{0, 1\}^\lambda : x \text{ is bad}\}|] \\ &\leq \frac{1}{\lambda^2} \end{aligned}$$

As the bad x are exactly those for which INV does not return a preimage, we get our desired result:

$$\Pr_{\mathbf{H}, \text{INV} \leftarrow \mathcal{D}} \left[\left| \left\{ \begin{array}{c} x \in \{0, 1\}^\lambda : \\ \text{INV}(f^\square, f^{\mathbf{H}, \text{INV}}(x), 1^\lambda, 1^{\lambda^c}) = f^{-1}(f^{\mathbf{H}, \text{INV}}(x)) \end{array} \right\} \right| < \frac{1}{2} 2^\lambda \right] \leq \frac{1}{\lambda^2}$$

Appendix C Unsalted and Salted dCRH Definitions

In the introduction, we defined dCRH in its unsalted variant:

Definition 1.2 (Unsalted dCRH). We say that a polynomial-time computable function h with $\forall x \in \{0, 1\}^n : |h(x)| \leq n - 1$ is an (*unsalted*) *distributional CRH* if there exists a polynomial $p(n)$ such that for all PPT \mathcal{A} :

$$\text{SD}(\mathcal{A}(1^n), \text{COL}_{h,n}) \geq \frac{1}{p(n)},$$

where Fig. 1.2 defines $\text{COL}_{h,n}$.

As we mentioned, a nice property of dCRH is that its unsalted variant is already secure against non-uniform adversaries. However, one can of course also define a salted variant of dCRH:

Definition C.1 (Salted dCRH). We say that a polynomial-time computable function h with $\forall x \in \{0, 1\}^n : |h(x)| = n - 1$ is a (salted) *distributional CRH* if there exists a polynomial $p(n)$ and a negligible function ϵ such that for all PPT \mathcal{A} and all large n :

$$\Pr_{\text{slt} \leftarrow \mathbb{S}\{0,1\}^n} \left[\text{SD}(\mathcal{A}(\text{slt}), \text{COL}_{h_{\text{slt}}, n}) \geq \frac{1}{p(n)} \right] \geq 1 - \epsilon(n)$$

Lemma C.2 (Unsalted dCRH implies Salted dCRH). *If h is an unsalted dCRH (Definition 1.2), then $h_{\text{slt}} := h$ is a salted dCRH (Definition C.1).*

Proof. Let h be an unsalted dCRH. Now for some polynomial $p(n)$ and for all PPT \mathcal{A} we have

$$\text{SD}(\mathcal{A}(1^n), \text{COL}_{h, n}) \geq \frac{1}{p(n)}.$$

Let \mathcal{A} be any PPT adversary, and let ϵ be any negligible functions. Now for all large enough n :

$$\begin{aligned} & \Pr_{\text{slt} \leftarrow \mathbb{S}\{0,1\}^n} \left[\text{SD}(\mathcal{A}(1^n), \text{COL}_{h_{\text{slt}}, n}) \geq \frac{1}{p(n)} \right] \\ & \geq \Pr_{\text{slt} \leftarrow \mathbb{S}\{0,1\}^n} \left[\text{SD}(\mathcal{A}(1^n), \text{COL}_{h, n}) \geq \frac{1}{p(n)} \right] = 1 \geq 1 - \epsilon(n) \end{aligned}$$

Appendix D Counterexample: F-queries first

Consider the adversary in Figure 4.12: Assume that y_1 or y_2 is in HARD and let \bar{x}_1 and \bar{x}_2 be the F-collisions on y_1 and y_2 respectively. Then, this adversary can return collisions in HARD which we cannot decompress: The query index will tell us whether the collision included x_1 or x_2 and therefore also the correct y value. However, any value w that we did not yet recover (i.e., it is not yet in G) could have originally mapped to y . Consequently, there is no way for the decompression routine to correctly emulate ColFind and \mathcal{A} will query F on whatever value the decompressor decides to return from the simulation.

```

 $\mathcal{A}^{\text{F}, \text{ColFind}}(1^\lambda)$ 
-----
 $x_1, x_2 \leftarrow \mathbb{S}\{0, 1\}^n \times \{0, 1\}^n$ 
 $y_1 \leftarrow \text{F}(x_1)$ 
 $y_2 \leftarrow \text{F}(x_2)$ 
 $w_1, w_2 \leftarrow \text{ColFind}(h^{\text{F}})$ 
    [-----]
     $h^{\text{F}}(w)$ 
    -----
     $z \leftarrow \text{F}(w)$ 
    if  $z = y_1 \vee z = y_2$ 
        return  $0^{|z|}$ 
    return  $z$ 
    [-----]
if  $\text{F}(w_1) = y_1$  return  $x_1, w_1$ 
if  $\text{F}(w_1) = y_2$  return  $x_2, w_1$ 
if  $\text{F}(w_2) = y_1$  return  $x_1, w_2$ 
if  $\text{F}(w_2) = y_2$  return  $x_2, w_2$ 
return  $w_1, w_2$ 
    
```

FIGURE 4.12 — Example Adversary

Searching for ELFs in the Cryptographic Forest

Marc Fischlin

Felix Rohrbach

Cryptoplexity, Technische Universität Darmstadt, Germany

www.cryptoplexity.de

{[marc.fischlin](mailto:marc.fischlin@cryptoplexity.de), [felix.rohrbach](mailto:felix.rohrbach@cryptoplexity.de)}@cryptoplexity.de

Abstract

Extremely Lossy Functions (ELFs) are families of functions that, depending on the choice during key generation, either operate in injective mode or instead have only a polynomial image size. The choice of the mode is indistinguishable to an outsider. ELFs were introduced by Zhandry (Crypto 2016) and have been shown to be very useful in replacing random oracles in a number of applications.

One open question is to determine the minimal assumption needed to instantiate ELFs. While all constructions of ELFs depend on some form of exponentially-secure public-key primitive, it was conjectured that exponentially-secure secret-key primitives, such as one-way functions, hash functions or one-way product functions, might be sufficient to build ELFs. In this work we answer this conjecture mostly negative: We show that no primitive, which can be derived from a random oracle (which includes all secret-key primitives mentioned above), is enough to construct even moderately lossy functions in a black-box manner. However, we also show that (extremely) lossy functions themselves do not imply public-key cryptography, leaving open the option to build ELFs from some intermediate primitive between the classical categories of secret-key and public-key cryptography.

1 Introduction

Extremely lossy functions, or short ELFs, are collections of functions that support two modes: the injective mode, in which each image has exactly one preimage, and the lossy mode, in which the function merely has a polynomial image size. The mode is defined by a seed or public key pk which parameterizes the function. The key pk itself should not reveal whether it describes the injective mode or the lossy mode. In case the lossy mode does not result in a polynomially-sized image, but the function compresses by at least a factor of 2, we will speak of a (moderately) lossy function (LF).

Extremely lossy functions were introduced by Zhandry [Zha16; Zha19] to replace the use of the random oracle model in some cases. The random oracle model (ROM) [BR93] introduces a truly random function to which all parties have access to. This random function turned out to be useful in modeling hash functions for security proofs of real-world protocols. However, such a truly random function clearly does not exist in reality and it has been shown that no hash function can replace such

an oracle without some protocols becoming insecure [CGH98]. Therefore, a long line of research aims to replace the random oracle by different modeling of hash functions, e.g., by the notion of correlation intractability [CGH98] or by Universal Computational Extractors (UCEs) [BHK13]. However, all these attempts seem to have their own problems: Current constructions of correlation intractability require extremely strong assumptions [CCR16], while for UCEs, it is not quite clear which versions are instantiable [BFM14; BST16]. Extremely lossy functions, in turn, can be built from relatively standard assumptions.

Indeed, it turns out that extremely lossy functions are useful in removing the need for a random oracle in many applications: Zhandry shows it can be used to generically boost selective security to adaptive security in signatures and identity-based encryption, construct a hash function which is output intractable, point obfuscation in the presence of auxiliary information and many more [Zha16; Zha19]. Agrikola, Couteau and Hofheinz [ACH20] show that ELF_s can be used to construct probabilistic indistinguishability obfuscation from only polynomially-secure indistinguishability obfuscation. In 2022, Murphy, O’Neill and Zaheri [MOZ22] used ELF_s to give full instantiations of the OAEP and Fujisaki-Okamoto transforms. Recently, Brzuska et al. [BCEKM23] improve on the instantiation of the Fujisaki-Okamoto transform and instantiate the hash-then-evaluate paradigm for pseudorandom functions using ELF_s.

While maybe not as popular as their extreme counterpart, moderately lossy functions have their own applications as well: Braverman, Hassidim and Kalai [BHK11] build leakage-resistant pseudo-entropy functions from lossy functions, and Dodis, Vaikuntanathan and Wichs [DVW20] use lossy functions to construct extractor-dependent extractors with auxiliary information.

1.1 Our Contributions

One important open question for extremely lossy functions, as well as for moderately lossy functions, is the minimal assumption to build them. The constructions presented by Zhandry are based on the exponential security of the decisional Diffie-Hellman problem, but he conjectures that public-key cryptography should not be necessary and suggests for future work to try to construct ELF_s from exponentially-secure symmetric primitives (As Zhandry shows as well in his work, polynomial security assumptions are unlikely to be enough for ELF_s¹). Holmgren and Lombardi [HL18] wondered whether their definition of one-way product functions might suffice to construct ELF_s.

For moderately lossy functions, the picture is quite similar: While all current constructions require (polynomially-secure) public-key cryptography, it is generally assumed that public-key cryptography should not be necessary for them and that private-key assumptions should suffice (see, e.g., [QWW21]).

In this work, we answer the questions about building (extremely) lossy functions from symmetric-key primitive mostly negative: There exists no fully-black box construction of extremely lossy functions, or even moderately lossy functions, from a large number of primitives, including exponentially-secure one-way functions, exponentially-secure collision resistant hash functions or one-way product functions. Indeed, any primitive that exists unconditionally relative to a random oracle is not enough. We will call this family of primitives *Oraclecrypt*, in reference to the famous naming convention by

¹ELF_s can be distinguished efficiently using a super-logarithmic amount of non-determinism. It is consistent with our knowledge, however, that NP with a super-logarithmic amount of non-determinism is solvable in polynomial time while polynomially-secure cryptographic primitives exist. Any construction of ELF_s from polynomially-secure cryptographic primitives would therefore change our understanding of NP-hardness.

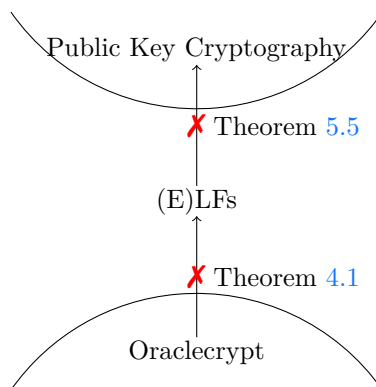


FIGURE 1.1 — We show both an oracle separation between Oraclecrypt and (E)LFs as well as (E)LFs and key agreement.

Impagliazzo [Imp95], in which Minicrypt refers to the family of primitives that can be built from one-way functions in a black-box way.

Note that most of the previous reductions and impossibility results, such as the renowned result about the impossibility of building key exchange protocols from black-box one-wayness [IR89], are in fact already cast in the Oraclecrypt world. We only use this term to emphasize that we also rule out primitives that are usually not included in Minicrypt, like collision resistant hash functions [Sim98].

On the other hand, we show that public-key primitives might not strictly be needed to construct ELFs or moderately lossy functions. Specifically, we show that no fully black-box construction of key agreement is possible from (moderately) lossy functions, and extend this result to prevent any fully black-box construction even from extremely lossy functions (for a slightly weaker setting, though). This puts the primitives lossy functions and extremely lossy functions into the intermediate area between the two classes Oraclecrypt and Public-Key Cryptography.

Finally, we discuss the relationship of lossy functions to hard-on-average problems in SZK, the class of problems that have a statistical zero-knowledge proof. We see hard-on-average SZK as a promising minimal assumption to build lossy functions from – indeed, it is already known that hard-on-average SZK problems follow from lossy functions with sufficient lossiness. While we leave open the question of building such a construction for future work, we give a lower bound for hard-on-average SZK problems that might be of independent interest, showing that hard-on-average SZK problems cannot be built from any Oraclecrypt primitive in a fully black-box way. While this is already known for some primitives in Oraclecrypt [BD19], these results do not generalize to all Oraclecrypt primitives as our proof does.

Note that all our impossibility results only rule out black-box constructions, leaving the possibility of future non-black-box constructions. However, while there is a growing number of non-black-box constructions in the area of cryptography, the overwhelming majority of constructions are still black-box constructions. Further, as all mentioned primitives like exponentially-secure one-way functions, extremely lossy functions or key agreement might exist unconditionally, ruling out black-box constructions is the best we can hope for to show that a construction probably does not exist.

1.2 Our Techniques

Our separation of Oraclecrypt primitives and extremely/moderately lossy functions is based on the famous oracle separation by Impagliazzo and Rudich [IR89]: We first introduce a strong oracle that makes sure no complexity-based cryptography exists unconditionally, and then add an independent random oracle that allows for specific cryptographic primitives (specifically, all Oraclecrypt primitives) to exist again. We then show that relative to these oracles, (extremely) lossy functions do not exist by constructing a distinguisher between the injective and lossy mode for any candidate construction. A key ingredient here is that we can identify the *heavy queries* in a lossy function with high probability with just polynomially many queries to the random oracle, a common technique used for example in the work by Bitansky and Degwekar [BD19]. Finally, we use the two-oracle technique by Hsiao and Reyzin [HR04] to fix a set of oracles. We note that our proof technique is similar to a technique in the work by Pietrzak, Rosen and Segev to show that the lossiness of lossy functions cannot be increased well in a black-box way [PRS12]. Our separation result for SZK, showing that primitives in Oraclecrypt may not suffice to derive hard problems in SZK, follows a similar line of reasoning.

Our separation between lossy functions and key agreement is once more based on the work by Impagliazzo and Rudich [IR89], but this time using their specific result for key agreement protocols. Similar to the techniques in [GHMM18], we try to compile out the lossy function to be then able to apply the Impagliazzo-Rudich adversary: We first show that one can build (extremely) lossy function oracles relative to a random oracle (where the lossy function itself is efficiently computable via oracle calls, but internally makes an exponentially number of random oracle evaluations). The heart of our separation is then a simulation lemma showing that any efficient game relative to our (extremely) lossy function oracle can be simulated efficiently and sufficiently close given only access to a random oracle. Here, sufficiently close means an inverse polynomial gap between the two cases but where the polynomial can be set arbitrarily. Given this we can apply the key agreement separation result of Impagliazzo and Rudich [IR89], with a careful argument that the simulation gap does not infringe with their separation.

1.3 Related Work

Lossy Trapdoor Functions Lossy trapdoor functions were defined by Peikert and Waters in [PW08; PW11] who exclusively considered such functions to have a trapdoor in injective mode. Whenever we talk about lossy functions in this work, we refer to the moderate version of extremely lossy functions which does not necessarily have a trapdoor. The term extremely lossy function (ELFs) is used as before to capture strongly compressing lossy functions, once more without requiring a trapdoor for the injective case.

Targeted Lossy Functions Targeted lossy functions were introduced by Quach, Waters and Wichs [QWW21] and are a relaxed version of lossy functions in which the lossiness only applies to a small set of specified inputs. The motivation of the authors is the lack of progress in creating lossy functions from other assumptions than public-key cryptography. Targeted lossy functions, however, can be built from Minicrypt assumptions, and, as the authors show, already suffices for many applications, such as construct extractor-dependent extractors with auxiliary information and pseudo-entropy functions. Our work very much supports this line of research, as it shows that any further progress in creating lossy functions from Minicrypt/Oraclecrypt assumptions is

unlikely (barring some construction using non-black-box techniques) and underlines the need of such a relaxation for lossy functions, if one wants to build them from Minicrypt assumptions.

Amplification of Lossy Functions Pietrzak, Rosen and Segev [PRS12] show that it is impossible to improve the relative lossiness of a lossy function in a black-box way by more than a logarithmic amount. This translates into another obstacle in building ELFs, even when having access to a moderately lossy function. Note that this result strengthens our result, as we show that even moderately lossy functions cannot be built from anything in Oraclecrypt.

2 Technical Overview

In this chapter, we will give an overview about our two main theorems of this paper and techniques used to prove them. The (extremely) lossy functions consists of two algorithms, $\text{Gen}(1^\lambda, \text{mode})$ for generating a public key in the input mode $\text{mode} = \text{loss}$ or $\text{mode} = \text{inj}$, and $\text{Eval}(\text{pk}, \cdot)$ taking a public key and some input of size $\text{in}(\lambda)$ and outputting a value. Here, the function Eval is (extremely) lossy if the key pk has been generated in lossy mode, and injective if the key has been generated in injective mode. In most of our results we give both algorithms also access to one or more oracles.

2.1 No (E)LFs in Oraclecrypt

Our first results says that one cannot build lossy functions, and thus neither extremely lossy functions, from any primitive that exists (unconditionally) relative to a random oracle:

Theorem 4.1 (informal). *There exists no fully black-box construction of lossy functions from any Oraclecrypt primitive.*

Our proof for this Theorem follows a proof idea by Pietrzak, Rosen and Segev [PRS12], which they used to show that lossy functions cannot be amplified well, i.e., one cannot build a lossy function which is very compressing in the lossy mode from a lossy function that is only slightly compressing in the lossy mode. We adapt the idea to show that lossy functions cannot be built unconditionally from a random oracle. Note that some technical details of our construction differ from [PRS12], though. For example, we show the result relative to a modified PSPACE oracle instead of an EXPTIME oracle.

We will now explain how the proof for our result works. First note that our result only holds for fully black-box constructions. The reason for this is that we use the two-oracle technique by Hsiao and Reyzin [HR04]. This approach considers a “constructive” oracle which allows to build the primitive in question, and a “breaking” oracle which allows to break any construction of the other primitive which is only based on the first oracle. Presenting such oracles provides a fully black-box separation of the two primitives.

In our case, the “constructive” oracle \mathcal{O} in the two-oracle technique is a random oracle, supporting the implementation of any Oraclecrypt primitive. For the “breaking” oracle, we introduce a PSPACE-oracle with a twist: The PSPACE-oracle will itself have oracle access to another, independent random oracle \mathcal{O}' . The reason for this extra random oracle will become apparent soon. For now we remark that this independent random oracle \mathcal{O}' does not invalidate the security of the Oraclecrypt primitives relative to \mathcal{O} .

The main part of our proof consists of showing that we can use the breaking oracle $\text{PSPACE}^{\mathcal{O}'}$ to distinguish $\text{Eval}^{\mathcal{O}}(\text{pk}, \cdot)$ for injective public key pk from $\text{Eval}^{\mathcal{O}}(\text{pk}, \cdot)$ for lossy public key pk . The approach is now to approximate the image size of $\text{Eval}^{\mathcal{O}}(\text{pk}, \cdot)$ by the image size of $\text{Eval}^{\mathcal{O}'}(\text{pk}, \cdot)$. If we can indeed approximate this sufficiently close, then we can use the $\text{PSPACE}^{\mathcal{O}'}$ -oracle to give an estimate for the image size $\text{Eval}^{\mathcal{O}'}(\text{pk}, \cdot)$ for the same oracle \mathcal{O}' to get the answer for the original function $\text{Eval}^{\mathcal{O}}(\text{pk}, \cdot)$. However, the evaluation algorithm for oracle \mathcal{O} and for oracle \mathcal{O}' may be quite far apart. To get a sufficiently close approximation we need to “modify” \mathcal{O}' to at least coincide on the more likely queries of the evaluation algorithm to \mathcal{O} . This is accomplished based on the *heavy queries* technique of Bitansky et al. [BDV17; BD19]:

Definition 4.3 (heavy queries, informal). *We call a query q to \mathcal{O} heavy for given key pk and oracle \mathcal{O} if, for a large fraction of $x \in \{0, 1\}^{\text{in}(\lambda)}$, the evaluation $\text{Eval}^{\mathcal{O}}(\text{pk}, x)$ queries \mathcal{O} about q at some point. We denote by Q_H the set of all heavy queries (for pk, \mathcal{O}).*

We note that we actually need to also take into account the queries of the key generating algorithm but omit this here in the overview. Remarkably, determining a superset of all heavy queries is easy:

Lemma 4.4 (informal). *We can compute in probabilistic polynomial-time (in λ) a set \hat{Q}_H which contains all heavy queries of $\text{Eval}^{\mathcal{O}}(\text{pk}, \cdot)$ for pk, \mathcal{O} with overwhelming probability.*

We simply run $\text{Eval}^{\mathcal{O}}(\text{pk}, \cdot)$ for a sufficiently high, but still polynomial number of random inputs $x \in \{0, 1\}^{\text{in}(\lambda)}$, recording all random oracle queries to get all heavy queries with probability at least $1 - 2^{-\text{in}(\lambda)}$. The next step is to switch from oracle \mathcal{O} to oracle \mathcal{O}' which is available to the attacker via the $\text{PSPACE}^{\mathcal{O}'}$ oracle. That is, we consider the evaluation function $\text{Eval}^{\mathcal{O}'}(\text{pk}, \cdot)$ for oracle \mathcal{O}' instead of $\text{Eval}^{\mathcal{O}}(\text{pk}, \cdot)$. In order to be still close to the original evaluation function for oracle \mathcal{O} , we let \mathcal{O}' agree on the (computable set of) heavy queries with \mathcal{O} . Hence, we formally consider the oracle \mathcal{O}'_H that is identical to \mathcal{O} on all heavy queries.

The next lemma now states that ignoring the non-heavy queries in oracle $\mathcal{R} := \mathcal{O}'_H$ is inconsequential. More precisely, the lemma says that if we would also enforce consistency with \mathcal{O} on non-heavy queries, denoted as oracle \mathcal{R}' , this would not change the size of the evaluation function noticeably:

Lemma 4.5 (informal). *For any oracle \mathcal{R}' that is identical to \mathcal{R} everywhere except for the non-heavy queries Q_G^{nonh} by key generation, i.e., $\mathcal{R}(q) = \mathcal{R}'(q)$ for any $q \notin Q_G^{\text{nonh}}$, the image sizes of $\text{Eval}^{\mathcal{R}}(\text{pk}, \cdot)$ and $\text{Eval}^{\mathcal{R}'}(\text{pk}, \cdot)$ differ by at most $\frac{2^{\text{in}(\lambda)}}{10}$.*

We next show that we can use an oracle \mathcal{O}'_H to distinguish lossy from injective keys: We know that for lossy keys, the image size of $\text{Eval}^{\mathcal{O}}(\text{pk}, x)$ over all inputs x should be at most $\frac{1}{2} \cdot 2^{\text{in}(\lambda)}$, while for injective keys, the image size should be $2^{\text{in}(\lambda)}$. As switching the oracle from \mathcal{O} to \mathcal{O}'_H only changes the image size slightly, there is still a large gap between the image size for injective keys and the image size for lossy keys. Therefore, if we can calculate the image size of $\text{Eval}^{\mathcal{O}'_H}(\text{pk}, \cdot)$ relative to \mathcal{O}'_H , we can decide between lossy keys and injective keys.

Now the reason why we need an augmented PSPACE oracle also becomes apparent. In general, we cannot compute the image size of a function relative to a randomly sampled function \mathcal{O}' in PSPACE . As we have to compute the function for every input x to calculate the image size, a lazily sampled \mathcal{O}' might be asked on exponentially many input queries, which we would have to save to answer consistently, which clearly is not in PSPACE anymore. To work around this problem, we add a randomly sampled \mathcal{O}' to which the PSPACE oracle has full access. Now we only have to modify

\mathcal{O}' on the polynomially many heavy queries and then can calculate the image size of any function relative to \mathcal{O}'_H .

Lemma 4.6 (informal). *Given a superset of the heavy queries $\hat{Q}_H \supseteq Q_H$, we can decide correctly whether $\text{Eval}^{\mathcal{O}}(\text{pk}, \cdot)$ is lossy or injective with overwhelming probability.*

This lemma suffices to prove the theorem's statement.

2.2 No Key Agreement from (E)LFs

Our first result shows that lossy functions (and therefore also ELF's) are not part of Oraclecrypt, a set which contains symmetric primitives and also, for example, collision-resistant hash functions. However, this does not mean that ELF's inherently require public-key cryptography to build them. To show this, our second main theorem shows that lossy functions do not imply key agreement in a fully black-box way. In Chapter 5, we also extend this result to ELF's, but in this overview, we will only focus on lossy functions.

Theorem 5.5 (informal). *There exists no fully black-box construction of a secure key agreement protocol from lossy functions.*

Our result relies on the famous result by Impagliazzo and Rudich [IR89], showing that no key agreement can be built in a black-box way from one-way functions. More concretely, they construct an adversary that, relative to a random oracle and, e.g., a PSPACE oracle, any candidate key agreement protocol can be broken. We would like to deploy this adversary, but we cannot use it directly: We would need to show that the adversary also works relative to an oracle that allows for lossy functions to exist, which a random oracle does not (as we have seen in the previous result). To explain why this makes the problem more challenging than in the case of separating key agreement from a random oracle, consider the following possibility to use lossy functions in interactive protocols: Assume that Alice picks a secret bit b randomly and, depending on the value of b , either generates an injective or a lossy key pk , and sends this public key over to Bob. For Bob, as well as for an outsider attacker, it is infeasible to determine if pk is lossy or not. However, Bob is now able to perform computations *implicitly depending on Alice's bit b* via the $\text{Eval}(\text{pk}, \cdot)$ algorithm. Potentially, Bob also returns information about the outcomes of these evaluations back to Alice. In other words, a lossy function already implements a very skewed form of secret bit transfer in which the recipient only has implicit access to the transferred bit. We have to show that this still does not facilitate the task of designing a full-fledged key exchange protocol.

Therefore, we start by defining an oracle that implements a lossy function. The oracle consists of two functions $\text{Gen}^{\Gamma, \Pi}(1^\lambda, \cdot)$ and $\text{Eval}^{\Gamma, \Pi}(\text{pk}, x)$, where the first function produces a public key either in injective or in lossy mode, and the second function lets us evaluate the lossy function for public key pk on input x . The oracle makes use of two random permutations Γ and Π , where Γ is considered an integral part of the oracle and cannot be accessed directly, while Π can be accessed directly. Note that our lossy function oracle is, by definition, efficient when queried as an oracle, but is internally inefficient, i.e., it makes exponentially many queries to Γ internally. Further, we define the oracle in such a way that it has another useful property for us: Every random public key, i.e., a key not generated by $\text{Gen}^{\Gamma, \Pi}$, is a valid public key in injective mode with overwhelming probability.

We could now try to modify the adversary of Impagliazzo-Rudich to our new oracle. Instead, the heart of our result is the Simulation Lemma which shows, for any security game Game , access to

the lossy function oracle can be simulated sufficiently close by an efficient algorithm (we call *Wrap*) that only has access to the random permutation Π . The quality of the simulation is not negligibly close, but instead determined by an inverse polynomial $\alpha(\lambda)$ and yields a statistical distance of at most $\alpha(\lambda)$.

Lemma 5.2 (Simulation Lemma, informal). *For any polynomial $p(\lambda)$ and any inverse polynomial $\alpha(\lambda)$ there exists an efficient algorithm *Wrap* such that for any efficient algorithm \mathcal{A} , any efficient experiment *Game* making at most $p(\lambda)$ calls to the oracle, the statistical distance between $\text{Game}^{\mathcal{A}, (\text{Gen}^{\Gamma, \Pi}, \text{Eval}^{\Gamma, \Pi}, \Pi)}(1^\lambda)$ and $\text{Game}^{\mathcal{A}, \text{Wrap}^\Pi}$ is at most $\alpha(\lambda)$.*

We prove Lemma 5.2 with a series of game hops, starting with the original game and ending with the finished wrapper algorithm. The main idea is to, at some point, replace all keys by injective keys. Indeed, as it should be hard to distinguish between lossy keys and injective keys, this switch cannot change the underlying game significantly. The proof is the main technical challenge in this part.

Now, we have everything together to show that key agreement is not possible relative to our lossy function oracle. Let us consider a key agreement protocol between Alice and Bob. We show that we can wrap Alice and Bob using the algorithm defined in the Simulation Lemma, and still have a valid key agreement protocol. Now, as the wrapped parties in the protocol only rely on Π , we show that the Impagliazzo-Rudich adversary is successful here. However, by the Simulation Lemma, this means that the Impagliazzo-Rudich adversary is also successful for the original game (as we would have an efficient distinguisher otherwise).

3 Preliminaries

We say that two random variables X and Y , both indexed by security parameter λ , are computationally indistinguishable if for any probabilistic polynomial-time algorithm A

$$|\Pr[A(1^\lambda, X(1^\lambda)) = 1] - \Pr[A(1^\lambda, Y(1^\lambda)) = 1]|$$

is negligible. We denote by $[X(1^\lambda)]$ the support of a random variable (or algorithm), i.e., the set of output values which are hit with positive probability.

We use the common notion for capturing the statistical distance between two random variables X and Y , both indexed by a security parameter λ . That is,

$$\text{SD}(X, Y) := \frac{1}{2} \sum_z |\Pr[X(1^\lambda) = z] - \Pr[Y(1^\lambda) = z]|,$$

such that the statistical distance is a function of the parameter λ . We sometimes use the fact that for any sequences of random variables X_1, X_2, \dots, X_n , all indexed by λ , it holds

$$\text{SD}(X_1, X_n) \leq \sum_{i=0}^{n-1} \text{SD}(X_i, X_{i+1}).$$

The latter allows us to perform game-hopping type of arguments and bound the statistical distance by the sum of the differences due to the individual hops.

3.1 Lossy Functions

A lossy function can be either injective or compressing, depending on the mode the public key pk has been generated with. The desired mode (inj or loss) is passed as argument to a (randomized) key generating algorithm Gen , together with the security parameter 1^λ . We sometimes write pk_{inj} or pk_{loss} to emphasize that the public key has been generated in either mode, and also $\text{Gen}_{\text{inj}}(\cdot) = \text{Gen}(\cdot, \text{inj})$ as well as $\text{Gen}_{\text{loss}}(\cdot) = \text{Gen}(\cdot, \text{loss})$ to explicitly refer to key generation in injective and lossy mode, respectively. The type of key is indistinguishable to outsiders. This holds even though the adversary can evaluate the function via deterministic algorithm Eval under this key, taking 1^λ , a key pk and a value x of input length $\text{in}(\lambda)$ as input, and returning an image $f_{\text{pk}}(x)$ of an implicitly defined function f . We usually assume that 1^λ is included in pk and thus omit 1^λ for Eval 's input.

In the literature, one can find two slightly different definitions of lossy function. One, which we call the strict variant, requires that for any key generated in injective or lossy mode, the corresponding function is perfectly injective or lossy. In the non-strict variant this only has to hold with overwhelming probability over the choice of the key pk . We define both variants together:

Definition 3.1 (Lossy Functions). An ω -lossy function consists of two efficient algorithms (Gen, Eval) of which Gen is probabilistic and Eval is deterministic and it holds that:

- (a) For $\text{pk}_{\text{inj}} \leftarrow_{\$} \text{Gen}(1^\lambda, \text{inj})$ the function $\text{Eval}(\text{pk}_{\text{inj}}, \cdot) : \{0, 1\}^{\text{in}(\lambda)} \rightarrow \{0, 1\}^*$ is injective with overwhelming probability over the choice of pk_{inj} .
- (b) For $\text{pk}_{\text{loss}} \leftarrow_{\$} \text{Gen}(1^\lambda, \text{loss})$, the function $\text{Eval}(\text{pk}_{\text{loss}}, \cdot) : \{0, 1\}^{\text{in}(\lambda)} \rightarrow \{0, 1\}^*$ is ω -compressing i.e., $|\{\text{Eval}(\text{pk}_{\text{loss}}, \{0, 1\}^{\text{in}(\lambda)})\}| \leq 2^{\text{in}(\lambda) - \omega}$, with overwhelming probability over the choice of pk_{loss} .
- (c) The random variables Gen_{inj} and Gen_{loss} are computationally indistinguishable.

We call the function *strict* if properties (a) and (b) hold with probability 1.

Extremely lossy functions need a more fine-grained approach where the key generation algorithm takes an integer r between 1 and $2^{\text{in}(\lambda)}$ instead of inj or loss . This integer determines the image size, with $r = 2^{\text{in}(\lambda)}$ asking for an injective function. As we want to have functions with a sufficiently high lossiness that the image size is polynomial, say, $p(\lambda)$, we cannot allow for any polynomial adversary. This is so because an adversary making $p(\lambda) + 1$ many random (but distinct) queries to the evaluating function will find a collision in case that pk was lossy, while no collision will be found for an injective key. Instead, we define the minimal r such that $\text{Gen}(1^\lambda, 2^\lambda)$ and $\text{Gen}(1^\lambda, r)$ are indistinguishable based on the runtime and desired advantage of the adversary:

Definition 3.2 (Extremely Lossy Function). An extremely lossy function consists of two efficient algorithms (Gen, Eval) of which Gen is probabilistic and Eval is deterministic and it holds that:

- (a) For $r = 2^{\text{in}(\lambda)}$ and $\text{pk} \leftarrow_{\$} \text{Gen}(1^\lambda, r)$ the function $\text{Eval}(\text{pk}, \cdot) : \{0, 1\}^{\text{in}(\lambda)} \rightarrow \{0, 1\}^*$ is injective with overwhelming probability.
- (b) For $r < 2^{\text{in}(\lambda)}$ and $\text{pk} \leftarrow_{\$} \text{Gen}(1^\lambda, r)$ the function $\text{Eval}(\text{pk}, \cdot) : \{0, 1\}^{\text{in}(\lambda)} \rightarrow \{0, 1\}^*$ has an image size of at most r with overwhelming probability.

- (c) For any polynomials p and d there exists a polynomial q such that for any adversary \mathcal{A} with a runtime bounded by $p(\lambda)$ and any $r \in [q(\lambda), 2^{\text{in}(\lambda)}]$, algorithm \mathcal{A} distinguishes $\text{Gen}(1^\lambda, 2^{\text{in}(\lambda)})$ from $\text{Gen}(1^\lambda, r)$ with advantage at most $\frac{1}{d(\lambda)}$.

Note that extremely lossy functions do indeed imply the definition of (moderately) lossy functions (as long as the lossiness-parameter ω still leaves an exponential-sized image size in the lossy mode):

Lemma 3.3. *Let $(\text{Gen}, \text{Eval})$ be an extremely lossy function. Then $(\text{Gen}, \text{Eval})$ is also a (moderately) lossy function with lossiness parameter $\omega = 0.9\lambda$.*

Proof. Assume $(\text{Gen}, \text{Eval})$ is not such a lossy function, i.e., for $r = 2^{\frac{\lambda}{10}}$ there exists an adversary \mathcal{A} running in time $p(\lambda)$ which is able to distinguish the lossy mode from the injective mode with advantage $d(\lambda)$. Now, as any polynomial $q(\lambda)$ will be eventually smaller than $2^{\frac{\lambda}{10}}$, this directly violates property (c) of the extremely lossy function. \square

3.2 Notions of Black-box Constructions and Oracle Separations

Most constructions in Cryptography are *black-box*, i.e., they are built in a way that they do not depend on the specifics of the instance of the underlying problem or construction (i.e., a one-way function), but only access it as an abstract primitive. Reingold, Trevisan and Vadhan [RTV04] as well as Baecker, Brzuska and Fischlin [BBF13] have given an extensive overview over the different classes of black-box constructions. In this paper, we will be mostly concerned with *fully black-box* (in the notion of [RTV04]) or *BBB black-box* (in the notion of [BBF13]) construction.

Oracle separations, introduced in the seminal work by Impagliazzo and Rudich [IR89], are very useful in proving black-box impossibility results, i.e., that a primitive P cannot be built from another primitive Q in a black-box way. To show such an impossibility result, one comes up with an oracle A (or a collection of such) such that, while a secure implementation of Q exists unconditionally relative to A , every implementation of P can be broken. As black-box constructions are relativizing, i.e., the result does not change if we provide access to some oracle, this shows that no black-box construction can exist.

A simplification of oracle separations, known as the two-oracle technique, was introduced by Hsiao and Reyzin [HR04]. Here, we have two oracles A and B , where the secure construction of Q only relies on A , any adversary gets access to both A and B , though. The two-oracles technique only rules out *fully* black-box constructions.

Lemma 3.4 (Two-Oracle Technique [HR04]). *To show that no fully black-box construction of P from Q exists, it suffices to show that*

1. *there is an efficient implementation N^A of Q ,*
2. *for every efficient implementation M^A of P , there exists an adversary \mathcal{A}^B such that \mathcal{A}^B breaks the security of M^A , and*
3. *there exists no adversary \mathcal{B} such that $\mathcal{B}^{A,B}$ breaks the security of N^A .*

Note that according to this lemma, adversary \mathcal{A} does not have access to the oracle A . However, we can always redefine our oracles as $A' = A$ and $B' = (A, B)$, which will yield the same result, but allows \mathcal{A} to access oracle A as well.

Another helpful lemma in fixing an oracle as part of an oracle separation is the Borel-Cantelli Lemma:

Lemma 3.5 (Borel-Cantelli). *Let E_1, E_2, \dots be a sequence of events on the same probability space. Then, if the sum of probabilities of events converges, the probability that infinitely many of the events occur is 0:*

$$\sum_{\lambda=1}^{\infty} \Pr[E_{\lambda}] < \infty \Rightarrow \Pr \left[\bigwedge_{k=1}^{\infty} \bigvee_{\lambda \geq k} E_{\lambda} \right] = 0$$

3.3 Oraclecrypt

In his seminal work [Imp95], Impagliazzo introduced five possible worlds we might be living in, including two in which computational cryptography exists: Minicrypt, in which one-way functions exist, but public-key cryptography does not, and Cryptomania, in which public-key cryptography exists as well. In reference to this classification, cryptographic primitives that can be built from one-way functions in a black-box way are often called Minicrypt primitives.

In this work, we are interested in the set of all primitives that exist relative to a truly random function. This of course includes all Minicrypt primitives, as one-way functions exist relative to a truly random function (with high probability), but it also includes a number of other primitives, like collision-resistant hash functions and exponentially-secure one-way functions, for which we don't know that they exist relative to a one-way function, or even have a black-box impossibility result. In reference to the set of Minicrypt primitives, we will call all primitives existing relative to a truly random function *Oraclecrypt* primitives.

Definition 3.6 (Oraclecrypt). We say that a cryptographic primitive is an Oraclecrypt primitive, if there exists an implementation relative to truly random function oracle (except for a measure zero of random oracles).

We will now show that by this definition, indeed, many symmetric primitives are Oraclecrypt primitives:

Lemma 3.7. *The following primitives are Oraclecrypt primitives:*

- *Exponentially-secure one-way functions,*
- *Exponentially-secure collision resistant hash functions,*
- *One-way product functions.*

We moved the proof for this lemma to the Appendix C.1.

4 On the Impossibility of Building (E)LFs in Oraclecrypt

In this chapter, we will show that we cannot build lossy functions from a number of symmetric primitives, including (exponentially-secure) one-way functions, collision-resistant hash functions and one-way product functions, in a black-box way. Indeed, we will show that any primitive in Oraclecrypt is not enough to build lossy functions. As extremely lossy functions imply (moderately) lossy functions, this result applies to them as well.

Note that for exponentially-secure one-way functions, this was already known for lossy functions that are sufficiently lossy: Lossy functions with sufficient lossiness imply collision-resistant hash functions, and Simon’s result [Sim98] separates these from (exponentially-secure) one-way functions. However, this does not apply for lossy functions with e.g. only a constant number of bits of lossiness.

Theorem 4.1. *There exists no fully black-box construction of lossy functions from any Oraclecrypt primitive, including exponentially-secure one-way functions, collision resistant hash functions, and one-way product functions.*

Our proof for this Theorem follows a proof idea by Pietrzak, Rosen and Segev [PRS12], which they used to show that lossy functions cannot be amplified well, i.e., one cannot build a lossy function which is very compressing in the lossy mode from a lossy function that is only slightly compressing in the lossy mode. Conceptually, we show an oracle separation between lossy functions and Oraclecrypt: For this, we will start by introducing two oracles, a random oracle and a modified PSPACE oracle. We will then, for a candidate construction of a lossy function based on the random oracle and a public key pk , approximate the heavy queries asked by $\text{Eval}(\text{pk}, \cdot)$ to the random oracle. Next, we show that this approximation of the set of heavy queries is actually enough for us approximating the image size of $\text{Eval}(\text{pk}, \cdot)$ (using our modified PSPACE oracle) and therefore gives an efficient way to distinguish lossy keys from injective keys. Finally, we have to fix a set of oracles (instead of arguing with a distribution of oracles) and then use the two-oracle technique [HR04] to show the theorem.

4.1 Introducing the Oracles

A common oracle to use in an oracle separation in cryptography is the PSPACE oracle, as relative to this oracle, all non-information theoretic cryptography is broken. As we do not know which (or whether any) cryptographic primitives exist unconditionally, this is a good way to level the playing field. However, in our case, PSPACE is not quite enough. In our proof, we want to calculate the image size of a function relative to a (newly chosen) random oracle. It is not possible to simulate this oracle by lazy-sampling, though, as to calculate the image size of a function, we might have to save an exponentially large set of queries, which is not possible in PSPACE. Therefore, we give the PSPACE oracle access to its own random oracle $\mathcal{O}' : \{0, 1\}^\lambda \rightarrow \{0, 1\}^\lambda$ and will give every adversary access to $\text{PSPACE}^{\mathcal{O}'}$.

The second oracle is a random oracle $\mathcal{O} : \{0, 1\}^\lambda \rightarrow \{0, 1\}^\lambda$. Now, we know that a number of primitives exist relative to a random function, including exponentially-secure one-way functions, collision-resistant hash functions and even more complicated primitives like one-way product functions. Further, they still exist if we give the adversary access to $\text{PSPACE}^{\mathcal{O}'}$, too, as \mathcal{O}' is independent from \mathcal{O} and $\text{PSPACE}^{\mathcal{O}'}$ does not have direct access to \mathcal{O} .

We will now show that every candidate construction of a lossy function with access to \mathcal{O} can be broken by an adversary $\mathcal{A}^{\mathcal{O}, \text{PSPACE}^{\mathcal{O}'}}$. Note that we do not give the construction access to $\text{PSPACE}^{\mathcal{O}'}$ — this is necessary, as \mathcal{O}' should look like a randomly sampled oracle to the construction. However, giving the construction access to $\text{PSPACE}^{\mathcal{O}'}$ would enable the construction to behave differently for this specific oracle \mathcal{O}' . Not giving the construction access to the oracle is fine, however, as we are using the two-oracle technique.

Our proof for Theorem 4.1 will now work in two steps. First, we will show that with overwhelming probability over independently sampled \mathcal{O} and \mathcal{O}' , no lossy functions exist relative to \mathcal{O} and $\text{PSPACE}^{\mathcal{O}'}$. However, for an oracle separation, we need one fixed oracle. Therefore, as a second step

(Section 4.4), we will use standard techniques to select one set of oracles relative to which any of our Oraclecrypt primitives exist, but lossy functions do not.

For the first step, we will now define how our definition of lossy functions with access to both oracles looks like:

Definition 4.2 (Lossy functions with Oracle Access). A family of functions

$$\text{Eval}^{\mathcal{O}}(\text{pk}, \cdot) : \{0, 1\}^{\text{in}(\lambda)} \rightarrow \{0, 1\}^*$$

with public key pk and access to the oracles \mathcal{O} is called ω -lossy if there exist two PPT algorithms Gen_{inj} and Gen_{loss} such that for all $\lambda \in \mathbb{N}$,

- (a) For all pk in $[\text{Gen}_{\text{inj}}^{\mathcal{O}}(1^\lambda)] \cup [\text{Gen}_{\text{loss}}^{\mathcal{O}}(1^\lambda)]$, $\text{Eval}^{\mathcal{O}}(\text{pk}, \cdot)$ is computable in polynomial time in λ ,
- (b) For $\text{pk} \leftarrow_{\$} \text{Gen}_{\text{inj}}^{\mathcal{O}}(1^\lambda)$, $\text{Eval}^{\mathcal{O}}(\text{pk}, \cdot)$ is injective with overwhelming probability (over the choice of pk as well as the random oracle \mathcal{O}),
- (c) For $\text{pk} \leftarrow_{\$} \text{Gen}_{\text{loss}}^{\mathcal{O}}(1^\lambda)$, $\text{Eval}^{\mathcal{O}}(\text{pk}, \cdot)$ is ω -compressing with overwhelming probability (over the choice of pk as well as the random oracle \mathcal{O})
- (d) The random variables $\text{Gen}_{\text{inj}}^{\mathcal{O}}$ and $\text{Gen}_{\text{loss}}^{\mathcal{O}}$ are computationally indistinguishable for any polynomial-time adversary $\mathcal{A}^{\mathcal{O}, \text{PSPACE}^{\mathcal{O}'}}$ with access to both \mathcal{O} and $\text{PSPACE}^{\mathcal{O}'}$.

4.2 Approximating the Set of Heavy Queries

In the next two subsections, we will construct an adversary $\mathcal{A}^{\mathcal{O}, \text{PSPACE}^{\mathcal{O}'}}$ against lossy functions with access to the random oracle \mathcal{O} as described in Definition 4.2.

Let $(\text{Gen}^{\mathcal{O}}, \text{Eval}^{\mathcal{O}})$ be some candidate implementation of a lossy function relative to the oracle \mathcal{O} . Further, let $\text{pk} \leftarrow \text{Gen}^{\mathcal{O}}$ be some public key generated by either Gen_{inj} or Gen_{loss} . Looking at the queries asked by the lossy function to \mathcal{O} , we can divide them into two parts: The queries asked during the generation of the key pk , and the queries asked during the execution of $\text{Eval}^{\mathcal{O}}(\text{pk}, \cdot)$. We will denote the queries asked during the generation of pk by the set Q_G . As the generation algorithm has to be efficient, Q_G has polynomial size. Let k_G be the maximal number of queries asked by any of the two generators. Further, denote by k_f the maximum number of queries of $\text{Eval}^{\mathcal{O}}(\text{pk}, x)$ for any pk and x — again, k_f is polynomial. Finally, let $k = \max\{k_G, k_f\}$.

The set of all queries done by $\text{Eval}(\text{pk}, \cdot)$ for a fixed key pk might be of exponential size, as the function might ask different queries for each input x . However, we are able to shrink the size of the relevant subset significantly, if we concentrate on *heavy* queries — queries that appear for a significant fraction of all inputs x :

Definition 4.3 (Heavy Queries). Let k be the maximum number of \mathcal{O} -queries made by the generator $\text{Gen}^{\mathcal{O}}$, or the maximum number of queries of $\text{Eval}(\text{pk}, \cdot)$ over all inputs $x \in \{0, 1\}^{\text{in}(\lambda)}$, whichever is higher. Fix some key pk and a random oracle \mathcal{O} . We call a query q to \mathcal{O} *heavy* if, for at least a $\frac{1}{10k}$ -fraction of $x \in \{0, 1\}^{\text{in}(\lambda)}$, the evaluation $\text{Eval}(\text{pk}, x)$ queries \mathcal{O} about q at some point. We denote by Q_H the set of all heavy queries (for pk, \mathcal{O}).

The set of heavy queries is polynomial, as $\text{Eval}^{\mathcal{O}}(\text{pk}, \cdot)$ only queries the oracle a polynomial number of times and each heavy query has to appear in a polynomial fraction of all x . Further, we

will show that the adversary $\mathcal{A}^{\mathcal{O}, \text{PSPACE}^{\mathcal{O}'}}$ is able to approximate the set of heavy queries, and that this approximation is actually enough to decide whether pk was generated in injective or in lossy mode. We will start with a few key observations that help us prove this statement.

The first one is that the generator, as it is an efficiently-computable function, will only query \mathcal{O} at polynomially-many positions, and these polynomially-many queries already define whether the function is injective or lossy:

Observation 1. *Let Q_G denote the queries by the generator. For a random $\text{pk} \leftarrow \text{Gen}_{\text{inj}}^{\mathcal{O}}$ generated in injective mode and a random \mathcal{O}' that is consistent with Q_G , the image size of $\text{Eval}^{\mathcal{O}'}(\text{pk}, \cdot)$ is 2^λ (except with a negligible probability over the choice of pk and \mathcal{O}'). Similarly, for a random $\text{pk} \leftarrow \text{Gen}_{\text{loss}}^{\mathcal{O}}$ generated in lossy mode and a random \mathcal{O}' that is consistent with Q_G , the image size of $\text{Eval}^{\mathcal{O}'}(\text{pk}, \cdot)$ is at most $2^{\lambda-1}$ (except with a negligible probability over the choice of pk and \mathcal{O}').*

This follows directly from the definition: As $\text{Gen}_{\mathcal{O}}^{\mathcal{O}}$ has no information about \mathcal{O} except the queries Q_G , properties (2) and (3) of Definition 3.1 have to hold for every random oracle that is consistent with \mathcal{O} on Q_G . We will use this multiple times in the proof to argue that queries to \mathcal{O} that are not in Q_G are, essentially, useless randomness for the construction, as the construction has to work with almost any possible answer returned by these queries.

An adversary is probably very much interested in learning the queries Q_G . There is no way to capture them in general, though. Here, we need our second key observation. Lossiness is very much a global property: to switch a function from lossy to injective, at least half of all inputs x to $\text{Eval}^{\mathcal{O}}(\text{pk}, x)$ must produce a different result, and vice versa. However, as we learned from the first observation, whether $\text{Eval}^{\mathcal{O}}(\text{pk}, \cdot)$ is lossy or injective, depends just on Q_G . Therefore, some queries in Q_G must be used over and over again for different inputs x — and will therefore appear in the heavy set Q_H . Further, due to the heaviness of these queries, the adversary is indeed able to learn them!

Our proof works alongside these two observations: First, we show in Lemma 4.4 that for any candidate lossy function, an adversary is able to compute a set \hat{Q}_H of the interesting heavy queries. Afterwards, we show in Lemma 4.6 that we can use \hat{Q}_H to decide whether $\text{Eval}^{\mathcal{O}}(\text{pk}, \cdot)$ is lossy or injective, breaking the indistinguishability property of the lossy function.

Lemma 4.4. *Let $\text{Eval}^{\mathcal{O}}(\text{pk}, \cdot)$ be a (non-strict) lossy function and $\text{pk} \leftarrow \text{Gen}_{\mathcal{O}}^{\mathcal{O}}(1^\lambda)$ for oracle \mathcal{O} . Then we can compute in probabilistic polynomial-time (in λ) a set \hat{Q}_H which contains all heavy queries of $\text{Eval}^{\mathcal{O}}(\text{pk}, \cdot)$ for pk, \mathcal{O} with overwhelming probability.*

Proof. To find the heavy queries we will execute $\text{Eval}^{\mathcal{O}}(\text{pk}, x)$ for t random inputs x and record all queries to \mathcal{O} in \hat{Q}_H . We will now argue that, with high probability, \hat{Q}_H contains all heavy queries.

First, recall that a query is heavy if it appears for at least an ε -fraction of inputs to $\text{Eval}^{\mathcal{O}}(\text{pk}, \cdot)$ for $\varepsilon = \frac{1}{10k}$. Therefore, the probability for any specific heavy query q_{heavy} to not appear in \hat{Q}_H after the t evaluations can be bounded by

$$\Pr[q_{\text{heavy}} \notin \hat{Q}_H] = (1 - \varepsilon)^t \leq 2^{-\varepsilon t}.$$

Furthermore, there exist at most $\frac{k}{\varepsilon}$ heavy queries, because each heavy query accounts for at least $\varepsilon \cdot 2^{\text{in}(\lambda)}$ of the at most $k \cdot 2^{\text{in}(\lambda)}$ possible queries of $\text{Eval}^{\mathcal{O}}(\text{pk}, x)$ when iterating over all x . Therefore,

the probability that any heavy query q_{heavy} is not included in \hat{Q}_H is given by

$$\Pr\left[\exists q_{\text{heavy}} \notin \hat{Q}_H\right] \leq \frac{k}{\varepsilon} \cdot 2^{-\varepsilon t}$$

Choosing $t = 10k\lambda$ we get

$$\Pr\left[\exists q_{\text{heavy}} \notin \hat{Q}_H\right] \leq 10k^2 \cdot 2^{-\lambda}$$

which is negligible. Therefore, with all but negligible probability, all heavy queries are included in \hat{Q}_H . \square

4.3 Distinguishing Lossiness from Injectivity

We next make the transition from oracle \mathcal{O} to our PSPACE-augmenting oracle \mathcal{O}' . According to the previous subsection, we can compute (a superset \hat{Q}_H of) the heavy queries efficiently. Then we can fix the answers of oracle \mathcal{O} on such frequently asked queries in \hat{Q}_H , but otherwise use the independent oracle \mathcal{O}' instead. Denote this partly-set oracle by $\mathcal{O}'_{|\hat{Q}_H}$. Then the distinguisher for injective and lossy keys, given some pk , can approximate the image size of $\#\text{im}(\text{Eval}^{\mathcal{O}'_{|\hat{Q}_H}}(\text{pk}, \cdot))$ with the help of its PSPACE $^{\mathcal{O}'}$ oracle and thus also derives a good approximation for the actual oracle \mathcal{O} . This will be done in Lemma 4.6.

We still have to show that the non-heavy queries do not violate the above approach. According to the proof of Lemma 4.5 it suffices to look at the case that the image sizes of oracles $\mathcal{R} := \mathcal{O}'_{|\hat{Q}_H}$ and for oracle $\mathcal{R}' := \mathcal{O}'_{|\hat{Q}_H \cup Q_G}$, where we also fix on the key generator's non-heavy queries to values from \mathcal{O} , cannot differ significantly. Put differently, missing out the generator's non-heavy queries Q_G in \hat{Q}_H only slightly affects the image size of $\text{Eval}^{\mathcal{O}'_{|\hat{Q}_H}}(\text{pk}, \cdot)$, and we can proceed with our approach to consider only heavy queries.

Lemma 4.5. *Let $\text{pk} \leftarrow \text{Gen}_?^{\mathcal{R}}(1^\lambda)$ and $Q_G^{\text{nonh}} = \{q_1, \dots, q_{k'}\}$ be the k' generator's queries to \mathcal{R} in Q_G when computing pk that are not heavy for pk, \mathcal{R} . Then, for any oracle \mathcal{R}' that is identical to \mathcal{R} everywhere except for the queries in Q_G^{nonh} , i.e., $\mathcal{R}(q) = \mathcal{R}'(q)$ for any $q \notin Q_G^{\text{nonh}}$, the image sizes of $\text{Eval}^{\mathcal{R}}(\text{pk}, \cdot)$ and $\text{Eval}^{\mathcal{R}'}(\text{pk}, \cdot)$ differ by at most $\frac{2^{\text{in}(\lambda)}}{10}$.*

Proof. As the queries in Q_G^{nonh} are non-heavy, every $q_i \in Q_G^{\text{nonh}}$ is queried for at most $\frac{2^{\text{in}(\lambda)}}{10k}$ inputs x to $\text{Eval}^{\mathcal{R}}(\text{pk}, \cdot)$ when evaluating the function. Therefore, any change in the oracle \mathcal{R} at $q_i \in Q_G^{\text{nonh}}$ affects the output of $\text{Eval}^{\mathcal{R}}(\text{pk}, \cdot)$ for at most $\frac{2^{\text{in}(\lambda)}}{10k}$ inputs. Hence, when considering the oracle \mathcal{R}' , which differs from \mathcal{R} only on the k' queries from Q_G^{nonh} , moving from \mathcal{R} to \mathcal{R}' for evaluating $\text{Eval}^{\mathcal{R}}(\text{pk}, \cdot)$ changes the output for at most $\frac{k' 2^{\text{in}(\lambda)}}{10k}$ inputs x . In other words, letting Δ_f denote the set of all x such that $\text{Eval}^{\mathcal{R}}(\text{pk}, x)$ queries some $q \in Q_G^{\text{nonh}}$ during the evaluation, we know that

$$|\Delta_f| \leq \frac{k' 2^{\text{in}(\lambda)}}{10k}$$

and

$$\text{Eval}^{\mathcal{R}}(\text{pk}, x) = \text{Eval}^{\mathcal{R}'}(\text{pk}, x) \text{ for all } x \notin \Delta_f.$$

We are interested in the difference of the two image sizes of $\text{Eval}^{\mathcal{R}}(\text{pk}, \cdot)$ and $\text{Eval}^{\mathcal{R}'}(\text{pk}, \cdot)$. Each $x \in \Delta_f$ may add or subtract an image in the difference, depending on whether the modified output $\text{Eval}^{\mathcal{R}'}(\text{pk}, x)$ introduces a new image or redirects the only image $\text{Eval}^{\mathcal{R}}(\text{pk}, x)$ to an already existing one. Therefore, the difference between the image sizes is at most

$$\left| \#\text{im}(\text{Eval}^{\mathcal{R}}(\text{pk}, \cdot)) - \#\text{im}(\text{Eval}^{\mathcal{R}'}(\text{pk}, \cdot)) \right| \leq \frac{k' 2^{\text{in}(\lambda)}}{10k} \leq \frac{2^{\text{in}(\lambda)}}{10},$$

where the last inequality is due to $k' \leq k$. \square

Lemma 4.6. *Given $\hat{Q}_H \supseteq Q_H$, we can decide correctly whether $\text{Eval}^{\mathcal{O}}(\text{pk}, \cdot)$ is lossy or injective with overwhelming probability.*

Proof. As described in Section 4.1, we give the adversary, who has to distinguish a lossy key from an injective key, access to $\text{PSPACE}^{\mathcal{O}'}$, where \mathcal{O}' is another random oracle sampled independently of \mathcal{O} . This is necessary for the adversary, as we want to calculate the image size of $\text{Eval}^{\mathcal{O}'}(\text{pk}, \cdot)$ relative to a random oracle \mathcal{O}' , and we cannot do this in PSPACE with lazy sampling.

We will consider the following adversary \mathcal{A} : It defines an oracle $\mathcal{O}'_{|\hat{Q}_H}$ that is identical to \mathcal{O}' for all queries $q \notin \hat{Q}_H$ and identical to \mathcal{O} for all queries $q \in \hat{Q}_H$. Then, it calculates the image size

$$\#im(\text{Eval}^{\mathcal{O}'_{|\hat{Q}_H}}(\text{pk}, \cdot)) = \left| \{ \text{Eval}^{\mathcal{O}'_{|\hat{Q}_H}}(\text{pk}, \{0, 1\}^{\text{in}(\lambda)}) \} \right|.$$

Note that this can be done efficiently using $\text{PSPACE}^{\mathcal{O}'}$ as well as polynomially many queries to \mathcal{O} . If $\#im(\text{Eval}^{\mathcal{O}'_{|\hat{Q}_H}}(\text{pk}, \cdot))$ is bigger than $\frac{3}{4}2^{\text{in}(\lambda)}$, \mathcal{A} will guess that $\text{Eval}^{\mathcal{O}}(\text{pk}, \cdot)$ is injective, and lossy otherwise. For simplicity reasons, we will assume from now on that pk was generated by Gen_{inj} — the case where pk was generated by Gen_{loss} follows by a symmetric argument.

First, assume that all queries Q_G of the generator are included in \hat{Q}_H . In this case, any \mathcal{O}' that is consistent with Q_H is also consistent with all the information Gen_{inj} have about \mathcal{O} . However, this means that by definition, $\text{Eval}^{\mathcal{O}}(\text{pk}, \cdot)$ has to be injective with overwhelming probability, and therefore, an adversary can easily check whether pk was created by Gen_{inj} .

Otherwise, let $q_1, \dots, q_{k'}$ be a set of queries in Q_G which are not included in \hat{Q}_H . With overwhelming probability, this means that $q_1, \dots, q_{k'}$ are all non-heavy. We now apply Lemma 4.5 for oracles $\mathcal{R} := \mathcal{O}'_{|\hat{Q}_H}$ and $\mathcal{R}' := \mathcal{O}'_{|\hat{Q}_H \cup Q_G}$. These two oracles may only differ on the non-heavy queries in Q_G , where \mathcal{R} coincides with \mathcal{O}' and \mathcal{R}' coincides with \mathcal{O} ; otherwise the oracles are identical. Lemma 4.5 tells us that this will change the image size by at most $\frac{2^{\text{in}(\lambda)}}{10}$. Therefore, with overwhelming probability, the image size calculated by the distinguisher is bounded from below by

$$\#im(\text{Eval}^{\mathcal{O}'_{|\hat{Q}_H}}(\text{pk}, \cdot)) \geq 2^{\text{in}(\lambda)} - \frac{2^{\text{in}(\lambda)}}{10} \geq \frac{3}{4}2^{\text{in}(\lambda)}$$

and the distinguisher will therefore correctly decide that $\text{Eval}^{\mathcal{O}}(\text{pk}, \cdot)$ is in injective mode. \square

Theorem 4.7. *Let \mathcal{O} and \mathcal{O}' be two independent random oracles. Then, with overwhelming probability over the choice of the two random oracles, lossy functions do not exist relative the oracles \mathcal{O} and $\text{PSPACE}^{\mathcal{O}'}$.*

Proof. Given the key pk , our distinguisher (with oracle access to random oracle \mathcal{O}) against the injective and lossy mode first runs the algorithm of Lemma 4.4 to efficiently construct a super set \hat{Q}_H of the heavy queries Q_H for pk, \mathcal{O} . This succeeds with overwhelming probability, and from now on we assume that indeed $Q_H \subseteq \hat{Q}_H$. Then our algorithm continues by running the decision procedure of Lemma 4.6 to distinguish the cases. Using the $\text{PSPACE}^{\mathcal{O}'}$ oracle, the latter can also be carried out efficiently. \square

4.4 Fixing an Oracle

We have shown now (in Theorem 4.7) that no lossy function exists relative to a random oracle with overwhelming probability. However, to prove our main theorem, we have to show that there exists one fixed oracle relative to which one-way functions (or collision-resistant hash functions, or one-way product functions) exist, but lossy functions do not.

In Lemma 3.7, we have already shown that (exponentially-secure) one-way functions, collision-resistant hash functions and one-way product functions exist relative to a random oracle with high probability. In the next lemma, we will show that there exists a fixed oracle relative to which exponentially-secure one-way functions exist, but lossy functions do not. The proofs for existence of oracles relative to which exponentially-secure collision-resistant hash functions or one-way product functions, but no lossy functions exist follow similarly.

Lemma 4.8. *There exists a fixed set of oracles \mathcal{O} , $\text{PSPACE}^{\mathcal{O}'}$ such that relative to these oracles, one-way functions using \mathcal{O} exist, but no construction of lossy functions from \mathcal{O} exists.*

Now, our main theorem of this section directly follows from this lemma (and its variants for the other primitives):

Theorem 4.1 (restated). *There exists no fully black-box construction of lossy functions from any Oraclecrypt primitive, including exponentially-secure one-way functions, collision resistant hash functions, and one-way product functions.*

Proof. By Lemma 4.8, there exist two oracles \mathcal{O} and $\text{PSPACE}^{\mathcal{O}'}$ such that exponentially-secure one-way functions (or any of the other Oraclecrypt primitives) exist relative to \mathcal{O} , even if the adversary against the one-wayness has additional access to $\text{PSPACE}^{\mathcal{O}'}$. However, there exists an adversary with access to \mathcal{O} and $\text{PSPACE}^{\mathcal{O}'}$ that breaks any construction of a lossy function relative to \mathcal{O} . The two-oracle technique then shows that this means no fully black-box construction of lossy functions from exponentially-secure one-way functions (or, from any other primitive in Oraclecrypt) can exist. \square

Let us now focus on Lemma 4.8 again. Up to this point, we have argued over distributions of oracles (i.e., we have required the oracles \mathcal{O} and \mathcal{O}' to be chosen at random from any possible oracle). For a proper oracle separation, however, we have to show that our results hold for a set of fixed oracles.

We use the following Borel-Cantelli-style theorem from Mahmoody, Mohammed, Nematihaji, Pass and Shelat [MMNP16]:

Lemma 4.9 ([MMNP16], Lemma 2.9). *Let E_1, E_2, \dots be a sequence of events such that there exists a constant c , $0 < c < 1$, such that for all $n \in \mathbb{N}$: $\Pr[E_n] \geq c$. Then,*

$$\Pr \left[\bigwedge_{k=1}^{\infty} \bigvee_{n>k} E_n \right] \geq c$$

Further, we also need the so-called splitting lemma [PS00] which allows to relate the probability of events over a product space $X \times Y$ to the ones when the X -part is fixed:

Lemma 4.10 (Splitting Lemma [PS00]). *Let $\mathcal{D} = \mathcal{D}_X \times \mathcal{D}_Y$ be some product distributions over $X \times Y$. Let $Z \subseteq X \times Y$ be such that $\Pr_{\mathcal{D}}[(x, y) \in Z] > \varepsilon$. For any $\alpha < \varepsilon$ call $x \in X$ to be α -good if*

$$\Pr_{y \leftarrow \mathcal{D}_Y} [(x, y) \in Z] > \varepsilon - \alpha.$$

Then we have $\Pr_{x \leftarrow \mathcal{D}_X}[x \text{ is } \alpha\text{-good}] \geq \alpha$.

Proof (of Lemma 4.8). We will show that for each primitive out of exponentially-secure one-way functions, collision-resistant hash functions and one-way product functions, there exist two fixed oracles \mathcal{O} and \mathcal{O}' such that, relative to \mathcal{O} and $\text{PSPACE}^{\mathcal{O}'}$, this primitive exists, but lossy functions do not. We will now prove this for exponentially-secure one-way functions – the proof for the other primitives works analogously.

Let $\mathcal{A}^{\mathcal{O}, \text{PSPACE}^{\mathcal{O}'}}$ be the adversary against lossy functions as described in the last sections. Then, by Theorem 4.7, we know that the adversary wins with overwhelming probability over the choice of the two random oracles as well as any internal randomness of \mathcal{A} , i.e., there exists a negligible function $\varepsilon(\lambda)$ such that

$$\forall \lambda \in \mathbb{N} : \Pr_{\mathcal{O}, \mathcal{O}', \mathcal{A}} \left[\mathcal{A}^{\mathcal{O}, \text{PSPACE}^{\mathcal{O}'}} \text{ wins} \right] \geq 1 - \varepsilon(\lambda).$$

We will now first fix an oracle \mathcal{O} . To do this, we have to split out oracle \mathcal{O} using the Splitting Lemma (4.10) with $\alpha = \frac{1}{2}(1 - \varepsilon(\lambda))$:

$$\forall \lambda \in \mathbb{N} : \Pr_{\mathcal{O}} \left[\Pr_{\mathcal{O}', \mathcal{A}} \left[\mathcal{A}^{\mathcal{O}, \text{PSPACE}^{\mathcal{O}'}} \text{ wins} \right] \geq \frac{1}{2}(1 - \varepsilon(\lambda)) \right] \geq \frac{1}{2}(1 - \varepsilon(\lambda))$$

Next, we want to use the constant version of the Borel-Cantelli Lemma (4.9), for which we need a constant bound in the outer probability. For large security parameters, we can just bound the outer probability by $\frac{1}{3}$, as a negligible function will eventually be smaller than any constant. For small security parameters, this might not work directly, but we can modify the negligible function ε' to be 1 in these cases (ε' stays negligible, as the modification only happens for small security parameters).

$$\forall \lambda \in \mathbb{N} : \Pr_{\mathcal{O}} \left[\Pr_{\mathcal{O}', \mathcal{A}} \left[\mathcal{A}^{\mathcal{O}, \text{PSPACE}^{\mathcal{O}'}} \text{ wins} \right] \geq \frac{1}{2}(1 - \varepsilon'(\lambda)) \right] \geq \frac{1}{3}$$

Now, the constant version of Borel-Cantelli gives us

$$\Pr_{\mathcal{O}} \left[\bigwedge_{k=1}^{\infty} \bigvee_{\lambda > k} \Pr_{\mathcal{O}', \mathcal{A}} \left[\mathcal{A}^{\mathcal{O}, \text{PSPACE}^{\mathcal{O}'}} \text{ wins} \right] \geq \frac{1}{2}(1 - \varepsilon'(\lambda)) \right] \geq \frac{1}{3}.$$

In other words, for a $\frac{1}{3}$ fraction of all oracles \mathcal{O} , adversary \mathcal{A} wins for infinitely many security parameters with probability $\frac{1}{2}(1 - \varepsilon'(\lambda))$ (over the choice of \mathcal{O}' and the randomness of \mathcal{A}).

Now, we want to have a fixed oracle \mathcal{O} relative to which not only lossy functions do not exist, but also exponentially-secure one-way functions do exist. However, by Lemma 3.7 we know that exponentially-secure one-way functions exist relative to a 1-measure of random oracles \mathcal{O} . Therefore, it is clear we will find a fixed oracle \mathcal{O} relative to which exponentially-secure one-way functions exist, but lossy functions do not. Let us now fix such an oracle \mathcal{O} .

Now, it remains for us to fix the other random oracle, \mathcal{O}' . We can apply the splitting lemma again to get

$$\forall \lambda \in \mathbb{N} : \Pr_{\mathcal{O}'} \left[\Pr_{\mathcal{A}} \left[\mathcal{A}^{\mathcal{O}, \text{PSPACE}^{\mathcal{O}'}} \text{ wins} \right] \geq \frac{1}{4}(1 - \varepsilon'(\lambda)) \right] \geq \frac{1}{4}(1 - \varepsilon')$$

By a similar argument as above, we can fix the outer probability by modifying the negligible function to ε'' , which lets us apply Borell-Cantelli again:

$$\Pr_{\mathcal{O}'} \left[\bigwedge_{k=1}^{\infty} \bigvee_{\lambda > k} \Pr_{\mathcal{A}} \left[\mathcal{A}^{\mathcal{O}, \text{PSPACE}^{\mathcal{O}'}} \text{ wins} \right] \geq \frac{1}{4}(1 - \varepsilon''(\lambda)) \right] \geq \frac{1}{5}.$$

Fixing an oracle \mathcal{O}' out of the $\frac{1}{5}$ fraction gives us the desired result. \square

5 On the Impossibility of Building Key Agreement Protocols from (Extremely) Lossy Functions

In the previous section we showed that lossy functions cannot be built from many symmetric primitives in a black-box way. This raises the question if lossy functions and extremely lossy functions might be inherent asymmetric primitives. In this section we provide evidence to the contrary, showing that key agreement cannot be built from lossy functions in a black-box way. For this, we adapt the proof by Impagliazzo and Rudich [IR89] showing that key agreement cannot be built from one-way functions to our setting. We extend this result to also hold for extremely lossy functions, but in a slightly weaker setting.

5.1 Lossy Function Oracle

We specify our lossy function oracle relative to a (random) permutation oracle Π , and further sample (independently of Π) a second random permutation Γ as integral part of our lossy function oracle. The core idea of the oracle is to evaluate $\text{Eval}^{\Gamma, \Pi}(\text{pk}_{\text{inj}}, x) = \Pi(\text{pk}_{\text{inj}} \| ax + b)$ for the injective mode, but set $\text{Eval}^{\Gamma, \Pi}(\text{pk}_{\text{loss}}, x) = \Pi(\text{pk}_{\text{loss}} \| \text{setlsb}(ax + b))$ for the lossy mode, where a, b describe a pairwise independent hash permutation $ax + b$ over the field $\text{GF}(2^\mu)$ with $a \neq 0$ and setlsb sets the least significant bit to 0. Then the lossy function is clearly two to one. The values a, b will be chosen during key generation and placed into the public key, but we need to hide them from the adversary in order to make the keys of the two modes indistinguishable. Else a distinguisher, given pk , could check if $\text{Eval}^{\Gamma, \Pi}(\text{pk}, x) = \text{Eval}^{\Gamma, \Pi}(\text{pk}, x')$ for appropriately computed $x \neq x'$ with $\text{setlsb}(ax + b) = \text{setlsb}(ax' + b)$. Therefore, we will use the secret permutation Γ to hide the values in the public key. We will denote the preimage of pk under Γ as pre-key.

Another feature of our construction is to ensure that the adversary cannot generate a lossy key pk_{loss} without calling $\text{Gen}^{\Gamma, \Pi}$ in lossy mode, while allowing it to generate keys in injective mode. We accomplish this by having a value k in our public pre-key that is zero for lossy keys and may take any non-zero value for an injective public key. Therefore, with overwhelming probability, any key generated by the adversary without a call to the $\text{Gen}^{\Gamma, \Pi}$ oracle will be an injective key.

We finally put both ideas together. For key generation we hide a, b and also the string k by creating pk as a commitment to the values, $\text{pk} \leftarrow \Gamma(k \| a \| b \| z)$ for random z . To unify calls to Γ in regard of the security parameter λ , we will choose all entries in the range of $\lambda/5$.² When receiving pk the evaluation algorithm $\text{Eval}^{\Gamma, \Pi}$ first recovers the preimage $k \| a \| b \| z$ under Π , then checks if k signals injective or lossy mode, and then computes $\Pi(a \| b \| ax + b)$ resp. $\Pi(a \| b \| \text{setlsb}(ax + b))$ as the output.

Definition 5.1 (Lossy Function Oracle). Let Π, Γ be permutation oracles with $\Pi, \Gamma : \{0, 1\}^\lambda \rightarrow \{0, 1\}^\lambda$ for all λ . Let $\mu = \mu(\lambda) = \lfloor (\lambda - 2)/5 \rfloor$ and $\text{pad} = \text{pad}(\lambda) = \lambda - 2 - 5\mu$ define the length that the rounding-off loses to $\lambda - 2$ in total (such that $\text{pad} \in \{0, 1, 2, 3, 4\}$). Define the lossy function $(\text{Gen}^{\Gamma, \Pi}, \text{Eval}^{\Gamma, \Pi})$ with input length $\text{in}(\lambda) = \mu(\lambda)$ relative to Π and Γ now as follows:

Key Generation: Oracle $\text{Gen}^{\Gamma, \Pi}$ on input 1^λ and either mode inj or loss picks random $b \leftarrow_{\$} \{0, 1\}^\mu$, $z \leftarrow_{\$} \{0, 1\}^{2\mu + \text{pad}}$ and random $a, k \leftarrow_{\$} \{0, 1\}^\mu \setminus \{0^\mu\}$. For mode inj the algorithm returns $\Gamma(k \| a \| b \| z)$. For mode loss the algorithm returns $\Gamma(0^\mu \| a \| b \| z)$ instead.

²For moderately lossy function we could actually use $\lambda/4$ but for compatibility to the extremely lossy case it is convenient to use $\lambda/5$ already here.

Evaluation: On input $\mathbf{pk} \in \{0, 1\}^\lambda$ and $x \in \{0, 1\}^\mu$ algorithm $\text{Eval}^{\Gamma, \Pi}$ first recovers (via exhaustive search) the preimage $k\|a\|b\|z$ of \mathbf{pk} under Γ for $k, a, b \in \{0, 1\}^\mu$, $z \in \{0, 1\}^{2\mu+\text{pad}}$. Check that $a \neq 0$ in the field $\text{GF}(2^\mu)$. If any check fails then return \perp . Else, next check if $k = 0^\mu$. If so, return $\Pi(a\|b\|\text{setlsb}(ax + b))$, else return $\Pi(a\|b\|ax + b)$.

We now show that there exist permutations Π and Γ such that relative to Π and the lossy function oracle $(\text{Gen}^{\Gamma, \Pi}, \text{Eval}^{\Gamma, \Pi})$, lossy functions exist, but key agreement does not. We will rely on the seminal result by Impagliazzo and Rudich [IR89] showing that no key agreement exists relative to a random permutation. Note that we do not give direct access to Γ — it will only be accessed by the lossy functions oracle and is considered an integral part of it.

The following lemma is the technical core of our results. It says that the partly exponential steps of the lossy-function oracles $\text{Gen}^{\Gamma, \Pi}$ and $\text{Eval}^{\Gamma, \Pi}$ in our construction can be simulated sufficiently close and efficiently through a stateful algorithm Wrap , given only oracle access to Π , even if we filter out the mode for key generation calls. For this we define security experiments as efficient algorithms Game with oracle access to an adversary \mathcal{A} and lossy function oracles $\text{Gen}^{\Gamma, \Pi}, \text{Eval}^{\Gamma, \Pi}, \Pi$ and which produces some output, usually indicating if the adversary has won or not. We note that we can assume for simplicity that \mathcal{A} makes oracle queries to the lossy function oracles and Π via the game only. Algorithm Wrap will be black-box with respect to \mathcal{A} and Game but needs to know the total number $p(\lambda)$ of queries the adversary and the game make to the primitive and the quality level $\alpha(\lambda)$ of the simulation upfront.

Lemma 5.2 (Simulation Lemma). *Let Filter be a deterministic algorithm which for calls $(1^\lambda, \text{mode})$ to $\text{Gen}^{\Gamma, \Pi}$ only outputs 1^λ and leaves any input to calls to $\text{Eval}^{\Gamma, \Pi}$ and to Π unchanged. For any polynomial $p(\lambda)$ and any inverse polynomial $\alpha(\lambda)$ there exists an efficient algorithm Wrap such that for any efficient algorithm \mathcal{A} , any efficient experiment Game making at most $p(\lambda)$ calls to the oracle, the statistical distance between $\text{Game}^{\mathcal{A}, (\text{Gen}^{\Gamma, \Pi}, \text{Eval}^{\Gamma, \Pi}, \Pi)}(1^\lambda)$ and $\text{Game}^{\mathcal{A}, \text{Wrap}^{\text{Gen}^{\Gamma, \Pi}, \Pi} \circ \text{Filter}}(1^\lambda)$ is at most $\alpha(\lambda)$. Furthermore Wrap initially makes a polynomial number of oracle calls to $\text{Gen}^{\Gamma, \Pi}$, but then makes at most two calls to Π for each query.*

In fact, since $\text{Gen}^{\Gamma, \Pi}$ is efficient relative to Γ , and Wrap only makes calls to $\text{Gen}^{\Gamma, \Pi}$ for all values up to a logarithmic length L_0 , we can also write $\text{Wrap}^{\Gamma_{L_0}, \Pi}$ to denote the limited access to the Γ -oracle. We also note that the (local) state of Wrap only consists of such small preimage-image pairs of Γ and Π for such small values (but Wrap later calls Π also about longer inputs).

Proof. The proof strategy is to process queries of Game and \mathcal{A} efficiently given only access to Π , making changes to the oracle gradually, depending on the type of query. The changes will be actually implemented by our stateful algorithm Wrap , and eventually we will add Filter at the end. To do so, we will perform a series of game hops where we change the behavior of the key generation and evaluation oracles. For each game $\text{Game}_1, \text{Game}_2, \dots$ let $\text{Game}_i(\lambda)$ be the randomized output of the game with access to \mathcal{A} . Let $p(\lambda)$ denote the total number of oracle queries the game itself and \mathcal{A} make through the game, and let $\text{Game}_0(\lambda)$ be the original attack of \mathcal{A} with the defined oracles. The final game will then immediately give our algorithm Wrap with the upstream Filter . We give an overview over all the game hops in Figure 5.2.

Game₁. In the first game hops we let Wrap collect all information about very short queries (of length related to L_0) in a list and use this list to answer subsequent queries. Change the oracles as

Game	Gen _{loss}	Gen _{inj}	Eval(pk, x)	Π(x)
Game ₀	pk ← Gen _{loss} ^{Γ, Π} (1 ^λ) return pk	pk ← Gen _{inj} ^{Γ, Π} (1 ^λ) return pk	y ← Eval ^{Γ, Π} (pk, x) return y	Π(x)
Game ₂	(pk, b) ← _{\$} {0, 1} ^{6μ} a ← _{\$} {0, 1} _{≠0^μ} k ← _{\$} {0, 1} _{≠0^μ} st _{pk} ← (k, a, b) return pk	(pk, b) ← _{\$} {0, 1} ^{6μ} a ← _{\$} {0, 1} _{≠0^μ} st _{pk} ← (0 ^μ , a, b) return pk	if st _{pk} = ⊥ k, b ← _{\$} {0, 1} ^{2μ} a ← _{\$} {0, 1} _{≠0^μ} st _{pk} ← (k, a, b) (k, a, b) ← st _{pk} if k = 0 ^μ return Π(pk setlsb(ax + b)) else return Π(pk ax + b)	Π(x)
Game ₃	[...] st _{pk} ← (loss, a, b) [...]	[...] st _{pk} ← (inj, a, b) [...]	if st _{pk} = ∅ b ← _{\$} {0, 1} ^μ a ← _{\$} {0, 1} _{≠0^μ} st _{pk} ← (inj, a, b) (mode, a, b) ← st _{pk} if mode = loss return Π(pk setlsb(ax + b)) else return Π(pk ax + b)	Π(x)
Game ₄	[...] st _{pk} ← (a, b) [...]	[...] st _{pk} ← (a, b) [...]	[...] st _{pk} ← (a, b) a, b ← st _{pk} return Π(pk ax + b)	Π(x)
Game ₅	[...]	[...]	[...] return Π ₁ (pk ax + b)	Π ₁ (x)
Game ₆	pk ← _{\$} {0, 1} ^{5μ} return pk	pk ← _{\$} {0, 1} ^{5μ} return pk	a b ... ← Π ₀ (pk) return Π ₁ (pk ax + b)	Π ₁ (x)
Game ₇	[...]	[...]	return Π ₀ (pk x)	Π ₁ (x)

FIGURE 5.2 — An overview of all the game hops. Note that for simplicity we ignored the modifications related to inputs of length L_0 here, in particular the game hop to Game₁.

follows. Let

$$L_0 := L_0(\lambda) := \lceil \log_2(80\alpha^{-1}(\lambda) \cdot p(\lambda)^2 + p(\lambda)) \rceil.$$

Then our current version of algorithm `Wrap`, upon initialization, queries Π about all inputs of size at most $2L_0$ and stores the list of queries and answers. The reason for using $2L_0$ is that the evaluation algorithm takes as input a key of security parameter λ and some input of size $\mu \approx \lambda/5$, such that we safely cover all evaluations for keys of security size $\lambda \leq L_0$.

Further, for any security parameter less than $2L_0$, our algorithm queries $\text{Gen}^{\Gamma, \Pi}$ for $\lambda 2^{2L_0}$ times; recall that we do not assume that parties have direct access to Γ but only via $\text{Gen}^{\Gamma, \Pi}$. This way, for any valid key, we know that it was created at some point except with probability $(1 - 2^{-2L_0})^{\lambda 2^{2L_0}} \leq 2^{-\lambda}$ and therefore the probability that any key was not generated is at most $2^{L_0} 2^{-\lambda}$, which is negligible.

Further, for every public key, it evaluates $\text{Eval}^{\Gamma, \Pi}$ at $x = 0$ and uses the precomputed list for Π to invert, revealing the corresponding a and b . Note that all of this can be done in polynomial time.

Any subsequent query to $\text{Gen}^{\Gamma, \Pi}$ for security parameter at most L_0 , as well as to $\text{Eval}^{\Gamma, \Pi}$ for a public keys of size at most L_0 (which corresponds to a key for security parameter at most L_0), as well as to Π for inputs of size at most $2L_0$, are answered by looking up all necessary data in the list. If any data is missing, we will return \perp . Note that as long as we do not return \perp , this is only a syntactical change. As returning \perp happens at most with negligible probability over the randomness of Wrap ,

$$\text{SD}(\text{Game}_0, \text{Game}_1) \leq 2^{2L_0} 2^{-\lambda}.$$

From now on we will implicitly assume that queries of short security length up to L_0 are answered genuinely with the help of tables and do not mention this explicitly anymore.

Game₂. In this game, we will stop using the lossy function oracles altogether, and instead introduce a global state for the Wrap algorithm. Note that this state will be shared between all parties having access to the oracles (via Wrap). Now, for every call to $\text{Gen}^{\Gamma, \Pi}$, we do the following: If the key is created in injective mode, Wrap will sample $b \leftarrow \{0, 1\}^\mu$ and $a, k \leftarrow \{0, 1\}^\mu \setminus \{0^\mu\}$, if the key is created in lossy mode, it sets $k = 0^\mu$. Further, it samples a public key $\text{pk} \leftarrow \{0, 1\}^{5\mu + \text{pad}}$, and sets the state $\text{st}_{\text{pk}} \leftarrow (k, a, b)$. Finally it returns pk . Any call to $\text{Eval}^{\Gamma, \Pi}(\text{pk}, x)$ will be handled as follows: First, Wrap checks whether a state for pk exists. If this is not the case, we generate $k, a, b \leftarrow \{0, 1\}^\mu$ (with checking that $a \neq 0$) and save $\text{st}_{\text{pk}} \leftarrow (k, a, b)$. Then, we read $(k, a, b) \leftarrow \text{st}_{\text{pk}}$ from the (possibly just initialized) state and return $\Pi(a \| b \| ax + b)$.

What algorithm Wrap does here can be seen as emulating Γ . However, there are two differences: We do not sample z , and we allow for collisions. The collisions can be of either of two types: Either we sample the same (random) public key $\text{pk} = \text{pk}'$ but for different state values $(k, a, b) \neq (k', a', b')$, or we sample the same values $(k, a, b) = (k', a', b')$ but end up with different public keys $\text{pk} \neq \text{pk}'$. In this case, an algorithm that finds such a collision of size at least μ for $\mu \geq L_0/5$ —smaller values are precomputed and still answered as before—could be able to distinguish the two games. Still, the two games are statistically close since such collisions happen with probability at most $2^{-2L_0/5+1}$ for each pair of generated keys:

$$\text{SD}(\text{Game}_2, \text{Game}_1) \leq 2p(\lambda)^2 \cdot 2^{-2L_0/5+1} \leq \frac{\alpha(\lambda)}{8}$$

Game₃. Next, instead of generating and saving a value k depending on the lossy or injective mode, we just save a label inj or loss for the mode the key was created for. Further, whenever $\text{Eval}^{\Gamma, \Pi}(\text{pk}, x)$ is called on a public key without saved state, i.e., if it has not been created via key generation, then we always label this key as injective.

The only way the adversary is able to recognize the game hop change is because a self-chosen public key, not determined by key generation, will now never be lossy (or will be invalid because $a = 0$). However, any adversarially chosen string of size at least $5\mu \geq L_0$ would only describe a lossy key with probability at most $\frac{1}{2^\mu - p(\lambda)}$ and yield an invalid $a = 0$ with the same probability. Hence, taking into account that the adversary learns at most $p(\lambda)$ values about Γ though genuinely generated keys, and the adversary makes at most $p(\lambda)$ queries, the statistical difference between the

two games is small:

$$\text{SD}(\text{Game}_2, \text{Game}_3) \leq 2p(\lambda) \cdot \frac{1}{2^{-L_0/5+1} - p(\lambda)} \leq \frac{\alpha(\lambda)}{8}.$$

Game₄. Now, we remove the label *inj* or *loss* again. *Wrap* will now, for any call to *Eval*, calculate everything in injective mode.

There are two ways an adversary can distinguish between the two games: Either by inverting Π , e.g., noting that the last bit in the preimage is not as expected, or by finding a pair $x \neq x'$ for a lossy key pk_{loss} such that $\text{Eval}(\text{pk}_{\text{loss}}, x) = \text{Eval}(\text{pk}_{\text{loss}}, x')$ in **Game₃**. Inverting Π (or guessing a and b) only succeeds with probability $\frac{2(p(\lambda)+1)}{2^\mu}$. For the probability of finding a collision, note that viewing the random permutation Π as being lazy sampled (see Appendix A) shows that the answers are chosen independently of the input (except for repeating previous answers), and especially of a, b for any lossy public key of the type considered here. Hence, we can imagine to choose a, b for any possible pairs of inputs only after x, x' have been determined. But then the probability of creating a collision among the $p(\lambda)^2$ many pairs for the same key is at most $\frac{2p(\lambda)^2}{2^\mu}$ for $\mu > L_0/5$. Therefore, the distance between these two games is bounded by

$$\text{SD}(\text{Game}_3, \text{Game}_4) \leq 3(p(\lambda) + p(\lambda)^2) \cdot 2^{-L_0/5+1} \leq \frac{\alpha(\lambda)}{8}.$$

Game₅. We split the random permutation Π to have two oracles. For $\beta \in \{0, 1\}$ and $x \in \{0, 1\}^{5\mu}$, we now define $\Pi_\beta(x) = \Pi(\beta\|x)_{1\dots 5\mu-1}$, i.e., we add a prefix β and drop the last bit. We now replace any use of Π in *Wrap*, including direct queries to Π , by Π_1 .

Would Π_1 be a permutation, this would be a perfect simulation. However, Π_1 is not even injective anymore, but finding a collision is still very unlikely (as random functions are collision resistant). In particular, using once more that we only look at sufficiently large values, the statistical distance of the games is still small:

$$\text{SD}(\text{Game}_4, \text{Game}_5) \leq \frac{2p(\lambda)^2}{2^{5\mu}} \leq \frac{\alpha(\lambda)}{8}.$$

Game₆. Next, we stop using the global state *st* for information about the values related to a public key (except for keys of security parameter at most L_0). The wrapper for *Gen* now only generates a uniformly random pk and returns it. For *Eval* calls, *Wrap* instead calculates $a\|b \leftarrow \Pi_0(\text{pk})$ on the fly. Note that there is a small probability of $2^{-L_0/5+1}$ of $a = 0$, yielding an invalid key. Except for this, since the adversary does not have access to Π_0 , this game otherwise looks completely identical to the adversary:

$$\text{SD}(\text{Game}_5, \text{Game}_6) \leq p(\lambda) \cdot 2^{-L_0/5+1} \leq \frac{\alpha(\lambda)}{8}.$$

Game₇. For our final game, we use Π_0 to evaluate the lossy function:

$$\text{Eval}^\Pi(\text{pk}, x) = \Pi_0(\text{pk}\|x).$$

Note that, as \mathcal{A} has no access to Π_0 , calls to *Eval* in **Game₇** are random for \mathcal{A} . For **Game₆**, calls to *Eval* looks random as long as \mathcal{A} does not invert Π_1 , which happens at most with probability $\frac{2(p(\lambda)+1)}{2^\mu}$. Therefore, the statistical distance between the two games is bound by

$$\text{SD}(\text{Game}_6, \text{Game}_7) \leq 3p(\lambda) \cdot 2^{-2L_0/5+1} \leq \frac{\alpha(\lambda)}{8}.$$

In the final game the algorithm `Wrap` now does not need to save any state related to large public keys, and it behaves identically for the lossy and injective generators. We can therefore safely add our algorithm `Filter`, stripping off the mode before passing key generation requests to `Wrap`. Summing up the statistical distances we obtain a maximal statistical of $\frac{7}{8}\alpha(\lambda) \leq \alpha(\lambda)$ between the original game and the one with our algorithms `Wrap` and `Filter`. \square

We next argue that the simulation lemma allows us to conclude immediately that the function oracle in Definition 5.1 is indeed a lossy function:

Theorem 5.3. *The function in Definition 5.1 is a lossy function for lossiness parameter 2.*

Proof. We first prove the easier structural properties. Clearly, the evaluation can be done efficiently given $\text{Eval}^{\Gamma, \Pi}$ as an oracle (despite $\text{Eval}^{\Gamma, \Pi}$ itself requiring exponential time). The same holds for key generation. We next argue that key generation in mode `inj` leads to an injective function. The reason is that, as Γ is a permutation, k will not be zero by construction when being reconstructed by $\text{Eval}^{\Gamma, \Pi}$, such that subsequent calls to $\text{Eval}^{\Gamma, \Pi}$ will transform different inputs $x \neq x' \in \{0, 1\}^\mu$ to different inputs values $a\|b\|ax + b \neq a\|b\|ax' + b$ for Π . Similarly, $\text{Eval}^{\Gamma, \Pi}$, for a key in mode `loss`, will recover a value $k = 0^\mu$, causing evaluation to set the least significant bit after the hashing step. Then there are exactly two inputs $x \neq x'$ such that $\text{setlsb}(ax + b) = \text{setlsb}(ax' + b)$. Both inputs thus yield the same output. In conclusion, the function is then two-to-one.

The indistinguishability of injective and lossy keys holds by the simulation lemma as follows. Let `Game` be the security experiment which on input 1^λ first picks a challenge bit $\beta \leftarrow \{0, 1\}$ at random, and then queries oracle Gen^Π about $(1^\lambda, \text{inj})$ if $\beta = 0$ resp. $(1^\lambda, \text{loss})$ if $\beta = 1$, obtaining a public key pk . It then initializes the adversary \mathcal{A} on input $(1^\lambda, \text{pk})$, from then on relaying all queries of \mathcal{A} to the oracles $\text{Gen}^{\Gamma, \Pi}$, $\text{Eval}^{\Gamma, \Pi}$, Π and the responses. When \mathcal{A} eventually outputs a guess $\beta' \in \{0, 1\}$ the game checks if $\beta = \beta'$ and outputs 1 if and only if this is the case.

Now assume that there was an adversary successfully attacking our lossy function. In particular, the game outputs 1 with probability at least $\frac{1}{2} + \alpha(\lambda)$ for some inverse polynomial α and infinitely many λ . Then we can simulate the game and this adversary with statistical distance at most $\alpha(\lambda)/2$ for any λ via our algorithms `Wrap` and `Filter` as in the Simulation Lemma 5.2, such that the game still outputs 1 with probability at least $\frac{1}{2} + \alpha(\lambda)/2$ infinitely often. This, however, contradicts the fact that `Wrap` with upstream `Filter` operate completely independent of the key mode, such that \mathcal{A} cannot do better than guessing β with probability $\frac{1}{2}$ in this simulation. \square

5.2 Key Exchange

We next argue that given our oracle-based lossy function in the previous section one cannot build a secure key agreement protocol based only this lossy function (and having also access to Π). The line of reasoning follows the one in the renowned work by Impagliazzo and Rudich [IR89]. They show that one cannot build a secure key agreement protocol between Alice and Bob, given only a random permutation oracle Π . To this end they argue that, if we can find NP-witnesses efficiently, say, if we have access to a PSPACE oracle, then the adversary with oracle access to Π can efficiently compute Alice's key given only a transcript of a protocol run between Alice and Bob (both having access to Π).

We use the same argument as in [IR89] here, noting that according to our Simulation Lemma 5.2 we could replace the lossy function oracle relative to Π by our algorithm Wrap^Π . This, however, requires some care, especially as Wrap does not provide access to the original Π .

We first define (weakly) secure key exchange protocols relative to some oracle (or a set of oracles) \mathcal{O} . We assume that we have an interactive protocol $\langle \text{Alice}^\mathcal{O}, \text{Bob}^\mathcal{O} \rangle$ between two efficient parties, both having access to the oracle \mathcal{O} . The interactive protocol execution for security parameter 1^λ runs the interactive protocol between $\text{Alice}^\mathcal{O}(1^\lambda; z_A)$ for randomness z_A and $\text{Bob}^\mathcal{O}(1^\lambda; z_B)$ with randomness z_B , and we define the output to be a triple $(k_A, T, k_B) \leftarrow \langle \text{Alice}^\mathcal{O}(1^\lambda; z_A), \text{Bob}^\mathcal{O}(1^\lambda; z_B) \rangle$, where k_A is the local key output by Alice, T is the transcript of communication between the two parties, and k_B is the local key output by Bob. When talking about probabilities over this output we refer to the random choice of randomness z_A and z_B .

Note that we define completeness in a slightly non-standard way by allowing the protocol to create non-matching keys with a polynomial (but non-constant) probability, compared to the negligible probability the standard definition would allow. The main motivation for this definition is that it makes our proof easier, but as we will prove a negative result, this relaxed definition makes our result even stronger.

Definition 5.4. A key agreement protocol $\langle \text{Alice}, \text{Bob} \rangle$ relative to an oracle \mathcal{O} is

complete if there exists an at least linear polynomial $p(\lambda)$ such that for all large enough security parameters λ :

$$\Pr[k_A \neq k_B : (k_A, T, k_B) \leftarrow \langle \text{Alice}^\Pi(1^\lambda), \text{Bob}^\mathcal{O}(1^\lambda) \rangle] \leq \frac{1}{p(\lambda)}.$$

secure if for any efficient adversary \mathcal{A} the probability that

$$\Pr[k^* = k_A : (k_A, T, k_B) \leftarrow \langle \text{Alice}^\mathcal{O}(1^\lambda), \text{Bob}^\mathcal{O}(1^\lambda) \rangle, k^* \leftarrow \mathcal{A}^\mathcal{O}(1^\lambda, T)]$$

is negligible.

Theorem 5.5. *There exist random oracles Π and Γ such that relative to $\text{Gen}^{\Gamma, \Pi}$, $\text{Eval}^{\Gamma, \Pi}$, Π and PSPACE, the function oracle $(\text{Gen}^{\Gamma, \Pi}, \text{Eval}^{\Gamma, \Pi})$ from Definition 5.1 is a lossy function, but no construction of secure key agreement from $\text{Gen}^{\Gamma, \Pi}$, $\text{Eval}^{\Gamma, \Pi}$ and Π exists.*

From this theorem and using the two-oracle technique, the following corollary follows directly:

Corollary 5.6. *There exists no fully black-box construction of a secure key agreement protocol from lossy functions.*

Proof (Theorem 5.5). Assume, to the contrary, that a secure key agreement exists relative to these oracles. We first note that it suffices to consider adversaries in the Wrap -based scenario. That is, \mathcal{A} obtains a transcript T generated by the execution of $\text{Alice}^{\text{Wrap}^{\Gamma, \Pi} \circ \text{Filter}}(1^\lambda; z_A)$ with $\text{Bob}^{\text{Wrap}^{\Gamma, \Pi} \circ \text{Filter}}(1^\lambda; z_A)$ where Wrap is initialized with randomness z_W and itself interacts with Π . Note that $\text{Wrap}^\Pi \circ \text{Filter}$ is efficiently computable and only requires local state (holding the oracle tables for small values), so we can interpret the wrapper as part of Alice and Bob without needing any additional communication between the two parties—see Figure 5.3.

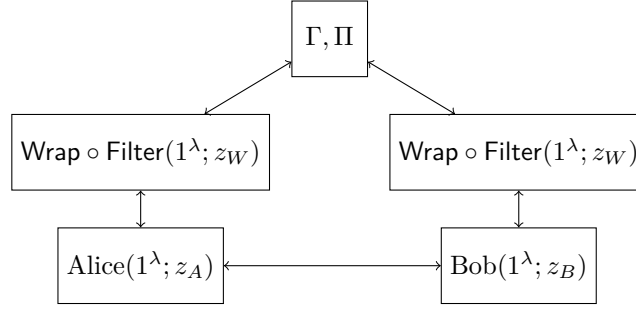


FIGURE 5.3 — The two parties Alice and Bob get access to the $\text{Wrap} \circ \text{Filter}$ algorithm with internal access to the permutations Γ and Π , instead of having access to the lossy function oracles as well as direct access to Π .

We now prove the following two statements about the key agreement protocol in the wrapped mode:

1. For non-constant $\alpha(\lambda)$, the protocol $\langle \text{Alice}^{\text{Wrap}^{\Gamma, \Pi} \circ \text{Filter}}, \text{Bob}^{\text{Wrap}^{\Gamma, \Pi} \circ \text{Filter}} \rangle$ still fulfills the completeness property of the key agreement, i.e., at most with polynomial probability, the keys generated by Alice and Bob differ; and
2. there exists a successful adversary $\mathcal{E}^{\text{Wrap}^{\Gamma, \Pi} \circ \text{Filter}, \text{PSPACE}}$ with additional PSPACE access, that, with at least polynomial probability, recovers the key from the transcript of Alice and Bob.

If we show these two properties, we have derived a contradiction: If there exists a successful adversary against the wrapped version of the protocol, then this adversary must also be successful against the protocol with the original oracles with at most a negligible difference in the success probability – otherwise, this adversary could be used as a distinguisher between the original and the wrapped oracles, contradicting the Simulation Lemma 5.2.

Completeness The first property holds by the Simulation Lemma: Assume there exists a protocol between Alice and Bob such that in the original game, the keys generated differ for at most a polynomial probability $\frac{1}{p(\lambda)}$, while in the case where we replace the access to the oracles by $\text{Wrap}^{\Gamma, \Pi} \circ \text{Filter}$ for some $\alpha(\lambda)$, the keys differ with constant probability $\frac{1}{c_\alpha}$. In such a case, we could—in a thought experiment—modify Alice and Bob to end their protocol by revealing their keys. A distinguisher could now tell from the transcripts whether the keys of the parties differ or match. Such a distinguisher would however now be able to distinguish between the oracles and the wrapper with probability $\frac{1}{c_\alpha} - \frac{1}{p(\lambda)}$, which is larger than $\alpha(\lambda)$ for large enough security parameters, which is a contradiction to the Simulation Lemma.

Attack For the second property, we will argue that the adversary by Impagliazzo and Rudich from their seminal work on key agreement from one-way functions [IR89] works in our case as well. For this, first note that the adversary has access to both Π_1 (by Π -calls to Wrap) and Π_0 (by Eval -calls to Wrap) and Wrap also makes the initial calls to Γ . Combining Γ , Π_0 and Π_1 into a single function we can apply the Impagliazzo-Rudich adversary. Specifically, [IR89, Theorem 6.4] relates the agreement error, denoted ϵ here, to the success probability approximately $1 - 2\epsilon$ of breaking the key agreement protocol. Hence, let $\epsilon(\lambda)$ be the at most polynomial error rate of the original key exchange protocol.

We choose now $\alpha(\lambda)$ sufficiently small such that $\epsilon(\lambda) + \alpha(\lambda)$ is an acceptable error rate for a key exchange, i.e., at most $1/4$. Then this key exchange using the wrapped oracles is a valid key exchange using only our combined random oracle, and therefore, we can use the Impagliazzo-Rudich adversary to recover the key with non-negligible probability.

Fixing the oracles Finally, we have to fix the random permutations Π and Γ such that the Simulation Lemma holds and the Impagliazzo-Rudich attack works. The Impagliazzo-Rudich attack is known to work for all but a zero-measure of random oracles. For the Simulation Lemma, we can use Borel-Cantelli to show that for a one-measure of oracle choices of Γ and Π , no successful adversary can distinguish between the original oracles and the wrapped ones.

To show this, let us fix some game, an adversary \mathcal{A} and an $\alpha(\lambda)$. We now know that there exists a wrapper Wrap^Π such that the statistical distance between the original game and the wrapped game is smaller than $\frac{\alpha(\lambda)}{\lambda^2}$:

$$\text{SD} \left(\text{Game}_{\mathcal{A}}^{\text{Gen}^{\Gamma, \Pi}, \text{Eval}^{\Gamma, \Pi}}, \text{Game}_{\mathcal{A}}^{\text{Wrap}^\Pi \circ \text{Filter}} \right) \leq \frac{\alpha(\lambda)}{\lambda^2}.$$

We can now define $sd_{\mathcal{A}, \Gamma, \Pi}$ as the statistical distance for fixed Γ and Π . Then, we know that the expected value of $sd_{\mathcal{A}, \Gamma, \Pi}$ is bound by

$$\mathbb{E}_{\Gamma, \Pi} [sd_{\mathcal{A}, \Gamma, \Pi}] \leq \frac{\alpha(\lambda)}{\lambda^2}.$$

Using the Markov inequality, we get

$$\Pr [sd_{\mathcal{A}, \Gamma, \Pi} \geq \alpha(\lambda)] \leq \frac{1}{\lambda^2}.$$

Let us now denote by E_λ the event that for security parameter λ , the statistical distance $sd_{\mathcal{A}, \Gamma, \Pi}$ is at least $\alpha(\lambda)$. Since the hyperharmonic series converges, we have that

$$\sum_{\lambda=1}^{\infty} \Pr [E_\lambda] < \infty$$

and we can therefore apply Borel-Cantelli to get that the statistical distance is not bound by $\alpha(\lambda)$ for infinitely many security parameters only happens for a zero measure of oracles Γ and Π :

$$\Pr_{\Gamma, \Pi} \left[\bigwedge_{k=1}^{\infty} \bigvee_{\lambda \geq k} E_\lambda \right] = 0.$$

As there exist only countable many (uniform) adversaries, there exists a one-measure of oracles Γ, Π such that the Simulation Lemma holds for any adversary. Therefore, there clearly exist some oracles Γ, Π such that the Simulation Lemma holds, while the Impagliazzo-Rudich attack is successful. \square

5.3 ELFs

We will show next that our result can also be extended to show that no fully black-box construction of key agreement from *extremely* lossy functions is possible. However, we are only able to show a slightly weaker result: In our separation, we only consider constructions that access the extremely lossy function on the same security parameter as used in the key agreement protocol. We call such

constructions *security-level-preserving*. This leaves the theoretic possibility of building key agreement from extremely lossy functions of (significantly) smaller security parameters. At the same time it simplifies the proof of the Simulation Lemma for this case significantly since we can omit the step where `Wrap` samples Γ for all small inputs, and we can immediately work with the common negligible terms.

We start by defining an ELF oracle. In general, the oracle is quite similar to our lossy function oracle. Especially, we still distinguish between an injective and a lossy mode, and make sure that any key sampled without a call to the $\text{Gen}_{\text{ELF}}^{\Gamma, \Pi}$ oracle will be injective with overwhelming probability. For the lossy mode, we now of course have to save the parameter r in the public key. Instead of using `setlsb` to lose one bit of information, we take the result of $ax + b$ (calculated in $GF(2^\mu)$) modulo r (calculated on the integers) to allow for the more fine-grained lossiness that is required by ELFs.

Definition 5.7 (Extremely Lossy Function Oracle). Let Π, Γ be permutation oracles with

$$\Pi, \Gamma : \{0, 1\}^\lambda \rightarrow \{0, 1\}^\lambda$$

for all λ . Let $\mu = \mu(\lambda) = \lfloor (\lambda - 2)/5 \rfloor$ and $\text{pad} = \text{pad}(\lambda) = \lambda - 2 - 5\mu$ defines the length that the rounding-off loses to $\lambda - 2$ in total (such that $\text{pad} \in \{0, 1, 2, 3, 4\}$). Define the extremely lossy function $(\text{Gen}_{\text{ELF}}^{\Gamma, \Pi}, \text{Eval}_{\text{ELF}}^{\Gamma, \Pi})$ with input length $\text{in}(\lambda) = \mu(\lambda)$ relative to Γ and Π now as follows:

Key Generation: The oracle $\text{Gen}_{\text{ELF}}^{\Gamma, \Pi}$, on input 1^λ and mode r , picks random $b \leftarrow_{\$} \{0, 1\}^\mu$, $z \leftarrow_{\$} \{0, 1\}^{\mu + \text{pad}}$ and random $a, k \leftarrow_{\$} \{0, 1\}^\mu \setminus \{0^\mu\}$. For mode $r = 2^{\text{in}(\lambda)}$ the algorithm returns $\Gamma(k \| a \| b \| r \| z)$. For mode $r < 2^{\text{in}(\lambda)}$ the algorithm returns $\Gamma(0^\mu \| a \| b \| r \| z)$ instead.

Evaluation: On input $\text{pk} \in \{0, 1\}^\lambda$ and $x \in \{0, 1\}^\mu$ algorithm $\text{Eval}_{\text{ELF}}^{\Gamma, \Pi}$ first recovers (via exhaustive search) the preimage $k \| a \| b \| r \| z$ of pk under Γ for $k, a, b, r \in \{0, 1\}^\mu$, $z \in \{0, 1\}^{\mu + \text{pad}}$. Check that $a \neq 0$ in the field $GF(2^\mu)$. If any check fails then return \perp . Else, next check if $k = 0^m$. If so, return $\Pi(a \| b \| (ax + b \bmod r))$, else return $\Pi(a \| b \| ax + b)$.

We can now formulate versions of Theorem 5.5 and Corollary 5.6 for the extremely lossy case.

Theorem 5.8. *There exist random oracles Π and Γ such that relative to $\text{Gen}_{\text{ELF}}^{\Gamma, \Pi}$, $\text{Eval}_{\text{ELF}}^{\Gamma, \Pi}$, Π and PSPACE, the extremely lossy function oracle $(\text{Gen}_{\text{ELF}}^{\Gamma, \Pi}, \text{Eval}_{\text{ELF}}^{\Gamma, \Pi})$ from Definition 5.7 is indeed an ELF, but no security-level-preserving construction of secure key agreement from $\text{Gen}_{\text{ELF}}^{\Gamma, \Pi}, \text{Eval}_{\text{ELF}}^{\Gamma, \Pi}$ and Π exists.*

Corollary 5.9. *There exists no fully black-box security-level-preserving construction of a secure key agreement protocol from extremely lossy functions.*

Proving Theorem 5.8 only needs minor modifications of the proof of Theorem 5.5 to go through. Indeed, the only real difference lies in a modified Simulation Lemma for ELFs, which we will formulate next, together with a proof sketch that explains where differences arrive in the proof compared to the original Simulation Lemma. To stay as close to the previous proof as possible, we will continue to distinguish between an injective generator $\text{Gen}_{\text{inj}}(1^\lambda)$ and a lossy generator $\text{Gen}_{\text{loss}}(1^\lambda, r)$, where the latter also receives the parameter r . Figure 5.4 provides an overview of the modified game hops.

Lemma 5.10 (Simulation Lemma (ELFs)). *Let `Filter` be a deterministic algorithm which for calls $(1^\lambda, \text{mode})$ to $\text{Gen}_{\text{ELF}}^{\Gamma, \Pi}$ only outputs 1^λ and leaves any input to calls to $\text{Eval}_{\text{ELF}}^{\Gamma, \Pi}$ and to Π unchanged.*

Game	Gen _{loss} (r)	Gen _{inj}	Eval(pk, x)	$\Pi(x)$
Game ₀	pk \leftarrow Gen _{loss} ^{Γ, Π} ($1^\lambda, r$) return pk	pk \leftarrow Gen _{inj} ^{Γ, Π} (1^λ) return pk	$y \leftarrow$ Eval _{ELF} ^{Γ, Π} (pk, x) return y	$\Pi(x)$
Game ₂	(pk, b) $\leftarrow_{\$}$ $\{0, 1\}^{6\mu}$ $a \leftarrow_{\$}$ $\{0, 1\}_{\neq 0^\mu}^\mu$ $k \leftarrow_{\$}$ $\{0, 1\}_{\neq 0^\mu}^\mu$ $\text{st}_{\text{pk}} \leftarrow (k, a, b, r)$ return pk	(pk, b) $\leftarrow_{\$}$ $\{0, 1\}^{6\mu}$ $a \leftarrow_{\$}$ $\{0, 1\}_{\neq 0^\mu}^\mu$ $\text{st}_{\text{pk}} \leftarrow (0^\mu, a, b, 0^\mu)$ return pk	if $\text{st}_{\text{pk}} = \emptyset$ $k, b, r \leftarrow_{\$}$ $\{0, 1\}^{3\mu}$ $a \leftarrow_{\$}$ $\{0, 1\}_{\neq 0^\mu}^\mu$ $\text{st}_{\text{pk}} \leftarrow (k, a, b, r)$ $k, a, b, r \leftarrow \text{st}_{\text{pk}}$ if $k = 0^\mu$ return $\Pi(\text{pk} \parallel (ax + b) \bmod r)$ else return $\Pi(\text{pk} \parallel ax + b)$	$\Pi(x)$
Game ₃	[...] $\text{st}_{\text{pk}} \leftarrow (\text{loss}, a, b, r)$ [...]	[...] $\text{st}_{\text{pk}} \leftarrow (\text{inj}, a, b, 0^\mu)$ [...]	if $\text{st}_{\text{pk}} = \emptyset$ $b \leftarrow_{\$}$ $\{0, 1\}^\mu$ $a \leftarrow_{\$}$ $\{0, 1\}_{\neq 0^\mu}^\mu$ $\text{st}_{\text{pk}} \leftarrow (\text{inj}, a, b, 0^\mu)$ $\text{mode}, a, b, r \leftarrow \text{st}_{\text{pk}}$ if mode = loss return $\Pi(\text{pk} \parallel (ax + b) \bmod r)$ else return $\Pi(\text{pk} \parallel ax + b)$	$\Pi(x)$
Game ₄	[...] $\text{st}_{\text{pk}} \leftarrow (a, b)$ [...]	[...] $\text{st}_{\text{pk}} \leftarrow (a, b)$ [...]	[...] $\text{st}_{\text{pk}} \leftarrow (a, b)$ $a, b \leftarrow \text{st}_{\text{pk}}$ return $\Pi(\text{pk} \parallel ax + b)$	$\Pi(x)$
Game ₅	[...]	[...]	[...] return $\Pi_1(\text{pk} \parallel ax + b)$	$\Pi_1(x)$
Game ₆	pk $\leftarrow_{\$}$ $\{0, 1\}^{5\mu}$ return pk	pk $\leftarrow_{\$}$ $\{0, 1\}^{5\mu}$ return pk	$a \parallel b \parallel \dots \leftarrow \Pi_0(\text{pk})$ return $\Pi_1(\text{pk} \parallel ax + b)$	$\Pi_1(x)$
Game ₇	[...]	[...]	return $\Pi_0(\text{pk} \parallel x)$	$\Pi_1(x)$

FIGURE 5.4 — An overview of all the game hops for the Simulation Lemma, ELF version.

There exists an efficient algorithm *Wrap* such that for any polynomials p and d' there exists a polynomial q such that for any adversary \mathcal{A} which makes at most $p(\lambda)$ queries to the oracles, any efficient experiment *Game* making calls to the Gen_{ELF} ^{Γ, Π} oracle with $r > q(\lambda)$ the distinguishing advantage between $\text{Game}^{\mathcal{A}, (\text{Gen}_{\text{ELF}}^{\Gamma, \Pi}, \text{Eval}_{\text{ELF}}^{\Gamma, \Pi}, \Pi)}(1^\lambda)$ and $\text{Game}^{\mathcal{A}, \text{Wrap}^\Pi \circ \text{Filter}}$ is at most $\frac{1}{d'(\lambda)}$ for sufficiently large λ . Furthermore *Wrap* makes at most two calls to Π for each query.

Proof (Sketch). We will now describe how the game hops differ from the proof of Lemma 5.2, and how these changes affect the advantage of the distinguisher. Note that allowing only access to the ELF oracle at the current security parameter allows us to argue that differences between game hops are negligible, instead of having to give a concrete bound.

Game₁. stays identical to **Game₀** – as we only allow access to the ELF oracle at the current security level, precomputing all values smaller than some L_0 is not necessary here.

Game₂. introduces changes similar to **Game₂** in Lemma 5.2 – however, we now of course also have to save the parameter r in the state. Again, the only notable difference to the distinguisher is that we sample pk independently of the public key parameters and therefore, collisions might happen more often. However, the probability for this is clearly negligible:

$$\text{SD}(\text{Game}_1, \text{Game}_2) \leq \text{negl}(\lambda)$$

Game₃. replaces k with a label inj or loss . Again, the only noticeable difference is that keys sampled without calling Gen_{inj} or Gen_{loss} will now always be injective, while they are lossy with probability $2^{-\mu}$ in **Game₂**, yielding only a negligible difference between the two games however.

$$\text{SD}(\text{Game}_2, \text{Game}_3) \leq \text{negl}(\lambda)$$

Game₄. is the game where we start to always evaluate in injective mode. There are two options a distinguisher might distinguish between the two games: Either by inverting Π , or by finding a collision for a lossy key. Inverting Π only happens with probability $\frac{2(p(\lambda)+1)}{2^\mu}$, while finding a collision happens with probability $\frac{2p(\lambda)^2}{r}$. Let $d(\lambda) = \frac{d'(\lambda)}{2}$ be the advantage we want to allow for the distinguisher in this game hop. Choosing $q(\lambda) = 4p(\lambda)^2 d(\lambda)$ for the bound on r of the ELF, we get

$$\text{Adv}_{\mathcal{A}}^{\text{Game}_3, \text{Game}_4}(\lambda) \leq \frac{1}{d(\lambda)}$$

Game₄ is now identical to **Game₄** in the proof of Lemma 5.2 (except for the different handling of calls to security parameters smaller than L_0). Therefore, all game hops up to **Game₇** are identical to the ones in the proof of Lemma 5.2, with the statistical difference being negligible for all of them. Therefore, the overall advantage of an distinguisher is bounded by $\frac{1}{d(\lambda)} + \text{negl}(\lambda) \leq \frac{1}{d'(\lambda)}$ for large enough security parameters λ . □

Let $\langle \text{Alice}^{\text{Gen}_{\text{ELF}}^r, \text{Eval}_{\text{ELF}}^r, \Pi}, \text{Bob}^{\text{Gen}_{\text{ELF}}^r, \text{Eval}_{\text{ELF}}^r, \Pi} \rangle$ be some candidate key agreement protocol with completeness error $\frac{1}{\epsilon(\lambda)} < \frac{1}{8}$ that makes at most $p(\lambda)$ queries in sum, and let $\frac{1}{d'(\lambda)} < \frac{1}{8}$ be the advantage bound for any adversary against the key agreement we are trying to reach.

To determine the correct parameters for the ELF oracle, we need to know how many queries the Impagliazzo-Rudich adversary makes against the transcript of the wrapped version of the protocol $\langle \text{Alice}^{\text{Wrap}^\Pi \circ \text{Filter}}, \text{Bob}^{\text{Wrap}^\Pi \circ \text{Filter}} \rangle$, which depends on the number of queries of the protocol. Note that we know that Wrap^Π makes at most two queries to Π for each internal query of Alice or Bob, so we know that the wrapped version makes at most $2p(\lambda)$ queries to Π . Let $p'(\lambda)$ be the number of queries needed by the Impagliazzo-Rudich protocol.

First, we have to show that completeness still holds for the wrapped version of the protocol. The wrapped protocol has an error rate of at most $\frac{1}{\epsilon'} < \frac{1}{\epsilon} + \frac{1}{d'} \leq \frac{1}{4}$, as otherwise, we would have a successful distinguisher for the Simulation Lemma. Further, as the error rate $\frac{1}{\epsilon'}$ is smaller than $\frac{1}{4}$, we know that Impagliazzo-Rudich will have a success probability of at least $\frac{1}{2}$.

Further, we know from the Simulation Lemma that we need $d(\lambda) = \frac{d'(\lambda)}{2}$ for it to hold. Therefore, we set the bound for r in the ELF oracle to $q(\lambda) = 4p'(\lambda)^2 d(\lambda)$. Now, the Impagliazzo-Rudich attack

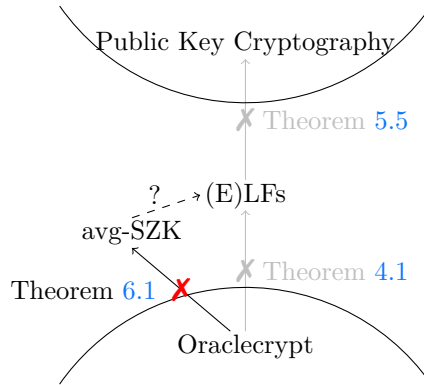


FIGURE 6.5 — We show an oracle separation between Oraclecrypt and average-case SZK as well. The question whether lossy functions can be built from average-case SZK is still open.

has to be successful for the original protocol with polynomial probability $\frac{1}{d^r}$, as otherwise, there would be a distinguisher for the Simulation Lemma with advantage $\frac{1}{2} - \text{negl}(\lambda) > \frac{1}{d^r(\lambda)}$. Fixing oracles Π, Γ such that $(\text{Gen}_{\text{ELF}}^{\Gamma, \Pi}, \text{Eval}_{\text{ELF}}^{\Gamma, \Pi})$ is an ELF, while the Impagliazzo-Rudich attack is successful yields the Theorem.

6 Relationship of Lossy Functions to Statistical Zero-Knowledge

The complexity class (average-case) SZK, introduced by Goldwasser, Micali and Rackoff [GMR85], contains all languages that can be proven by a statistical zero-knowledge proof, and is often characterized by its complete promise problem (average-case) Statistical Distance [SV03]. Hardness of Statistical Zero-Knowledge follows from a number of algebraic assumptions like Discrete Logarithm [GK93] and lattice problems [MV03] and the existence of some Minicrypt primitives like one-way functions [Ost91] and distributional collision resistant hash functions [KY18] follow from hard problems in SZK – it is not known to follow from any Minicrypt assumptions, however, and for some, e.g., collision-resistant hash functions, there exist black-box separations [BD19].

Therefore, average-case hard problems in SZK seem to be a natural candidate for a non-public key assumption to build lossy functions from. Intuitively, one can see similarities between lossy functions and statistical distance: Both are, in a sense, promise problems, if one looks at the image size of a lossy function with a large gap between the injective mode and the lossy mode. Further, it is known that hard problems in SZK follow from lossy functions (this seems to be folklore knowledge – we give a proof for this fact in Appendix B).

Note that a construction of lossy functions would also be interesting from a different perspective: As collision-resistant hash functions can be built from sufficiently lossy functions, a construction of (sufficiently) lossy functions from average-case SZK hardness would mean that collision resistance follows from average-case SZK hardness. However, right now, this is only known for *distributional* collision resistance, a weaker primitive [KY18].

Alas, we are unable to either give a construction of a lossy function from a hard-on-average statistical zero-knowledge problem or to prove a black-box impossibility result between the two, leaving this as an interesting open question for future work. Instead, we give a lower bound on the

needed assumptions for hard-on-average problems in SZK by showing that no Oraclecrypt primitive can be used in a black-box way to construct a hard-on-average problem in SZK – this serves as hint that indeed SZK is an interesting class of problems to look at for building lossy functions, but the result might also be interesting independently.

Note some Oraclecrypt primitives, such a separation already exists: For example, Bitansky and Degwekar give an oracle separation between collision-resistant hash functions and (even worst-case) hard problems in SZK. However, this result uses a Simon-style oracle separation (using a *break*-oracle that depends on the random oracle), which means that the result is specific to the primitive and does not easily generalize to all Oraclecrypt primitives.

Theorem 6.1. *There exists no black-box construction of an hard-on-average problem in SZK from any Oraclecrypt primitive.*

Our proof techniques is quite similar to Chapter 4: First, we will reuse the oracles \mathcal{O} and $\text{PSPACE}^{\mathcal{O}'}$. As average-case statistical distance is complete for average-case SZK, we will assume there exists an hard-on-average statistical distance problem relative to these random oracles. We will then calculate the heavy queries of the circuits produced by the statistical distance problem and show that the heavy queries are sufficient to decide whether the circuits are statistically far from each other or not, yielding a contradiction to the assumed hardness-on-average of statistical distance. Fixing \mathcal{O} and $\text{PSPACE}^{\mathcal{O}'}$ to specific oracles then yields the theorem.

We will start by defining average-case statistical distance:

Definition 6.2 (Average-case Statistical Distance). An average-case statistical distance problem is characterized by a pair of probabilistic polynomial-time algorithms $D_Y^{\mathcal{O}}(1^\lambda)$ and $D_N^{\mathcal{O}}(1^\lambda)$ each producing pairs of circuits $C_0^{\mathcal{O}}, C_1^{\mathcal{O}} : \{0, 1\}^\lambda \rightarrow \{0, 1\}^\lambda$ such that:

$$\begin{aligned} \forall (C_0^{\mathcal{O}}, C_1^{\mathcal{O}}) \leftarrow D_Y^{\mathcal{O}}(1^\lambda) : \text{SD}(C_0^{\mathcal{O}}, C_1^{\mathcal{O}}) &\geq \frac{2}{3}, \\ \forall (C_0^{\mathcal{O}}, C_1^{\mathcal{O}}) \leftarrow D_N^{\mathcal{O}}(1^\lambda) : \text{SD}(C_0^{\mathcal{O}}, C_1^{\mathcal{O}}) &\leq \frac{1}{3}. \end{aligned}$$

Statistical distance is considered hard-on-average if there exists such a pair (D_Y, D_N) such that no polynomial-time adversary \mathcal{A} has more than negligible advantage in distinguishing Yes-instances from No-instances:

$$\forall \mathcal{A} \forall \lambda \in \mathbb{N} : \Pr_{b \leftarrow \{Y, N\}; (C_0, C_1) \leftarrow D_b(1^\lambda)} [\mathcal{A}(1^\lambda, C_0, C_1) = b] \leq \frac{1}{2} + \text{negl}(\lambda).$$

Lemma 6.3. *Let $D_{\mathcal{Y}}^{\mathcal{O}}(1^\lambda)$ be a polynomial-time sampler of instances of the statistical distance problem for either Yes- or No-instances, and let $(C_0^{\mathcal{O}}, C_1^{\mathcal{O}}) \leftarrow D_{\mathcal{Y}}^{\mathcal{O}}(1^\lambda)$. Then we can compute in probabilistic polynomial-time (in λ) a set \hat{Q}_H which contains all $\frac{1}{10k}$ -heavy queries of both $C_0^{\mathcal{O}}$ and $C_1^{\mathcal{O}}$ to \mathcal{O} with overwhelming probability.*

Again, k denotes the maximum number of queries of either $D_{\mathcal{Y}}^{\mathcal{O}}$ or $C_0^{\mathcal{O}}$ or $C_1^{\mathcal{O}}$, whichever is highest. The proof is identical to the one of Lemma 4.4, except that we now have to find the heavy queries for two circuits instead of one function.

Next, we want to show that we can approximate the statistical distance of $C_0^{\mathcal{O}}$ and $C_1^{\mathcal{O}}$ with the heavy queries and by using the $\text{PSPACE}^{\mathcal{O}'}$ oracle. First, note that we can calculate the statistical difference of $C_0^{\mathcal{O}'}$ and $C_1^{\mathcal{O}'}$ efficiently using the $\text{PSPACE}^{\mathcal{O}'}$ oracle – see Figure 6.6 for an algorithm.

$$\begin{array}{l}
\text{SD}(C_0^{\mathcal{O}'}, C_1^{\mathcal{O}'}) \\
\hline
d \leftarrow 0 \\
\forall y \in \text{im}(C_0^{\mathcal{O}'} \cup C_1^{\mathcal{O}'}): \\
\quad s_0 \leftarrow |\{x : C_0^{\mathcal{O}'}(x) = y\}| \\
\quad s_1 \leftarrow |\{x : C_1^{\mathcal{O}'}(x) = y\}| \\
\quad d \leftarrow d + |s_0 - s_1| \\
\mathbf{return} \frac{1}{2} \cdot \frac{d}{2^n}
\end{array}$$

FIGURE 6.6 — An algorithm to calculate the statistical distance of two oracle-aided circuits using the $\text{PSPACE}^{\mathcal{O}'}$ oracle. We go through all possible images of the two circuits, then check for each image how many preimages it has under either circuit. We then add up the difference of the number of preimages between the circuits, and finally return the normalized sum. As we do not use more than polynomial space at any point in time, this algorithm is indeed in $\text{PSPACE}^{\mathcal{O}'}$.

Similar to Chapter 4, we observe that the circuit sampler $D_Y^{\mathcal{O}}$ (or $D_N^{\mathcal{O}}$) only makes polynomially many queries to the oracle \mathcal{O} and, by correctness, has therefore to produce circuits $C_0^{\tilde{\mathcal{O}}}, C_1^{\tilde{\mathcal{O}}}$ for any extension $\tilde{\mathcal{O}}$ of the polynomially-many queries.

Lemma 6.4. *Let $D_{\gamma}^{\mathcal{O}}(1^\lambda)$ be a polynomial-time sampler of instances of the statistical distance problem for either Yes- or No-instances, and let $(C_0^{\mathcal{O}}, C_1^{\mathcal{O}}) \leftarrow D_{\gamma}^{\mathcal{O}}(1^\lambda)$. Let \hat{Q}_H be the heavy queries of both $C_0^{\mathcal{O}}$ and $C_1^{\mathcal{O}}$. Then, we can distinguish whether $(C_0^{\mathcal{O}}, C_1^{\mathcal{O}})$ are far or close with high probability using $\text{PSPACE}^{\mathcal{O}'}$.*

Proof. We distinguish the two cases as follows: We define $\mathcal{O}'_{\hat{Q}_H}$ as the random oracle \mathcal{O}' modified to match \mathcal{O} for all queries in \hat{Q}_H . Now, we use the $\text{PSPACE}^{\mathcal{O}'}$ oracle to calculate the statistical distance of $C_0^{\mathcal{O}'_{\hat{Q}_H}}$ and $C_1^{\mathcal{O}'_{\hat{Q}_H}}$ using the $\text{PSPACE}^{\mathcal{O}'}$ oracle (note that by definition, \hat{Q}_H is of polynomial size and therefore can be passed to the $\text{PSPACE}^{\mathcal{O}'}$ oracle). If the statistical distance is at least $\frac{1}{2}$, we claim it was generated by $D_{\gamma}^{\mathcal{O}}$, otherwise, we claim it was created by $D_N^{\mathcal{O}}$.

Now, of course, we still need to show that if the two circuits were far for the original oracle, they are still far for our new oracle – and that if they were close for the original oracle, they are still close for our new oracle. We know this must true for all changes in the oracle that were not queried by the distribution sampler D_{γ} , and we made sure that no heavy queries are changed between the two oracles, so the only thing that might influence the closeness or farness of the circuits are the polynomially many queries done by D_{γ} that are not heavy queries. We will now show that while these queries might push the statistical difference beyond the $\frac{1}{3}$ or $\frac{2}{3}$ bound, they will not change the statistical difference enough to push the statistical difference beyond the $\frac{1}{2}$ bound, meaning that our distinguisher is still able to tell them apart.

Let k' denote the number of these non-heavy queries asked by the distribution sampler. By definition of k , we know that $k' \leq k$. Each of the queries is non-heavy, which means that only a $\frac{1}{10k}$ -fraction of inputs generates different outputs if one of these queries is changed between the oracles. Each changed output changes the statistical difference by at most $\frac{1}{2^\lambda}$. Therefore, each changed non-heavy query changes the statistical distance by at most $\frac{1}{10k}$, and all non-heavy queries asked by the distribution sampler change the statistical distance by at most $\frac{k'}{10k} \leq \frac{1}{10}$. However, as

$\frac{1}{3} + \frac{1}{10} < \frac{1}{2}$ (and $\frac{2}{3} - \frac{1}{10} > \frac{1}{2}$), the changes do not push the statistical distance beyond the $\frac{1}{2}$ bound, and our distinguisher is therefore successful. \square

Proof (of Theorem 6.1). The proof works again in the same vein as the proof of Theorem 4.1: Lemma 6.3 and Lemma 6.4 show that with overwhelming probability, hard-on-average SZK problems do not exist relative to \mathcal{O} and $\text{PSPACE}^{\mathcal{O}'}$, over the probability of choosing \mathcal{O} and \mathcal{O}' . Now, we just have to show that there exists a fixed pair of oracles such that our Oraclecrypt primitive exists, but hard-on-average SZK problems do not – which we can easily do using the methods described in Section 4.4. This separation then proves that no fully-black-box construction of a hard-on-average SZK problem from any Oraclecrypt primitive may exist. \square

Acknowledgments

We thank the anonymous reviewers for valuable comments.

Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – SFB 1119 – 236615297 and by the German Federal Ministry of Education and Research and the Hessian Ministry of Higher Education, Research, Science and the Arts within their joint support of the National Research Center for Applied Cybersecurity ATHENE.

References

- [ACH20] Thomas Agrikola, Geoffroy Couteau, and Dennis Hofheinz. “The Usefulness of Sparsifiable Inputs: How to Avoid Subexponential iO”. In: *PKC 2020, Part I*. Ed. by Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas. Vol. 12110. LNCS. Springer, Cham, May 2020, pp. 187–219. DOI: [10.1007/978-3-030-45374-9_7](https://doi.org/10.1007/978-3-030-45374-9_7).
- [BBF13] Paul Baecker, Christina Brzuska, and Marc Fischlin. “Notions of Black-Box Reductions, Revisited”. In: *ASIACRYPT 2013, Part I*. Ed. by Kazue Sako and Palash Sarkar. Vol. 8269. LNCS. Springer, Berlin, Heidelberg, Dec. 2013, pp. 296–315. DOI: [10.1007/978-3-642-42033-7_16](https://doi.org/10.1007/978-3-642-42033-7_16).
- [BCEKM23] Chris Brzuska, Geoffroy Couteau, Christoph Egger, Pihla Karanko, and Pierre Meyer. *Instantiating the Hash-Then-Evaluate Paradigm: Strengthening PRFs, PCFs, and OPRFs*. Cryptology ePrint Archive, Report 2023/1145. 2023. URL: <https://eprint.iacr.org/2023/1145>.
- [BD19] Nir Bitansky and Akshay Degwekar. “On the Complexity of Collision Resistant Hash Functions: New and Old Black-Box Separations”. In: *TCC 2019, Part I*. Ed. by Dennis Hofheinz and Alon Rosen. Vol. 11891. LNCS. Springer, Cham, Dec. 2019, pp. 422–450. DOI: [10.1007/978-3-030-36030-6_17](https://doi.org/10.1007/978-3-030-36030-6_17).
- [BDV17] Nir Bitansky, Akshay Degwekar, and Vinod Vaikuntanathan. “Structure vs. Hardness Through the Obfuscation Lens”. In: *CRYPTO 2017, Part I*. Ed. by Jonathan Katz and Hovav Shacham. Vol. 10401. LNCS. Springer, Cham, Aug. 2017, pp. 696–723. DOI: [10.1007/978-3-319-63688-7_23](https://doi.org/10.1007/978-3-319-63688-7_23).

- [BFM14] Christina Brzuska, Pooya Farshim, and Arno Mittelbach. “Indistinguishability Obfuscation and UCEs: The Case of Computationally Unpredictable Sources”. In: *CRYPTO 2014, Part I*. Ed. by Juan A. Garay and Rosario Gennaro. Vol. 8616. LNCS. Springer, Berlin, Heidelberg, Aug. 2014, pp. 188–205. DOI: [10.1007/978-3-662-44371-2_11](https://doi.org/10.1007/978-3-662-44371-2_11).
- [BHK11] Mark Braverman, Avinatan Hassidim, and Yael Tauman Kalai. “Leaky Pseudo-Entropy Functions”. In: *Innovations in Computer Science - ICS 2011, Tsinghua University, Beijing, China, January 7-9, 2011. Proceedings*. Ed. by Bernard Chazelle. Tsinghua University Press, 2011, pp. 353–366. ISBN: 978-7-302-24517-9. URL: <http://conference.iis.tsinghua.edu.cn/ICS2011/content/papers/17.html>.
- [BHK13] Mihir Bellare, Viet Tung Hoang, and Sriram Keelveedhi. “Instantiating Random Oracles via UCEs”. In: *CRYPTO 2013, Part II*. Ed. by Ran Canetti and Juan A. Garay. Vol. 8043. LNCS. Springer, Berlin, Heidelberg, Aug. 2013, pp. 398–415. DOI: [10.1007/978-3-642-40084-1_23](https://doi.org/10.1007/978-3-642-40084-1_23).
- [BR93] Mihir Bellare and Phillip Rogaway. “Random Oracles are Practical: A Paradigm for Designing Efficient Protocols”. In: *ACM CCS 93*. Ed. by Dorothy E. Denning, Raymond Pyle, Ravi Ganesan, Ravi S. Sandhu, and Victoria Ashby. ACM Press, Nov. 1993, pp. 62–73. DOI: [10.1145/168588.168596](https://doi.org/10.1145/168588.168596).
- [BST16] Mihir Bellare, Igors Stepanovs, and Stefano Tessaro. “Contention in Cryptoland: Obfuscation, Leakage and UCE”. In: *TCC 2016-A, Part II*. Ed. by Eyal Kushilevitz and Tal Malkin. Vol. 9563. LNCS. Springer, Berlin, Heidelberg, Jan. 2016, pp. 542–564. DOI: [10.1007/978-3-662-49099-0_20](https://doi.org/10.1007/978-3-662-49099-0_20).
- [CCR16] Ran Canetti, Yilei Chen, and Leonid Reyzin. “On the Correlation Intractability of Obfuscated Pseudorandom Functions”. In: *TCC 2016-A, Part I*. Ed. by Eyal Kushilevitz and Tal Malkin. Vol. 9562. LNCS. Springer, Berlin, Heidelberg, Jan. 2016, pp. 389–415. DOI: [10.1007/978-3-662-49096-9_17](https://doi.org/10.1007/978-3-662-49096-9_17).
- [CGH98] Ran Canetti, Oded Goldreich, and Shai Halevi. “The Random Oracle Methodology, Revisited (Preliminary Version)”. In: *30th ACM STOC*. ACM Press, May 1998, pp. 209–218. DOI: [10.1145/276698.276741](https://doi.org/10.1145/276698.276741).
- [Dur64] Richard Durstenfeld. “Algorithm 235: Random permutation”. In: *Commun. ACM* 7.7 (1964), p. 420.
- [DVW20] Yevgeniy Dodis, Vinod Vaikuntanathan, and Daniel Wichs. “Extracting Randomness from Extractor-Dependent Sources”. In: *EUROCRYPT 2020, Part I*. Ed. by Anne Canteaut and Yuval Ishai. Vol. 12105. LNCS. Springer, Cham, May 2020, pp. 313–342. DOI: [10.1007/978-3-030-45721-1_12](https://doi.org/10.1007/978-3-030-45721-1_12).
- [GHMM18] Sanjam Garg, Mohammad Hajiabadi, Mohammad Mahmoody, and Ameer Mohammed. “Limits on the Power of Garbling Techniques for Public-Key Encryption”. In: *CRYPTO 2018, Part III*. Ed. by Hovav Shacham and Alexandra Boldyreva. Vol. 10993. LNCS. Springer, Cham, Aug. 2018, pp. 335–364. DOI: [10.1007/978-3-319-96878-0_12](https://doi.org/10.1007/978-3-319-96878-0_12).

- [GK93] Oded Goldreich and Eyal Kushilevitz. “A Perfect Zero-Knowledge Proof System for a Problem Equivalent to the Discrete Logarithm”. In: *Journal of Cryptology* 6.2 (June 1993), pp. 97–116. DOI: [10.1007/BF02620137](https://doi.org/10.1007/BF02620137).
- [GMR85] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. “The Knowledge Complexity of Interactive Proof-Systems (Extended Abstract)”. In: *17th ACM STOC*. ACM Press, May 1985, pp. 291–304. DOI: [10.1145/22145.22178](https://doi.org/10.1145/22145.22178).
- [HL18] Justin Holmgren and Alex Lombardi. “Cryptographic Hashing from Strong One-Way Functions (Or: One-Way Product Functions and Their Applications)”. In: *59th FOCS*. Ed. by Mikkel Thorup. IEEE Computer Society Press, Oct. 2018, pp. 850–858. DOI: [10.1109/FOCS.2018.00085](https://doi.org/10.1109/FOCS.2018.00085).
- [HR04] Chun-Yuan Hsiao and Leonid Reyzin. “Finding Collisions on a Public Road, or Do Secure Hash Functions Need Secret Coins?” In: *CRYPTO 2004*. Ed. by Matthew Franklin. Vol. 3152. LNCS. Springer, Berlin, Heidelberg, Aug. 2004, pp. 92–105. DOI: [10.1007/978-3-540-28628-8_6](https://doi.org/10.1007/978-3-540-28628-8_6).
- [Imp95] Russell Impagliazzo. “A Personal View of Average-Case Complexity”. In: *Proceedings of the Tenth Annual Structure in Complexity Theory Conference, Minneapolis, Minnesota, USA, June 19-22, 1995*. IEEE Computer Society, 1995, pp. 134–147. ISBN: 0-8186-7052-5. URL: <https://doi.org/10.1109/SCT.1995.514853>.
- [IR89] Russell Impagliazzo and Steven Rudich. “Limits on the Provable Consequences of One-Way Permutations”. In: *21st ACM STOC*. ACM Press, May 1989, pp. 44–61. DOI: [10.1145/73007.73012](https://doi.org/10.1145/73007.73012).
- [Knu98] Donald Ervin Knuth. *The art of computer programming, Volume II: Seminumerical Algorithms, 3rd Edition*. Addison-Wesley, 1998.
- [KY18] Ilan Komargodski and Eylon Yogev. “On Distributional Collision Resistant Hashing”. In: *CRYPTO 2018, Part II*. Ed. by Hovav Shacham and Alexandra Boldyreva. Vol. 10992. LNCS. Springer, Cham, Aug. 2018, pp. 303–327. DOI: [10.1007/978-3-319-96881-0_11](https://doi.org/10.1007/978-3-319-96881-0_11).
- [MF21] Arno Mittelbach and Marc Fischlin. *The Theory of Hash Functions and Random Oracles*. Springer, 2021. DOI: [10.1007/978-3-030-63287-8](https://doi.org/10.1007/978-3-030-63287-8).
- [MMNPs16] Mohammad Mahmoody, Ameer Mohammed, Soheil Nematihaji, Rafael Pass, and abhi shelat. *A Note on Black-Box Separations for Indistinguishability Obfuscation*. Cryptology ePrint Archive, Report 2016/316. 2016. URL: <https://eprint.iacr.org/2016/316>.
- [MOZ22] Alice Murphy, Adam O’Neill, and Mohammad Zaheri. “Instantiability of Classical Random-Oracle-Model Encryption Transforms”. In: *ASIACRYPT 2022, Part IV*. Ed. by Shweta Agrawal and Dongdai Lin. Vol. 13794. LNCS. Springer, Cham, Dec. 2022, pp. 323–352. DOI: [10.1007/978-3-031-22972-5_12](https://doi.org/10.1007/978-3-031-22972-5_12).
- [MV03] Daniele Micciancio and Salil P. Vadhan. “Statistical Zero-Knowledge Proofs with Efficient Provers: Lattice Problems and More”. In: *CRYPTO 2003*. Ed. by Dan Boneh. Vol. 2729. LNCS. Springer, Berlin, Heidelberg, Aug. 2003, pp. 282–298. DOI: [10.1007/978-3-540-45146-4_17](https://doi.org/10.1007/978-3-540-45146-4_17).

- [Ost91] Rafail Ostrovsky. “One-Way Functions, Hard on Average Problems, and Statistical Zero-Knowledge Proofs”. In: *Proceedings of the Sixth Annual Structure in Complexity Theory Conference, Chicago, Illinois, USA, June 30 - July 3, 1991*. IEEE Computer Society, 1991, pp. 133–138. ISBN: 0-8186-2255-5. URL: <https://doi.org/10.1109/SCT.1991.160253>.
- [PRS12] Krzysztof Pietrzak, Alon Rosen, and Gil Segev. “Lossy Functions Do Not Amplify Well”. In: *TCC 2012*. Ed. by Ronald Cramer. Vol. 7194. LNCS. Springer, Berlin, Heidelberg, Mar. 2012, pp. 458–475. DOI: [10.1007/978-3-642-28914-9_26](https://doi.org/10.1007/978-3-642-28914-9_26).
- [PS00] David Pointcheval and Jacques Stern. “Security Arguments for Digital Signatures and Blind Signatures”. In: *Journal of Cryptology* 13.3 (June 2000), pp. 361–396. DOI: [10.1007/s001450010003](https://doi.org/10.1007/s001450010003).
- [PW08] Chris Peikert and Brent Waters. “Lossy trapdoor functions and their applications”. In: *40th ACM STOC*. Ed. by Richard E. Ladner and Cynthia Dwork. ACM Press, May 2008, pp. 187–196. DOI: [10.1145/1374376.1374406](https://doi.org/10.1145/1374376.1374406).
- [PW11] Chris Peikert and Brent Waters. “Lossy Trapdoor Functions and Their Applications”. In: *SIAM J. Comput.* 40.6 (2011), pp. 1803–1844.
- [QWW21] Willy Quach, Brent Waters, and Daniel Wichs. “Targeted Lossy Functions and Applications”. In: *CRYPTO 2021, Part IV*. Ed. by Tal Malkin and Chris Peikert. Vol. 12828. LNCS. Virtual Event: Springer, Cham, Aug. 2021, pp. 424–453. DOI: [10.1007/978-3-030-84259-8_15](https://doi.org/10.1007/978-3-030-84259-8_15).
- [RTV04] Omer Reingold, Luca Trevisan, and Salil P. Vadhan. “Notions of Reducibility between Cryptographic Primitives”. In: *TCC 2004*. Ed. by Moni Naor. Vol. 2951. LNCS. Springer, Berlin, Heidelberg, Feb. 2004, pp. 1–20. DOI: [10.1007/978-3-540-24638-1_1](https://doi.org/10.1007/978-3-540-24638-1_1).
- [Sim98] Daniel R. Simon. “Finding Collisions on a One-Way Street: Can Secure Hash Functions Be Based on General Assumptions?” In: *EUROCRYPT’98*. Ed. by Kaisa Nyberg. Vol. 1403. LNCS. Springer, Berlin, Heidelberg, May 1998, pp. 334–345. DOI: [10.1007/BFb0054137](https://doi.org/10.1007/BFb0054137).
- [SV03] Amit Sahai and Salil P. Vadhan. “A complete problem for statistical zero knowledge”. In: *J. ACM* 50.2 (2003), pp. 196–249. URL: <https://doi.org/10.1145/636865.636868>.
- [Zha16] Mark Zhandry. “The Magic of ELFs”. In: *CRYPTO 2016, Part I*. Ed. by Matthew Robshaw and Jonathan Katz. Vol. 9814. LNCS. Springer, Berlin, Heidelberg, Aug. 2016, pp. 479–508. DOI: [10.1007/978-3-662-53018-4_18](https://doi.org/10.1007/978-3-662-53018-4_18).
- [Zha19] Mark Zhandry. “The Magic of ELFs”. In: *Journal of Cryptology* 32.3 (July 2019), pp. 825–866. DOI: [10.1007/s00145-018-9289-9](https://doi.org/10.1007/s00145-018-9289-9).

Appendix A Lazy-Sampling a Random Permutation

Lazy sampling of random *functions* is quite easy. For each new input x pick a fresh random response $y \leftarrow_{\$} \{0, 1\}^\lambda$, and store y in a table $T[x]$ for value x to answer subsequent requests about x consistently.

For lazy-sampling a random *permutation* one often reads that, in this case, y needs to be chosen randomly from $\{0, 1\}^\lambda \setminus \{y \mid T[x] \neq \perp\}$. While mathematically precise, this leaves open how this actual sampling should be carried out algorithmically. One option, if the number of previous queries q is sufficiently small, say, $q \leq 2^{\lambda/2}$, is to sample y repeatedly till a fresh value is found. This, however, yields only a lazy sampler with expected run time guarantees. Alternatively, one aborts after a sufficient number of attempts and needs to account for the statistical failure error and the run time influence due to the repeated trials, especially regarding the checks if y is actually fresh.

We present here a different method to lazy-sample a random permutation. The method is based on the Fisher-Yates algorithm, described here in the more efficient version popularized by Durstenfeld [Dur64] and Knuth [Knu98]. The idea of the Fisher-Yates algorithm is to start with an array $A[\]$ of the integers $A[0] = 0, A[1] = 1, A[2] = 2, \dots, A[N-1] = N-1$ and to iterate through all array cells, each time swapping the current value with one of the remaining cells: For $i = 0, 1, 2, \dots, N-1$ pick $j \leftarrow \$ \{i, i+1, \dots, N-1\}$ at random and exchange the contents of $A[i]$ and $A[j]$. Eventually one obtains the permutation by mapping each value i to $A[i]$. It is easy to see that each value has a chance of $\frac{1}{N}$ to end up at each position r , since it has a probability of $\prod_{i=0}^{r-1} \frac{N-1-i}{N-i} = \frac{N-r}{N}$ of being not picked in the first $r-1$ rounds, times $\frac{1}{N-r}$ of being chosen in the r -th round.

Of course, in our case a random permutation over $\{0, 1\}^\lambda$ is too large to be generated at the outset, such that we perform the new assignment on the fly. For this it is convenient to also keep track of the swaps in a table S , but where we use a sparse book keeping of only assigning values $S[j] \neq \perp$ to cells which have been involved in a swap (and setting $S[j] \leftarrow j$ only at the point in time where we access a previously untouched cell).

Init()	Lazy-Sample(x)
$S[\], T[\] \leftarrow \perp$ // of max. size 2^λ	if $T[x] \neq \perp$ then return $T[x]$
$c \leftarrow 0$ // counter	$j \leftarrow \$ \{c, c+1, \dots, 2^\lambda - 1\}$
return (S, T, c)	// swap values at c and j :
	if $S[j] = \perp$ then $S[j] \leftarrow j$
	if $S[c] = \perp$ then $S[c] \leftarrow c$
	$temp \leftarrow S[c]; S[c] \leftarrow S[j]; S[j] \leftarrow temp$
	// Int2Str $_\lambda : \{0, 1, 2, \dots, 2^\lambda - 1\} \rightarrow \{0, 1\}^\lambda$ canonically
	$T[x] \leftarrow \text{Int2Str}_\lambda(S[c])$
	$c \leftarrow c + 1$
	return $T[x]$

Note that we can store the tables S and T via common structures like search trees or skip lists with logarithmic run time. That is, after q queries the sampling procedures needs at most $\mathcal{O}(\log q)$ steps each to check if $T[x] \neq \perp$ among the at most q non-empty entries in T . This is also true for table S since each sampling sets at most two values in S , such that S contains at most $2q$ entries at this point. But then checking that $S[*] \neq \perp$ and swapping the values can be carried out in $\mathcal{O}(\log q)$ steps as well.

At first glance it seems as if the random sampling $j \leftarrow \$ \{c, c+1, \dots, 2^\lambda - 1\}$ suffers from similar problems as for sampling a fresh y from $\{0, 1\}^\lambda \setminus \{y \mid T[x] \neq \perp\}$. But note that the interval $[c, 2^\lambda - 1]$

is consecutive, and it suffices to pick a random integer between 0 and $2^\lambda - 1 - c$ and add c to the sample. Sampling such a random integer can be done, for instance, by choosing a 2λ -bit string *once* and reducing the corresponding integer modulo $2^\lambda - c$. The outcome is statistically close to uniform on $[0, 2^\lambda - 1 - c]$.

Appendix B Lossy Functions Imply Hard Statistical Zero-Knowledge Problems

We will show that HVPZK, the class of all problems with an honest-verifier perfect zero-knowledge proof, contains hard problems if lossy functions exist. As HVPZK is contained in HVSZK (honest-verifier *statistical* zero-knowledge) and as HVSZK = SZK, this means lossy functions imply hard problems in SZK.

Proposition B.1. *Let $(\text{Gen}, \text{Eval})$ be an ω -lossy function for $\omega \geq 1$. Then the language*

$$L = \{(1^\lambda, \text{pk}) \mid \text{pk} \in [\text{Gen}_{\text{inj}}(1^\lambda)]\}$$

is in HVPZK \ BPP. In particular, with the distribution $\mathcal{D}(1^\lambda)$ picking a random bit $b \leftarrow_{\$} \{0, 1\}$ and returning $\text{pk} \leftarrow_{\$} \text{Gen}_{\text{inj}}(1^\lambda)$ if $b = 0$ resp. $\text{pk} \leftarrow_{\$} \text{Gen}_{\text{loss}}(1^\lambda)$ if $b = 1$, we get a hard-on-average problem in HVPZK.

Proof. Note that injective public keys $\text{pk} \in [\text{Gen}_{\text{inj}}(1^\lambda)]$ cannot lie in the support of lossy keys $[\text{Gen}_{\text{loss}}(1^\lambda)]$, because the function cannot be injective and lossy at the same time. Hence, if the language L was in BPP, then one could the decision algorithm to decide with error at most $1/3$ if a given public key pk is injective or lossy. This, however, contradicts the indistinguishability of keys of the lossy function. This translates accordingly to the defined distribution \mathcal{D} for the hard-on-average problem.

Next, we present our honest-verifier perfect zero-knowledge protocol for L . The input to both parties, the prover and the verifier, is $(1^\lambda, \text{pk})$. The verifier picks a random $x \leftarrow_{\$} \{0, 1\}^\lambda$ and computes $y \leftarrow \text{Eval}(\text{pk}, x)$ and sends y to the prover. The prover searches for the preimage x^* of y under pk and returns x^* . The verifier accepts if and only if $x = x^*$.

For $(1^\lambda, \text{pk}) \in L$ the key is injective such that the prover finds the correct preimage $x^* = x$, making the verifier accept. For a lossy key pk , however, there are at least two potential preimages for y , each one equally like. Hence, a malicious prover can make the verifier accept with probability at most $1/2$. The simulator for the honest verifier works as follows: It samples $x \leftarrow_{\$} \{0, 1\}^\lambda$ and computes $y \leftarrow \text{Eval}(\text{pk}, x)$ as the verifier would. Then it pretends that the prover returns x . Note that this view is identical distributed to an actual protocol run between the prover and the verifier (for an injective key). \square

Appendix C Deferred Proofs

C.1 Proof for Lemma 3.7

Proof. Let us start with one-wayness. It is well-known that a random oracle is exponentially one-way in a distributional sense, i.e., over the choice of the random oracle (see e.g. [MF21] for a full proof):

$$\forall \mathcal{A}, \forall \lambda : \Pr_{\mathcal{O}, \mathcal{A}}[\mathcal{A} \text{ wins}] \leq \text{poly}(\lambda)2^{-\lambda}$$

Let us fix one adversary \mathcal{A} . Using the Markov inequality, we get

$$\forall \lambda : \Pr_{\mathcal{O}} [\Pr_{\mathcal{A}} [\mathcal{A} \text{ wins}] \geq \lambda^2 \text{poly}(\lambda) 2^{-\lambda}] \leq \frac{1}{\lambda^2}.$$

As the sum over all $\frac{1}{\lambda^2}$ converges, we can use the Borell-Cantelli lemma to show there only exists a zero-measure of random oracles such that the adversary is successful for infinitely many security parameters. Using the fact that there are only countable many adversaries, the set of random oracles for which some adversary is successful infinitely often is also a zero-measure. Therefore, every random oracle except for this zero-measure set of oracles is one-way.

Similarly to one-wayness, we know that a truncated random oracle is an exponentially-secure collision resistant hash function (as long as the output is still long enough; see [MF21] again for a full proof). Then, by a similar argument, we know that every random oracle except for a zero-measure can be used to construct a collision-resistant hash function.

Finally, let us show the result for one-way product functions. One-way product functions are a set of functions f_1, \dots, f_k such that any adversary trying to invert $f_1(x_1), \dots, f_k(x_k)$ for independent, uniform x_i will have a success probability of at most $2^{-kn} \cdot \text{poly}(n)$. In their paper, Holmgren and Lombardi [HL18] do not give an implementation from a random oracle, but it is quite clear one-way product functions can be built relative to a random oracle (with overwhelming probability). For this, note that OWPF can be seen as the combination of two properties: First, every f_i is exponentially-hard to invert, and second, inverting multiple f_i 's is as hard as inverting them independently. Assuming every f_i equals the random oracle, we've already shown the first property. The second property follows from the fact that for two different values x_1, x_2 , $\mathcal{O}(x_1)$ is completely independent of $\mathcal{O}(x_2)$, so as long as all inputs are different, inverting all f_i at once does not yield any advantage over inverting them independently. Therefore, OWPF can be built from random oracles (with overwhelming probability). Using Borel-Cantelli again yields the proof. \square

Single-to-Multi-Theorem Transformations for Non-Interactive Statistical Zero-Knowledge

Marc Fischlin

Felix Rohrbach

Cryptoplexity, Technische Universität Darmstadt, Germany

www.cryptoplexity.de

marc.fischlin@cryptoplexity.de

felix.rohrbach@cryptoplexity.de

Abstract

Non-interactive zero-knowledge proofs or arguments allow a prover to show validity of a statement without further interaction. For non-trivial statements such protocols require a setup assumption in form of a common random or reference string (CRS). Generally, the CRS can only be used for one statement (single-theorem zero-knowledge) such that a fresh CRS would need to be generated for each proof. Fortunately, Feige, Lapidot and Shamir (FOCS 1990) presented a transformation for any non-interactive zero-knowledge proof system that allows the CRS to be reused any polynomial number of times (multi-theorem zero-knowledge). This FLS transformation, however, is only known to work for either computational zero-knowledge or requires a structured, non-uniform common reference string.

In this paper we present FLS-like transformations that work for non-interactive statistical zero-knowledge arguments in the common *random* string model. They allow to go from single-theorem to multi-theorem zero-knowledge and also preserve soundness, for both properties in the adaptive and non-adaptive case. Our first transformation is based on the general assumption that one-way permutations exist, while our second transformation uses lattice-based assumptions. Additionally, we define different possible soundness notions for non-interactive arguments and discuss their relationships.

1 Introduction

In a non-interactive proof for a language \mathcal{L} the prover P shows validity of some theorem $x \in \mathcal{L}$ via a proof π based on a common string crs chosen by some external setup procedure. The common requirements are completeness —that the honest prover is able to convince the verifier V for true statements x — and soundness —that the verifier will not accept false statements $x \notin \mathcal{L}$ from malicious provers. Blum et al. [BFM88] showed that such non-interactive proofs can also be zero-knowledge [GMR89], saying that a simulator can create a proof π on behalf of P if it has the ability to place some trapdoor information in crs .

1.1 Flavors of Non-Interactive Zero-Knowledge

Non-interactive zero-knowledge protocols come in many variations:

- If the prover is computationally unbounded then one speaks of a NIZK *proof system* whereas in *arguments* or *argument systems* the prover runs in polynomial time [BCC88].
- Zero-knowledge may be *computational* (NICZK) or *statistical* (NISZK) or even *perfect* (NIPZK). Note that non-interactive statistical (or perfect) zero-knowledge for NP requires that the prover is computationally bounded, unless the polynomial hierarchy collapses [Ps05].
- The common string crs may be uniformly distributed over all bit strings of a certain length, in which case one speaks of the *common random string* or, less frequently, of the *uniform reference string* model. In any other case the string may have more structure and one calls it a *common reference string* or, sometimes, also *public parameter* model. In this work, we will focus on the case where the crs is uniformly distributed.

Another important aspect is the question of when malicious parties choose their challenge statement x . Both zero-knowledge and soundness come in an adaptive and in a non-adaptive version. The adaptive versions say that the adversary may choose the statement x after having seen the common reference string. For zero-knowledge this means that the simulator must prepare crs independently of x and then find a valid proof π after learning a maliciously chosen $x \in \mathcal{L}$. Adaptive soundness says that the malicious prover P^* first receives crs and then tries to find a false statement $x \notin \mathcal{L}$ with a convincing proof π .

Remarkably, for soundness one usually merely distinguishes between non-adaptive and adaptive notions. But there are also different ways how to capture the fact that a malicious prover P^* needs to succeed for an invalid statement $x \notin \mathcal{L}$. Either one assumes that the prover only outputs invalid statements, thus excluding some adversaries, or one penalizes the prover and declares it to lose if it chooses some $x \in \mathcal{L}$.¹ The penalizing definition implies the exclusive one. We note that Arte and Bellare [AB20], in a concurrent work, have proposed a similar distinction between exclusive and penalizing soundness.

Both notions, exclusive and penalizing soundness, already appeared implicitly in the literature, e.g., the work by Blum et al. [BSMP91] gives both an adaptive and a non-adaptive soundness definition in the exclusive setting. Indeed, non-adaptive soundness in the literature is often cast in this style. In contrast, for adaptive soundness nowadays one often encounters the penalizing variant. It seems, however, that the adaptive/exclusive version is already sufficient for many applications, e.g., to build universally composable NIZK protocols [GOS12]. We discuss this in more detail in Section 3 when defining the different versions.

1.2 From Single-Theorem to Multi-Theorem Proofs

In this work we focus on another important property of NIZK, namely, if the crs can be used only once (*bounded* or *single-theorem*) or is applicable for many proofs (*unbounded* or *multi-theorem*). The latter is of course preferable, and indeed Feige et al. [FLS90; FLS99] show how to generally turn

¹We use here the terminology from [BHK15] for the comparable scenario of admissible decryption queries in chosen-ciphertext security.

single-theorem NIZK proofs and arguments into multi-theorem zero-knowledge protocols. We call this the FLS-transformation.

The idea of the FLS-transformation is to augment the common random string by an extra uniformly distributed portion crs^{aux} and let the prover for this NP-language show that “ $x \in \mathcal{L}$ or crs^{aux} is the output of a pseudorandom generator”. This allows the simulator to create this part crs^{aux} pseudorandomly and use the generator’s seed as a witness for simulating the or-proof. If the original proof is zero-knowledge, then it is also witness indistinguishable [FS90], and then one cannot distinguish or-proofs generated by the genuine prover with the witness for x from proofs created by the simulator with the witness for crs^{aux} .

Soundness, on the other hand, is not affected because a random string crs^{aux} is not pseudorandom, except with exponentially small probability. Hence, for invalid x the “or” of the statements $x \notin \mathcal{L}$ or “ crs^{aux} is pseudorandom” would not be satisfied either with overwhelming probability. This implies that a prover would still need to break soundness of the or-protocol.

The FLS-transformation, per se, is only known to work for non-interactive *computational* zero-knowledge. The reason is that the pseudorandom string crs^{aux} of the zero-knowledge simulator is only computationally indistinguishable from a truly random string. There exists a folklore “dual version” of the FLS-transformation for non-interactive perfect (and therefore also statistical) zero-knowledge, where the crs contains a pseudorandom value by construction. But this transformation requires a structured, non-uniformly chosen crs, whereas we are interested in the setting of common *random* strings. For completeness, we provide a formal description of that folklore result along our terminology in Appendix A.

It is thus unclear if the FLS-transformation can be used equally smoothly for statistical zero-knowledge in the common random string model. For example, Peikert and Shiehian [PS19] recently presented a statistical zero-knowledge argument for NP based on LWE in the common random string model, which is only zero-knowledge for a single theorem. They therefore asked whether there is an FLS-like transformation to achieve multi-theorem zero-knowledge in the statistical case.

1.3 Known NISZK Constructions

There are only a few known constructions of NISZK and NIPZK protocols for the general class NP. Groth et al. [GOS06; GOS12] were the first to give a NIPZK argument for NP based on specific number-theoretic constructions over bilinear groups. Their protocol achieves multi-theorem adaptive zero-knowledge, but only non-adaptive/exclusive soundness (although this can be extended to some limited form of adaptive soundness, called adaptive culpable soundness). It is cast in the common reference string model.

Abe and Fehr [AF07] later showed how to achieve NIPZK arguments for NP under some form of the knowledge-of-exponent assumption. Their protocol achieves adaptive multi-theorem zero-knowledge and is adaptively sound (in the penalizing setting). This protocol is again in the common reference string model.

Sahai and Waters [SW14] show how to build NIPZK arguments for NP based on indistinguishability obfuscation and one-way functions. Their solution is adaptive multi-theorem zero-knowledge and non-adaptively/exclusively sound. It is designed in the common reference string model.

Peikert and Shiehian [PS19] constructed NISZK arguments for NP based on the LWE assumption. Their construction is based on the NIZK framework of Canetti et al. [CCRR18; Can+19] as well

Work	Soundness	CRS uniform?	ZK	Required
FLS* [FLS90; FLS99]	adaptive/ penalizing	✓	computational	PRGs
folklore* (see Appendix A)	adaptive/ exclusive	✗	perfect	PRGs
Groth et al. [GOS06; GOS12]	non-adaptive/ exclusive	✗	perfect	bilinear groups
Abe and Fehr [AF07]	adaptive/ penalizing	✗	perfect	knowledge-of- exponent
Sahai and Waters [SW14]	non-adaptive/ exclusive	✗	perfect	iO
Libert et al. [LPWW20]	non-adaptive/ exclusive	✗	statistical	k-linear
Libert et al. [LNPT20]	non-adaptive/ non-uniform	✓	statistical	LWE
this work*	adaptive/ exclusive	✓	statistical	OWP or LWE
	adaptive/ exclusive	✓	perfect	OWP+expected simulation

FIGURE 1.1 — Comparison of different multi-theorem NIZK schemes. The entries marked with * are actually transformations for the single-to-multi-theorem cases.

as Holmgren and Lombardi [HL18] which, among others, constructs a non-adaptively/exclusively sound NISZK argument for NP in the common random string model. Their protocol is adaptively zero-knowledge for single theorems. The instantiation of Peikert and Shiehian [PS19] uses the LWE assumption to implement the primitives and inherits the characteristics of the solutions in [CCRR18; HL18; Can+19].

An interesting observation, based on [CLW18, Footnote 13], is that one should be able to show adaptive soundness for the constructions in [Can+19] when using the exclusive notion. Noteworthy, Canetti et al. [CLW18] merely claim non-adaptive soundness, because for the adaptive version they switch to the penalizing variant. They detail why this notion cannot be achieved with the current construction, and the point touches precisely the difference between penalizing and exclusive soundness. Reverting to adaptive/exclusive soundness, the construction may satisfy this weaker level. This gives the interesting twist that the solution by Peikert and Shiehian [PS19] may already be adaptively/exclusively sound, such that our transformation lifts it from single-theorem to multi-theorem (adaptive) zero-knowledge.

Libert et al. [LPWW20] recently showed how to build *designated-verifier* statistical zero-knowledge arguments based on the (kernel) k -linear assumption, and how this construction can also be turned into a public verifiable NISZK argument. Their public verifiable construction achieves multi-theorem zero-knowledge and non-adaptive/exclusive soundness in the common reference string model.

In another work, Libert et al. [LNPT20] achieve multi-theorem zero-knowledge in the common *random* string model. Their protocol provides non-adaptive/non-uniform soundness, i.e., where one quantifies over all inputs $x \notin \mathcal{L}$ and the crs is chosen as part of the experiment. We will later argue that in the non-adaptive case this notion is equivalent to non-adaptive/exclusive and to non-adaptive/penalizing soundness for non-uniform provers.

1.4 Our Results

In this work we show multiple FLS-SZK-transformations which preserve statistical zero-knowledge. Moreover, they allow to preserve non-adaptive or adaptive zero-knowledge and also inherit the adaptive security of soundness (in the exclusive variant). In detail, we show:

- For statistical zero-knowledge we show how to transform any single-theorem zero-knowledge NISZK argument for NP-languages into one which is a multi-theorem zero-knowledge NISZK argument in the common random string model. This requires only the existence of one-way permutations².
- For perfect zero-knowledge we show that our transformation can be augmented to preserve perfect zero-knowledge. This, however, comes at the cost of having a zero-knowledge simulator which runs in expected polynomial-time.
- Finally, we show that we can build a transformation for statistical zero-knowledge from the Learning with Errors (LWE) assumption in the common random string model. This transformation, in contrast to the construction by Libert et al. [LNPT20], even works for *adaptively* sound NISZK arguments. This fits in nicely with the recent construction of statistical zero-knowledge arguments based on LWE [PS19].
- Additionally, we define and discuss the different soundness properties for non-interactive arguments and analyze their relationship. In particular, we show that in the non-adaptive case, the notions of exclusive, penalizing, and non-uniform soundness are all equivalent when considering non-uniform provers.

Our techniques for the constructions based on general assumptions uses a “dual” version of the original FLS-transformation. That is, instead of building the or-language for crs^{aux} being pseudorandom, we use that crs^{aux} is *not* pseudorandom. Since this is in general a coNP -language we need to make sure that it is also in NP . We achieve this by using the Blum-Micali-Yao pseudorandom generator [Yao82; BM84] based on one-way permutations and hardcore bits, which lies in $\text{NP} \cap \text{coNP}$. Soundness for our dual FLS-transformation then follows since we can let the malicious prover run on a pseudorandom string crs^{aux} instead, since this is indistinguishable for the efficient prover in an argument. Then the or of the two statements, $x \in \mathcal{L}$ or crs^{aux} is not pseudorandom, is again not satisfied.

The construction based on LWE is inspired by a primitive called dual-mode commitment scheme, i.e., a commitment which can be either perfectly-binding or statistically-hiding, based on the choice of how to generate the public key. The public keys for both modes are computationally indistinguishable. We note that the usefulness of such dual-mode commitments for non-interactive zero-knowledge is well known, starting with the work by Groth et al. [GOS06] where this technique was called parameter switching, to recent efforts like the construction of Libert et al. [LPWW20]. Most times, however, the solutions work over certain structures and yield arguments in the common reference string model.

Here, we use a construction of Gorbunov et al. [GVW15] to build these dual-mode commitments where the (statistically-hiding) public key and a commitment can be chosen as uniform bit strings. As in the FLS-transformation we extend the CRS by a public key string pk and a random commitment

²Note that we define one-way permutations as one-way functions that are 1-1 and length-preserving, not as a family of such functions.

string c and extend the language to “ $x \in \mathcal{L}$ or c is a commitment to 1”. For the simulator, we choose our public key to be statistically-hiding. In our construction, a statistically-hiding public key will be statistically close to a uniformly random string and indeed generate a commitment to the value 1.

However, for the soundness game we exchange the public key pk for a perfectly-binding one and change the commitment to 0, thereby forcing the malicious prover to prove x to be in \mathcal{L} . We emphasize that we only switch between these modes and merely require computational indistinguishability of the different types of public keys. In particular, we do not need to rely on the SIS assumption as considered in [GVW15] but, as pointed out in [CH19], the LWE assumption suffices. Indeed, one could directly use Regev’s LWE encryption scheme [Reg05] which also supports a statistically-hiding, lossy mode.

1.5 Squeezing in into Possibility and Impossibility Results

There are some known impossibility results for statistical and perfect zero-knowledge arguments. Strictly speaking, these results do not infringe with our results here, since we show how to *transform* statistical zero-knowledge arguments (from single to multiple theorems) but do not give constructions. Still, one may wonder if the combination of our transformations with the impossibility results have any implications on potential constructions.

Abe and Fehr [AF07] were the first to show that NISZK arguments cannot be proven to be adaptively sound via so-called direct black-box reductions, unless the language is in $P/poly$. One property which such direct reductions has is that one can use an efficient alternative to the crs generator which in addition outputs the simulator’s trapdoor information (property II.(b) in [AF07]). Our construction, however, bypasses this property because for the soundness proof it generates a bad crs which does not have a trapdoor. In this sense, our technique indicates that the notion of direct black-box reductions may be too restrictive.

Pass [Pas16], using similar ideas and techniques as [AF07], shows that *adaptive* statistical and perfect zero-knowledge arguments with *adaptive* soundness cannot be based on hard primitives via black-box reductions. How does the result of Pass [Pas16] match our results? First we remark that our NIPZK is indeed *adaptively* sound and *adaptively* zero-knowledge. But the simulator only runs in polynomial time averaged over its internal randomness. Such simulators escape the results in [Pas16].

Yet, the most striking difference between the results in [AF07; Pas16] and our transformations lies in the distinct notions of adaptive soundness. We show that our transformations preserve adaptive/exclusive soundness. Opposite to that, the impossibility results of [AF07; Pas16] rely on the ability of the malicious prover to occasionally output theorems $x \in \mathcal{L}$. Put differently, they rule out the stronger form of adaptive/penalizing sound arguments, whereas we argue that adaptive/exclusive soundness is preserved. As remarked above, however, adaptive/exclusive sound arguments may still be sufficient for applications.

1.6 Concurrent Work

As mentioned earlier, Arte and Bellare [AB20] have touched upon the issue of different soundness notions in non-interactive proofs as well. Their starting point are dual-mode systems in which the common reference string can be generated in two modes, and in how far such systems allow

for transference of security properties in the different modes. Our work instead focuses on the transformations for multi-theorem statistical zero-knowledge arguments.

Arte and Bellare define notions of penalizing and exclusive soundness, called SND-P and SND-E, with which our adaptive notions for soundness coincide (for efficient provers).³ Remarkably, they show a separating example of their exclusive and penalizing soundness notion in the adaptive case, under the decisional Diffie-Hellman assumption. This example applies to our notions in the adaptive setting as well. We complement this result by showing that the notions are equivalent in the non-adaptive case, assuming non-uniform provers.

Another notable difference between the two works lies in the applications of the different soundness notions. Arte and Bellare discuss the example of the Bellare-Goldwasser signature scheme where penalizing soundness is required and exclusive soundness is insufficient. We argue along the implication of culpability that exclusive soundness may suffice in many settings.

2 Preliminaries

An NP-relation \mathcal{R} consists of pairs (x, ω) of theorems and witnesses where the length of witness is polynomially bounded in the length of the theorem, and where one can efficiently decide membership. More formally, there exists a polynomial-time Turing machine $M_{\mathcal{R}}$ and a polynomial $p_{\mathcal{R}}$ such that

$$\mathcal{R} = \{(x, \omega) \mid |\omega| \leq p_{\mathcal{R}}(|x|) \wedge M_{\mathcal{R}}(x, \omega) = 1\}.$$

The induced language $\mathcal{L}_{\mathcal{R}}$ is given by

$$\mathcal{L}_{\mathcal{R}} = \{x \in \{0, 1\}^* \mid \exists \omega : (x, \omega) \in \mathcal{R}\}.$$

2.1 Non-Interactive Arguments

A non-interactive argument or proof system for an NP-relation is now a protocol in which the setup algorithm Setup generates a common string crs which the prover P then uses to generate a proof π for the input (x, ω) . The verifier V then checks this proof against crs and x only. There are some length restrictions, of course, namely that the length of the theorem x determines the length of the common string. In particular, we assume that there is a polynomial p_{Setup} such that $\text{crs} \in \{0, 1\}^{p_{\text{Setup}}(|x|)}$ for any $\text{crs} \stackrel{\$}{\leftarrow} \text{Setup}(1^\lambda)$. Let $\mathcal{R}(1^\lambda) = \{(x, \omega) \in \mathcal{R} \mid |x| = \lambda\}$ and $\mathcal{L}_{\mathcal{R}}(1^\lambda) = \{x \in \mathcal{L}_{\mathcal{R}} \mid |x| = \lambda\}$ denote the restriction of inputs of the relation and language with length $|x| = \lambda$ such that the length of the common string for such inputs is given by $p_{\text{Setup}}(\lambda)$. Note that the verifier can easily check that $|x|$ matches the security parameter λ such that we can assume that this is always the case.

We note that the string crs generated by Setup may be uniformly distributed, in which case we speak of a common random string. It may have a different distribution, in which case we call it a common reference string. In particular, we see a common random string as a special case of a common reference string.

The usual completeness notion of non-interactive arguments and proofs asks that the verifier V accepts genuine proofs π generated by the prover P for input $x \in \mathcal{L}_{\mathcal{R}}$. Soundness, on the hand, demands that the verifier does not accept false proofs generated by a malicious prover P^* for inputs

³Strictly speaking, their notion of exclusiveness allows for a negligible error which could be integrated in our notion as well.

$x \notin \mathcal{L}_{\mathcal{R}}$. As explained in the introduction there are various possibilities to define soundness, which we will discuss in Section 3, and just use one example of the possible definitions here.

Definition 2.1 (Non-interactive Argument). A non-interactive argument for an NP-relation \mathcal{R} (in the common reference string model) is a triple of probabilistic polynomial-time algorithms $\Pi = (\text{Setup}, P, V)$ satisfying the completeness and soundness condition:

(Perfect) Completeness: For every $\lambda \in \mathbb{N}$, every $(x, \omega) \in \mathcal{R}(1^\lambda)$, every $\text{crs} \xleftarrow{\$} \text{Setup}(1^\lambda)$, every $\pi \xleftarrow{\$} P(1^\lambda, x, \omega, \text{crs})$ we have that $V(1^\lambda, x, \pi, \text{crs}) = 1$ with probability 1.

(Non-Adaptive/Exclusive) Soundness: For every (possibly malicious) probabilistic polynomial-time prover P^* outputting only $x \notin \mathcal{L}_{\mathcal{R}}$ there exists a negligible function $\epsilon(\lambda)$ such that for every $\lambda \in \mathbb{N}$ we have

$$\Pr[V(1^\lambda, x, \pi, \text{crs}) = 1] \leq \epsilon(|x|),$$

where the probability is over $(x, \text{st}) \xleftarrow{\$} P^*(1^\lambda)$, $\text{crs} \xleftarrow{\$} \text{Setup}(1^\lambda)$, as well as $\pi \xleftarrow{\$} P^*(1^\lambda, \text{st}, \text{crs})$, and V 's randomness.

We say that the argument is in the common *random* string model if $\text{Setup}(\lambda)$ outputs uniformly distributed strings over $\{0, 1\}^{p_{\text{Setup}}(\lambda)}$ for every $\lambda \in \mathbb{N}$.

2.2 Zero-Knowledge

We next define zero-knowledge with the usual notion of a simulator ZKSim . In the non-interactive setting this algorithm has the advantage to choose the common string crs to simulate proofs. In the bounded case the distinguisher only gets to see a single proof for a chosen theorem, where the proof is either genuine or fabricated by the simulator. We simultaneously define the single-theorem and multi-theorem case where the distinguisher learns one or many (genuine or simulated) proofs. We first define both cases in the adaptive setting where the distinguisher selects the theorems in dependence of the common string and of previous proofs and in the non-adaptive case where the distinguisher chooses the statement(s) in advance. We stress that we are interested in statistical zero-knowledge here such that the distinguisher is unbounded, except that it can only ask for polynomially many proofs. We also allow the simulator to run in *expected* polynomial time in specially marked cases.

Definition 2.2 (Statistical and Perfect Zero Knowledge). Let \mathcal{R} be an NP-relation and let $\Pi = (\text{Setup}, P, V)$ be a non-interactive argument for \mathcal{R} . The argument is zero-knowledge if it satisfies one of the following properties:

Non-adaptive multi-theorem zero-knowledge: For any unbounded algorithm D there exists a probabilistic algorithm ZKSim , the simulator, running in (expected) polynomial time, such that the advantage

$$\text{Adv}_{\Pi, \text{ZKSim}, D}^{\text{naSZK}}(1^\lambda) := \Pr \left[\text{Expt}_{\Pi, \text{ZKSim}, D}^{\text{naSZK}}(1^\lambda) = 1 \right] - \frac{1}{2}$$

is negligible for polynomially bounded q , where experiment $\text{Expt}_{\Pi, \text{ZKSim}, D}^{\text{naSZK}}(1^\lambda)$ is defined in Figure 2.2. If the advantage of any such D is always 0 then the argument is called perfect zero-knowledge.

$\text{Expt}_{\Pi, \text{ZKSim}, D}^{\text{naSZK}}(1^\lambda)$:	$\text{Expt}_{(\text{Setup}, P, V), \text{ZKSim}, D}^{\text{aSZK}}(1^\lambda)$:
1 $b \xleftarrow{\$} \{0, 1\}$	1 $b \xleftarrow{\$} \{0, 1\}, q \leftarrow 0, \text{st}_D \leftarrow \perp$
2 $(\text{st}_D, x_1, \omega_1, \dots, x_q, \omega_q) \xleftarrow{\$} D(1^\lambda)$	2 $\text{crs}_0 \xleftarrow{\$} \text{Setup}(1^\lambda)$
3 $\text{crs}_0 \xleftarrow{\$} \text{Setup}(1^\lambda)$	3 $(\text{crs}_1, \text{st}_{\text{ZKSim}}) \xleftarrow{\$} \text{ZKSim}(1^\lambda)$
4 $(\text{crs}_1, \text{st}_{\text{ZKSim}}) \xleftarrow{\$} \text{ZKSim}(1^\lambda)$	4 repeat
5 for $i = 1..q$ do	5 $q \leftarrow q + 1$
6 if $(x_i, \omega_i) \in \mathcal{R}$ then	6 $(\text{st}_D, x, \omega) \xleftarrow{\$} D(1^\lambda, \text{st}_D, \text{crs}_b)$
7 $\pi_{i,0} \xleftarrow{\$} P(1^\lambda, x_i, \omega_i, \text{crs}_0)$	7 if $(x, \omega) \in \mathcal{R}$ then
8 $\pi_{i,1} \xleftarrow{\$} \text{ZKSim}(1^\lambda, \text{st}_{\text{ZKSim}}, x_i)$	8 $\pi_0 \xleftarrow{\$} P(1^\lambda, x, \omega, \text{crs}_0)$
9 else $\pi_{i,0} \leftarrow \pi_{i,1} \leftarrow \perp$	9 $\pi_1 \xleftarrow{\$} \text{ZKSim}(1^\lambda, \text{st}_{\text{ZKSim}}, x)$
10 $d \xleftarrow{\$} D(1^\lambda, \text{st}_D, \pi_{1,b}, \dots, \pi_{q,b}, \text{crs}_b)$	10 else $\pi_0 \leftarrow \pi_1 \leftarrow \perp$
11 return $b = d$	11 $(\text{st}_D, \text{cont}, d) \xleftarrow{\$} D(1^\lambda, \text{st}_D, \pi_b)$
	12 until $\text{cont} = \text{false}$
	13 return $b = d$

FIGURE 2.2 — Non-adaptive and adaptive statistical zero-knowledge experiments.

Adaptive multi-theorem zero knowledge: For any unbounded algorithm D there exists a probabilistic algorithm ZKSim , the simulator, running in (expected) polynomial time, such that the advantage

$$\text{Adv}_{\Pi, \text{ZKSim}, D}^{\text{aSZK}}(1^\lambda) := \Pr \left[\text{Expt}_{\Pi, \text{ZKSim}, D}^{\text{aSZK}}(1^\lambda) = 1 \right] - \frac{1}{2}$$

is negligible for polynomially bounded q , where experiment $\text{Expt}_{\Pi, \text{ZKSim}, D}^{\text{aSZK}}(1^\lambda)$ is defined in Figure 2.2. If the advantage of any such D is always 0 then the argument is called perfect zero-knowledge.

The argument is single-theorem zero-knowledge of the corresponding type if the property holds for $q = 1$.

Definition 2.3 (Statistical Witness Indistinguishability). Let \mathcal{R} be an NP-relation. A non-interactive argument $\Pi = (\text{Setup}, P, V)$ for \mathcal{R} is called *statistical witness indistinguishable* (NISWI) if it satisfies one of the following properties:

Non-Adaptive multi-theorem witness indistinguishability: For any unbounded algorithm D the advantage

$$\text{Adv}_{\Pi, D}^{\text{naSWI}}(1^\lambda) := \Pr \left[\text{Expt}_{\Pi, D}^{\text{naSWI}}(1^\lambda) = 1 \right] - \frac{1}{2}$$

is negligible for polynomially bounded q , where the experiment $\text{Expt}_{\Pi, D}^{\text{naSWI}}(1^\lambda)$ is defined in Figure 2.3. If the advantage of any such D is always 0 then the argument is called perfect witness indistinguishable.

Adaptive multi-theorem witness indistinguishability: For any unbounded algorithm D the advantage

$$\text{Adv}_{\Pi, D}^{\text{aSWI}}(1^\lambda) := \Pr \left[\text{Expt}_{\Pi, D}^{\text{aSWI}}(1^\lambda) = 1 \right] - \frac{1}{2}$$

is negligible for polynomially bounded q , where the experiment $\text{Expt}_{\Pi, D}^{\text{aSWI}}(1^\lambda)$ is defined in Figure 2.3. If the advantage of any such D is always 0 then the argument is called perfect witness indistinguishable.

<u>Expt_{Π,D}^{naSWI}(1^λ):</u>	<u>Expt_{Π,D}^{aSWI}(1^λ):</u>
1 $b \xleftarrow{\$} \{0, 1\}$	1 $b \xleftarrow{\$} \{0, 1\}, q \leftarrow 0, \text{st}_D \leftarrow \perp$
2 $(\text{st}_D, (x_i, \omega_{i,0}, \omega_{i,1})_{i=1..q}) \xleftarrow{\$} D(1^\lambda)$	2 $\text{crs} \xleftarrow{\$} \text{Setup}(1^\lambda)$
3 $\text{crs} \xleftarrow{\$} \text{Setup}(1^\lambda)$	3 repeat
4 for $i = 1..q$ do	4 $q \leftarrow q + 1$
5 if $(x_i, \omega_{i,0}) \in \mathcal{R} \wedge (x_i, \omega_{i,1}) \in \mathcal{R}$	5 $(\text{st}_D, x, \omega_0, \omega_1) \xleftarrow{\$} D(1^\lambda, \text{st}_D, \text{crs}_b)$
6 $\pi_{i,0} \xleftarrow{\$} P(1^\lambda, x_i, \omega_{i,0}, \text{crs})$	6 if $(x, \omega_0) \in \mathcal{R} \wedge (x, \omega_1) \in \mathcal{R}$
7 $\pi_{i,1} \xleftarrow{\$} P(1^\lambda, x_i, \omega_{i,1}, \text{crs})$	7 $\pi_0 \xleftarrow{\$} P(1^\lambda, x, \omega_0, \text{crs})$
8 else $\pi_{i,0} \leftarrow \pi_{i,1} \leftarrow \perp$	8 $\pi_1 \xleftarrow{\$} P(1^\lambda, x, \omega_1, \text{crs})$
9 $d \xleftarrow{\$} D(1^\lambda, \text{st}_D, \pi_{1,b}, \dots, \pi_{q,b}, \text{crs})$	9 else $\pi_0 \leftarrow \pi_1 \leftarrow \perp$
10 return $b = d$	10 $(\text{st}_D, \text{cont}, d) \xleftarrow{\$} D(1^\lambda, \text{st}_D, \pi_b)$
	11 until cont = false
	12 return $b = d$

FIGURE 2.3 — Non-adaptive and adaptive statistical witness indistinguishability experiments.

The argument is single-theorem witness indistinguishable of the corresponding type if the property holds for $q = 1$.

2.3 From Single-Theorem Zero-Knowledge to Multi-Theorem Witness Indistinguishability

We repeat here the well known fact that zero-knowledge implies witness indistinguishability, and that witness indistinguishability is closed under repetitions [FS90]. We state the results here for sake of completeness and according to our terminology in the statistical setting.

Lemma 2.4. *Any adaptive resp. non-adaptive single-theorem NISZK argument is also an adaptive resp. non-adaptive single-theorem NISWI argument.*

Proof (Sketch). We only argue the adaptive case; the non-adaptive case follows analogously. We can perform a game hop starting with the witness-indistinguishability experiment $\text{Expt}_{\Pi,D}^{\text{aSWI}}(1^\lambda)$. In this hop we replace the CRS and both proofs π_0 and π_1 in each iteration by simulated ones, all created by the simulator ZKSim without knowledge of the witnesses ω_0 and ω_1 but using the same trapdoor. Note that we can view the proofs in the WI experiment as two sequentially requested proofs in the ZK experiment, such that the SZK property ensures that this hop is statistically indistinguishable. (In the non-adaptive case we would split each entry $(x_i, \omega_{i,0}, \omega_{i,1})$ in D 's initial choice into two entries $(x_i, \omega_{i,0})$ and $(x, \omega_{i,1})$.)

But now both proofs π_0 and π_1 are created without the specific witness, and since the simulator does not update its state for giving proofs, the order in which the proofs are computed is irrelevant. In this case the bit b is perfectly hidden from the distinguisher such that the advantage in predicting b is 0. \square

Lemma 2.5. *Any adaptive resp. non-adaptive single-theorem NISWI argument is also an adaptive resp. non-adaptive multi-theorem NISWI argument.*

Proof (Sketch). We again only discuss the adaptive case since the non-adaptive case follows analogously. The proof follows by a hybrid argument. For this we reduce the multi-theorem distinguisher D to a bounded one D_1 which only makes one query. Let $Q(\lambda)$ be a polynomial upper

bound on the number of queries q which D makes. The bounded distinguisher D_1 initially picks an index $i \xleftarrow{\$} \{1, 2, \dots, Q(\lambda)\}$ and then internally runs in the first stage (Line 5) the distinguisher D up to the i -th query $(st_D, x, \omega_0, \omega_1)$. All requested proofs up to this step are computed internally by D_1 via P and the left witness, and returned to D . The i -th query is then computed externally, and D_1 then hands the proof back to D . In the final steps till halting, D_1 computes the remaining proofs for ω_1 , and eventually returns D 's decision bit d unchanged.

It can be shown that the advantage of the bounded distinguisher D_1 is at most a factor $Q(\lambda)$ larger than the one of D . Since $Q(\lambda)$ is polynomial, the difference is negligible. \square

3 Soundness of Non-Interactive Arguments

Soundness of a non-interactive argument assures that a (computationally-bound) malicious prover is unable to convince the verifier of a false statement. Commonly, soundness is defined in two variants: Adaptive soundness, which allows the (possibly malicious) prover P^* to choose the statement to prove x before seeing the common random string crs, and non-adaptive soundness, in which the prover P^* has to decide on the statement x before the common random string crs is generated.

Remarkably, there is another dimension of definitional choice for soundness which often goes unnoticed in the literature. This dimension refers to the question how we measure success of the malicious prover. Clearly, the malicious prover should not make the verifier accept for a statement x not in the language. But there are two possibilities to capture the non-membership requirement. One is to disallow P^* to output $x \in \mathcal{L}$ at all. The other one is to declare P^* to lose if it picks $x \in \mathcal{L}$. Following the work of Bellare et al. [BHK15] about the question how to deal with inadmissible decryption queries in CCA-secure encryption schemes, we call the former stipulation of P^* outputting only $x \notin \mathcal{L}$ *exclusive*, because it excludes certain adversaries. The latter is called *penalizing* as it punishes P^* if it chooses $x \in \mathcal{L}$.

3.1 Soundness Definitions

In total, we define five soundness notions: adaptive vs. non-adaptive, and exclusive vs. penalizing, as well as a non-uniform variant that only exists for the non-adaptive case. We typically speak of non-adaptive/exclusive and adaptive/penalizing soundness etc. to distinguish the different types. Figure 3.4 provides an overview. It is also easy to see that adaptive soundness implies non-adaptive soundness in both settings, and penalizing soundness implies exclusive soundness in any of the other dimensions. The latter is easy to see because any malicious prover P^* breaking exclusive soundness must output $x \notin \mathcal{L}$ such that this prover also satisfies the winning condition in the penalizing setting. In this chapter, we highlight the further connections between these definitions and their implications.

The difference between exclusive and penalizing soundness may appear to be insignificant. Indeed, for non-interactive *proofs* it is folklore to show that the weakest one of the five notions, non-adaptive/exclusive soundness, implies the strongest one, adaptive/penalizing soundness. See for instance [Gol06]. This may explain why today's literature mostly distinguishes between the (exclusive) non-adaptive notion and the (penalizing) adaptive notion. An exception is the seminal paper by Blum et al. [BDMP91] which defines the adaptive version according to the exclusive dimension (without using our terminology here, of course). We emphasize, however, that the equivalence of all notions is not known to hold for non-interactive *arguments*.

<i>non-adaptive</i>	<i>adaptive</i>
<p><u>exclusive soundness:</u></p> <ol style="list-style-type: none"> 1 // only P^* outputting $x \notin \mathcal{L}$ 2 $(x, \text{stp}^*) \xleftarrow{\\$} P^*(1^\lambda)$ 3 $\text{crs} \xleftarrow{\\$} \text{Setup}(1^\lambda)$ 4 $\pi \xleftarrow{\\$} P^*(1^\lambda, \text{crs}, \text{stp}^*)$ 5 return $V(1^\lambda, x, \pi, \text{crs})$ <p><u>penalizing soundness:</u></p> <ol style="list-style-type: none"> 1 $(x, \text{stp}^*) \xleftarrow{\\$} P^*(1^\lambda)$ 2 $\text{crs} \xleftarrow{\\$} \text{Setup}(1^\lambda)$ 3 $\pi \xleftarrow{\\$} P^*(1^\lambda, \text{crs}, \text{stp}^*)$ 4 return $V(1^\lambda, x, \pi, \text{crs}) \wedge x \notin \mathcal{L}$ <p><u>non-uniform soundness:</u></p> <ol style="list-style-type: none"> 1 // for all $x \notin \mathcal{L}$ 2 $\text{crs} \xleftarrow{\\$} \text{Setup}(1^\lambda)$ 3 $\pi \xleftarrow{\\$} P^*(1^\lambda, \text{crs}, x)$ 4 return $V(1^\lambda, x, \pi, \text{crs})$ 	<p><u>exclusive soundness:</u></p> <ol style="list-style-type: none"> 1 // only P^* outputting $x \notin \mathcal{L}$ 2 3 $\text{crs} \xleftarrow{\\$} \text{Setup}(1^\lambda)$ 4 $(x, \pi) \xleftarrow{\\$} P^*(1^\lambda, \text{crs})$ 5 return $V(1^\lambda, x, \pi, \text{crs})$ <p><u>penalizing soundness:</u></p> <ol style="list-style-type: none"> 1 2 $\text{crs} \xleftarrow{\\$} \text{Setup}(1^\lambda)$ 3 $(x, \pi) \xleftarrow{\\$} P^*(1^\lambda, \text{crs})$ 4 return $V(1^\lambda, x, \pi, \text{crs}) \wedge x \notin \mathcal{L}$

FIGURE 3.4 — Different notions of soundness.

Is a more fine-grained distinction between exclusive and penalizing soundness in arguments necessary? We argue that it is. Roughly, the difference is that in the exclusive case the malicious prover (and any other party) knows that its output is not in the language, in the penalizing case even the prover may itself be oblivious about this. This is an important ingredient in Pass' impossibility result to build adaptive sound and adaptive statistical zero-knowledge arguments based on black-box reductions [Pas16]. The result crucially relies on the malicious prover choosing a (random or pseudorandom) statement for which it does not know the status. In other words, this impossibility results rules out the strongest form of adaptive/penalizing soundness.

We next argue that the weaker form of adaptive/exclusive soundness is very relevant. It is easy to see that this notion implies a slightly weaker notion of adaptive/*culpable* soundness [GOS12]. This notion is similar to our definition of adaptive/exclusive soundness, but also requires the malicious prover to output an efficiently verifiable witness (denoted ω_{guilt} in [GOS12]) that the statement x is *not* in the language \mathcal{L} . Our exclusive notion asks P^* to output $x \notin \mathcal{L}$. We prove the implication that adaptive/exclusive yields adaptive/culpable soundness formally in Section 3.3.

The noteworthy fact is that adaptive/culpable soundness suffices for many applications. One of the most important ones is the possibility to derive universally composable NIZK argument [GOS12]. Other applications include correctness proofs for shuffles [GL07; FL16; FLSZ17] or for e-voting [CG15]. Since adaptive/exclusive soundness implies adaptive/culpable soundness, any protocol

satisfying the exclusive notion is also applicable in such settings.

We can now define non-interactive arguments with the different soundness properties:

Definition 3.1 (Soundness of non-interactive Arguments). A non-interactive argument for an NP-relation \mathcal{R} (in the common reference string model) is a triple of probabilistic polynomial-time algorithms $\Pi = (\text{Setup}, P, V)$ satisfying the completeness as well as at least one of the soundness conditions:

Non-Adaptive/Exclusive Soundness: For every (possibly malicious) probabilistic polynomial-time prover P^* outputting only $x \notin \mathcal{L}_{\mathcal{R}}$ there exists a negligible function $\epsilon(\lambda)$ such that for every $\lambda \in \mathbb{N}$ we have

$$\Pr[V(1^\lambda, x, \pi, \text{crs}) = 1] \leq \epsilon(|x|),$$

where the probability is over $(x, \text{st}) \xleftarrow{\$} P^*(1^\lambda)$, $\text{crs} \xleftarrow{\$} \text{Setup}(1^\lambda)$, as well as $\pi \xleftarrow{\$} P^*(1^\lambda, \text{st}, \text{crs})$, and V 's randomness.

Non-Adaptive/Penalizing Soundness: For every (possibly malicious) probabilistic polynomial-time prover P^* there exists a negligible function $\epsilon(\lambda)$ such that for every $\lambda \in \mathbb{N}$ we have

$$\Pr[V(1^\lambda, x, \pi, \text{crs}) = 1 \wedge x \notin \mathcal{L}_{\mathcal{R}}] \leq \epsilon(|x|),$$

where the probability is over $(x, \text{st}) \xleftarrow{\$} P^*(1^\lambda)$, $\text{crs} \xleftarrow{\$} \text{Setup}(1^\lambda)$, as well as $\pi \xleftarrow{\$} P^*(1^\lambda, \text{st}, \text{crs})$, and V 's randomness.

Adaptive/Exclusive Soundness: For every (possibly malicious) probabilistic polynomial-time prover P^* outputting only $x \notin \mathcal{L}_{\mathcal{R}}$ there exists a negligible function $\epsilon(\lambda)$ such that for every $\lambda \in \mathbb{N}$ we have

$$\Pr[V(1^\lambda, x, \pi, \text{crs}) = 1] \leq \epsilon(|x|),$$

where the probability is over $\text{crs} \xleftarrow{\$} \text{Setup}(1^\lambda)$, $(x, \pi) \xleftarrow{\$} P^*(1^\lambda, \text{crs})$, and V 's randomness.

Adaptive/Penalizing Soundness: For every (possibly malicious) probabilistic polynomial-time prover P^* there exists a negligible function $\epsilon(\lambda)$ such that for every $\lambda \in \mathbb{N}$ we have

$$\Pr[V(1^\lambda, x, \pi, \text{crs}) = 1 \wedge x \notin \mathcal{L}_{\mathcal{R}}] \leq \epsilon(|x|),$$

where the probability is over $\text{crs} \xleftarrow{\$} \text{Setup}(1^\lambda)$, $(x, \pi) \xleftarrow{\$} P^*(1^\lambda, \text{crs})$, and V 's randomness.

Non-Adaptive/Non-Uniform Soundness: For every (possibly malicious) probabilistic polynomial-time prover P^* there exists a negligible function $\epsilon(\lambda)$ such that for every $\lambda \in \mathbb{N}$ and every $x \notin \mathcal{L}_{\mathcal{R}}$ with $|x| = \lambda$, we have

$$\Pr[V(1^\lambda, x, \pi, \text{crs}) = 1 \wedge x \notin \mathcal{L}_{\mathcal{R}}] \leq \epsilon(|x|),$$

where the probability is over $\text{crs} \xleftarrow{\$} \text{Setup}(1^\lambda)$, and $\pi \xleftarrow{\$} P^*(1^\lambda, x, \text{crs})$, and V 's randomness.

3.2 Equivalence of the Non-Adaptive Soundness Notions

We now show that the non-adaptive soundness definitions are all equivalent if we allow the malicious provers to be non-uniform:

Theorem 3.2. *For non-uniform (malicious) provers, a non-interactive argument $\Pi = (\text{Setup}, P, V)$ has non-adaptive/exclusive soundness iff it has non-adaptive/non-uniform soundness, and has non-adaptive/non-uniform soundness iff it has non-adaptive/penalizing soundness.*

Proof. Non-adaptive/exclusive soundness follows directly from non-adaptive/penalizing soundness, therefore we only need to show that non-adaptive/non-uniform soundness follows from non-adaptive/exclusive soundness and that non-adaptive/penalizing soundness follows from non-adaptive/non-uniform soundness.

We start by showing non-adaptive/non-uniform soundness follows from non-adaptive/exclusive soundness. Let $\Pi = (\text{Setup}, P, V)$ be the non-interactive argument in question. Assume that there exists a successful malicious prover $P_{na/nu}^*$ against the non-adaptive/non-uniform soundness, i.e., for any negligible function $\epsilon(n)$ there exists an $x \notin \mathcal{L}$ such that

$$\Pr \left[V(\text{crs}, x, P_{na/nu}^*(\text{crs}, x)) \right] > \epsilon(|x|),$$

where the probability is over $\text{crs} \stackrel{\$}{\leftarrow} \text{Setup}(1^\lambda)$, as well as $P_{na/nu}^*$'s and V 's randomness. We can now construct a malicious prover $P_{na/ex}^*$ against non-adaptive/exclusive soundness as follows: We define the first-stage algorithm $P_{na/ex,1}^*(1^\lambda)$ to choose $x \notin \mathcal{L}$ of length λ non-uniformly, such that $P_{na/nu}^*$'s success probability is maximized. The state st is left empty. Further, the second-stage algorithm $P_{na/ex,2}^*$ merely calls $P_{na/nu}^*$ internally, ignoring the state st . Then, the success probability of $P_{na/ex}^*$ is at least as large as the one of $P_{na/nu}^*$ and thus non-negligible.

Next, we show that non-adaptive/penalizing soundness follows from non-adaptive/non-uniform soundness. Assume that there exists a successful malicious prover $P_{na/pn}^*$ against the non-adaptive/penalizing soundness, i.e., for any negligible function ϵ there exists an $\lambda \in \mathbb{N}$ such that

$$\Pr[V(\text{crs}, x, \pi) = 1 \wedge x \notin \mathcal{L}] > \epsilon(\lambda),$$

where the probability is over $(x, st) \stackrel{\$}{\leftarrow} P_{na/pn,1}^*(1^\lambda)$, $\text{crs} \stackrel{\$}{\leftarrow} \text{Setup}(1^\lambda)$, $\pi \stackrel{\$}{\leftarrow} P_{na/pn,2}^*$ as well as V 's internal randomness.

We can now construct a malicious prover $P_{na/nu}^*$ against non-adaptive/non-uniform soundness as follows: For each input length λ , we fix the pair (\bar{x}, \bar{st}) , $\bar{x} \in \{0, 1\}^\lambda$, $\bar{x} \notin \mathcal{L}$, on which $P_{na/pn,2}^*$'s success probability is maximized (we bound the length of \bar{st} by $P_{na/pn,1}^*$'s running time). Next we define $P_{na/nu}^*$ as follows: On input x , $P_{na/nu}^*$ checks whether x equals \bar{x} , and if that is the case, it internally calls $P_{na/pn,2}^*(\text{crs}, \bar{x}, \bar{st})$ to generate a proof. Otherwise, $P_{na/nu}^*$ returns an empty proof. Note that we use the non-uniformity to save the sequence of (\bar{x}, \bar{st}) for each input length. It is again easy to see that this prover is indeed a successful malicious prover against non-adaptive/non-uniform soundness. \square

For adaptive soundness, Arte and Bellare [AB20] showed that there exists a protocol that provides *adaptive/exclusive* soundness but not *adaptive/penalizing* soundness. This indicates that a NISZK protocol with *adaptive/exclusive* soundness might indeed be achievable, compared to one with *adaptive/penalizing* soundness, for which Pass [Pas16] showed a black-box impossibility result.

3.3 Exclusive Soundness Implies Culpable Soundness

In this section we show that adaptive/exclusive soundness implies the notion of adaptive/culpable soundness of [GOS12]. We first recall the definition of culpable soundness (according to our terminology). For an NP-relation \mathcal{R} let $\mathcal{R}_{\text{guilt}}$ be an NP-relation for the complement of $\mathcal{L}_{\mathcal{R}}$, i.e., $x \notin \mathcal{L}_{\mathcal{R}}$ means that there is a polynomial size ω_{guilt} such that $(x, \omega_{\text{guilt}}) \in \mathcal{R}_{\text{guilt}}$. Note that the relation $\mathcal{R}_{\text{guilt}}$ is efficiently verifiable as an NP-relation (and $\mathcal{L}_{\mathcal{R}}$ is therefore in co-NP).

Definition 3.3 (Adaptive/Culpable Soundness). A non-interactive argument (Setup, P, V) for an NP-relation \mathcal{R} (in the common reference string model) has adaptive culpable soundness if for any PPT algorithm P_{culp}^* there exists a negligible function ϵ such that

$$\Pr[V(1^\lambda, x, \pi, \text{crs}) = 1 \wedge (x, \omega_{\text{guilt}}) \in \mathcal{R}_{\text{guilt}}] \leq \epsilon(\lambda),$$

where the probability is over $\text{crs} \xleftarrow{\$} \text{Setup}(1^\lambda)$, $(x, \pi, \omega_{\text{guilt}}) \xleftarrow{\$} P_{\text{culp}}^*(1^\lambda, \text{crs})$, and V 's internal randomness.

Proposition 3.4. *A non-interactive argument (Setup, P, V) for an NP-relation \mathcal{R} (in the common reference string model) which has a corresponding relation $\mathcal{R}_{\text{guilt}}$ and is adaptive/exclusive sound is also adaptive/culpable sound.*

Proof. Assume that we have a successful prover P_{culp}^* against culpable soundness. We construct a malicious prover P_{ex}^* against exclusive soundness as follows. P_{ex}^* receives as input crs and forwards this to P_{culp}^* which, then, outputs $(x, \pi, \omega_{\text{guilt}})$. Our prover P_{ex}^* checks in polynomial time if $(x, \omega_{\text{guilt}}) \in \mathcal{R}_{\text{guilt}}$. If not it immediately outputs \perp , else it returns (x, π) .

Note that since we interpret outputs \perp as $\perp \notin \mathcal{L}_{\mathcal{R}}$ our prover P_{ex}^* only outputs values not in the language. It is thus an admissible attacker against exclusive soundness. Furthermore, P_{culp}^* can only win for $x \notin \mathcal{L}_{\mathcal{R}}$ such that only outputting (x, π) for those x cannot decrease the success probability. This yields that P_{ex}^* has the same success probability as P_{culp}^* . \square

4 Constructions based on General Assumptions

In this section we discuss our constructions based on one-wayness.

4.1 Multi-theorem NISZK based on One-way Permutations

Our approach uses the same idea as in [FLS90] of having crs^{aux} , but we apply it in a dual way. That is, we use a language saying that crs^{aux} is *not* pseudorandom. Since this is in general a coNP-relation we use the Blum-Micali-Yao [Yao82; BM84] generator for one-way permutations,

$$G(s) = f^{|s|}(s) \| \text{hb}(s) \| \text{hb}(f(s)) \| \dots \| \text{hb}(f^{|s|-1}(s)),$$

where s is the seed of length $|s| = \lambda$, f is a one-way permutation, $f^i(s)$ the i -fold iteration of f for input s , and hb is a hardcore bit for f . Proving that a string crs^{aux} is *not* in the range of G is easy if one presents the unique seed s such that the first bits are equal to $f^{|s|}(s)$ and that the remaining bits are not the hardcore bits.

$\text{Setup}(1^\lambda)$	$\text{P}(1^\lambda, x, \omega, \text{crs})$	$\text{V}(1^\lambda, x, \pi, \text{crs})$
$\text{crs}^{\text{or}} \xleftarrow{\$} \text{Setup}^{\text{or}}(1^\lambda)$	$\text{// crs} = \text{crs}^{\text{or}} \parallel \text{crs}^{\text{aux}}$	$\text{// crs} = \text{crs}^{\text{or}} \parallel \text{crs}^{\text{aux}}$
$\text{crs}^{\text{aux}} \xleftarrow{\$} \{0, 1\}^{2\lambda}$	$\pi^{\text{or}} \xleftarrow{\$} \text{P}^{\text{or}}((x, \text{crs}^{\text{aux}}), \omega, \text{crs}^{\text{or}})$	$d \xleftarrow{\$} \text{V}^{\text{or}}((x, \text{crs}^{\text{aux}}), \pi, \text{crs}^{\text{or}})$
$\text{crs} \leftarrow \text{crs}^{\text{or}} \parallel \text{crs}^{\text{aux}}$	$\pi \leftarrow \pi^{\text{or}}$	return d
return crs	return π	

FIGURE 4.5 — SZK-FLS-Transformation for multi-theorem NISZK argument (additional input 1^λ omitted for P^{or} and V^{or} for space reasons).

For our simulator we can thus generate a perfectly distributed common random string by picking s randomly, computing $G(s)$, and randomly flipping the hardcore bits:

$$\text{crs}^{\text{aux}} \leftarrow G(s) \text{ xor } 0^{|s|} \parallel t$$

where each bit $t_i \xleftarrow{\$} \{0, 1\}$ in $t = t_1 \parallel \dots \parallel t_{|s|}$ is chosen uniformly and independently. Unless all t_i 's are 0—which happens with probability $2^{-|s|}$ —this gives the simulator a witness for crs^{aux} not being pseudorandom in form of s, t . If $t = 0^{|s|}$ then we let the simulator abort. This unlikely event of all t_i 's being 0 causes our simulator to be statistical zero-knowledge instead of being perfect zero-knowledge.

For the malicious prover in the soundness game we will hand over a pseudorandom string $G(s)$ instead of a truly random one. For the bounded prover this is computationally indistinguishable. But then the prover does not have a witness for the or-part and would thus need to break soundness of the other protocol part for $x \notin \mathcal{L}_{\mathcal{R}}$. This step preserves any exclusive soundness notion but not penalizing soundness, because we need to be able to detect diverging success behavior of the prover in the two cases (which we may not necessarily be able to in the penalizing setting since we cannot check if x is in the language or not).

Below we formally define the augmented language $\mathcal{L}_{\mathcal{R}}^{\text{or}}$ as

$$\mathcal{L}_{\mathcal{R}}^{\text{or}} = \left\{ (x, y) \mid \exists \omega : (x, \omega) \in \mathcal{R} \vee \exists s, t \in \{0, 1\}^{\lfloor |y|/2 \rfloor} : y = G(s) \text{ xor } 0^{|s|} \parallel t, t \neq 0^{|s|} \right\}$$

and the corresponding relation \mathcal{R}^{or} accordingly. Note that this is an NP-relation such that, if we have any single-theorem statistical NIZK for general NP-relations, then we also have an multi-theorem statistical witness-indistinguishable argument for this relation \mathcal{R}^{or} .

For pseudorandomness of G we consider for any probabilistic polynomial-time algorithm \mathcal{D} the probability that $\mathcal{D}(1^\lambda, y_{b'}) = b'$ where the probability is taken over $b' \xleftarrow{\$} \{0, 1\}$, $y_0 \leftarrow G(s)$ for $s \xleftarrow{\$} \{0, 1\}^\lambda$, $y_1 \xleftarrow{\$} \{0, 1\}^{2\lambda}$. Let $\text{Adv}_{G, \mathcal{D}}^{\text{PRG}}(1^\lambda) := \Pr[\mathcal{D}(1^\lambda, y_{b'}) = b'] - \frac{1}{2}$ be \mathcal{D} 's advantage. We say that G is a pseudorandom generator if for any probabilistic polynomial-time algorithm \mathcal{D} this advantage is negligible. Note that the Blum-Micali-Yao generator based on a one-way permutation f achieves this property.

Construction 4.1 (SZK-FLS-Transformation). Let \mathcal{R} be an NP-relation. Let f be a one-way permutation and $\Pi^{\text{or}} = (\text{Setup}^{\text{or}}, \text{P}^{\text{or}}, \text{V}^{\text{or}})$ be a multi-theorem non-interactive statistical witness-indistinguishable argument for the NP-relation \mathcal{R}^{or} . We construct a multi-theorem non-interactive statistical zero knowledge argument $\Pi = (\text{Setup}, \text{P}, \text{V})$ for \mathcal{R} as follows (see also Figure 4.5):

CRS: We define the sampling algorithm $\text{Setup}(1^\lambda)$ for the common random string crs for our construction as

$$\text{Setup}(1^\lambda) = \text{Setup}^{\text{or}}(1^\lambda) \parallel U_{2\lambda},$$

where U_{2n} is the uniform distribution on all $2n$ -bit strings.

Prover: The prover P , receiving 1^λ , $\text{crs} = \text{crs}^{\text{or}} \parallel \text{crs}^{\text{aux}}$, x and ω (for \mathcal{R}) as input, uses $(x, \text{crs}^{\text{aux}})$ and ω for the augmented relation \mathcal{R}^{or} and computes a witness-indistinguishable proof π^{or} for this NP-relation using the string crs^{or} .

Verifier: The verifier V receives 1^λ , $\text{crs} = \text{crs}^{\text{or}} \parallel \text{crs}^{\text{aux}}$, x , and a proof π^{or} for \mathcal{R}^{or} . The verifier accepts iff $V^{\text{or}}(1^\lambda, (x, \text{crs}^{\text{aux}}), \pi^{\text{or}}, \text{crs}^{\text{or}})$ accepts.

Theorem 4.2. *Let \mathcal{R} be an NP-relation. Assuming that $\Pi^{\text{or}} = (\text{Setup}^{\text{or}}, P^{\text{or}}, V^{\text{or}})$ is a non-interactive statistical single-theorem zero-knowledge argument for \mathcal{R}^{or} and that f is a one-way permutation, the non-interactive argument system $\Pi = (\text{Setup}, P, V)$ in Construction 4.1 is a multi-theorem statistical zero-knowledge argument. Furthermore, if the underlying protocol Π^{or} is (non-adaptively resp. adaptively) exclusively sound, then so is the derived protocol Π ; if Π^{or} is adaptive resp. non-adaptive zero-knowledge, then so is Π .*

Proof. (Perfect) Completeness: Note that the verifier V accepts a genuine proof $\pi^{\text{or}} \xleftarrow{\$} P(1^\lambda, x, \omega, \text{crs})$ for original data $\text{crs} = \text{crs}^{\text{or}} \parallel \text{crs}^{\text{aux}} \xleftarrow{\$} \text{Setup}(1^\lambda)$ and $x \in \mathcal{L}_{\mathcal{R}}$ if and only if V^{or} accepts π^{or} for $(x, \text{crs}^{\text{aux}})$ under crs^{or} . The latter is always true since $x \in \mathcal{L}_{\mathcal{R}}$ such that the pair $(x, \text{crs}^{\text{aux}})$ of the or-relation is also in $\mathcal{L}_{\mathcal{R}^{\text{or}}}$, the output of P is given by the output of P^{or} for valid input, and the verifier V^{or} accepts genuine proofs of P^{or} .

Non-adaptive/Exclusive Soundness: Assume that Π^{or} is non-adaptively/exclusively sound. Our argument to show that Π , too, has this property is as follows. We will first substitute the “real” common random string by one in which the augmented component crs^{aux} is always in the range of the pseudorandom generator G . This will be indistinguishable for the bounded prover P^* such that P^* outputs a valid proof with roughly equal probability for pseudorandom G . In this step we exploit the property of non-adaptive/exclusive soundness that $x \notin \mathcal{L}_{\mathcal{R}}$ is chosen before crs . But then the or-language does not have a witness for either part, such that the malicious prover would have to break (non-adaptive) exclusive soundness of the protocol for \mathcal{R}^{or} .

More formally, let crs be a CRS generated as described above and crs_G an artificial CRS generated as

$$\text{crs}_G \leftarrow \text{Setup}^{\text{or}}(1^\lambda) \parallel G(s),$$

where s is chosen uniformly from $\{0, 1\}^\lambda$. In a first game hop we argue that a successful malicious prover P^* for such a CRS is almost as successful as for a genuine one, that is,

$$\Pr[V(1^\lambda, x, \pi, \text{crs}) = 1] \approx \Pr[V(1^\lambda, x, \pi, \text{crs}_G) = 1]$$

are negligibly close, where the probability is over $(x, \text{st}) \xleftarrow{\$} P^*(1^\lambda)$, $\text{crs} \xleftarrow{\$} \text{Setup}(1^\lambda)$ and $\pi \xleftarrow{\$} P^*(1^\lambda, \text{st}, \text{crs})$ and V 's randomness in the first case, and accordingly over $(x, \text{st}) \xleftarrow{\$} P^*(1^\lambda)$, $\text{crs}_G \xleftarrow{\$} \text{Setup}^{\text{or}}(1^\lambda) \parallel G(s)$, $\pi \xleftarrow{\$} P^*(1^\lambda, \text{st}, \text{crs}_G)$ and V 's randomness in the second case.

We show the indistinguishability by defining a distinguisher \mathcal{D} against the pseudorandom generator G . For security parameter λ the distinguisher receives a string $y \in \{0, 1\}^{2\lambda}$ as input, either picked uniformly at random, or being the output of the pseudorandom generator. The distinguisher then invokes the prover and verifier to decide:

$$\begin{array}{l}
\frac{\mathcal{D}(1^\lambda, y)}{(x, \text{st}) \xleftarrow{\$} \mathbf{P}^*(1^\lambda)} \\
\text{crs}^{\text{or}} \xleftarrow{\$} \mathbf{Setup}^{\text{or}}(1^\lambda) \\
\text{crs} \leftarrow \text{crs}^{\text{or}} \| y \\
\pi \xleftarrow{\$} \mathbf{P}^*(1^\lambda, \text{st}, \text{crs}) \\
\mathbf{return} \mathbf{V}(1^\lambda, x, \pi, \text{crs})
\end{array}$$

We claim that the distinguishing advantage bounds the difference between the two games, where \mathbf{G}_0 is the original soundness game (with output 1 indicating that \mathbf{P}^* has won) and \mathbf{G}_1 describes the game where we use the artificial string crs_G instead. Since the two games correspond syntactically to the cases that the distinguisher receives a random y resp. a pseudorandom y we get:

$$\Pr[\mathbf{G}_0(1^\lambda) = 1] - \Pr[\mathbf{G}_1(1^\lambda)] \leq 2 \cdot \text{Adv}_{G, \mathcal{D}}^{\text{PRG}}(1^\lambda).$$

Next we turn the malicious prover \mathbf{P}^* in \mathbf{G}_1 against non-adaptive/exclusive soundness against the unbounded scheme Π into one of the same type for the augmented scheme Π^{or} . Note that we are guaranteed that \mathbf{P}^* always outputs $x \notin \mathcal{L}_{\mathcal{R}}$ by assumption. Our prover \mathbf{P}_{or}^* against Π^{or} works as follows:

$$\begin{array}{ll}
\frac{\mathbf{P}_{\text{or}}^*(1^\lambda)}{(x, \text{st}) \xleftarrow{\$} \mathbf{P}^*(1^\lambda)} & \frac{\mathbf{P}_{\text{or}}^*(1^\lambda, \text{st}_{\text{or}}, \text{crs}^{\text{or}})}{\text{// } \text{st}_{\text{or}} = (\text{st}, \text{crs}^{\text{aux}})} \\
s \xleftarrow{\$} \{0, 1\}^\lambda & \text{crs} \leftarrow \text{crs}^{\text{or}} \| \text{crs}^{\text{aux}} \\
\text{crs}^{\text{aux}} \leftarrow G(s) & \pi \xleftarrow{\$} \mathbf{P}^*(1^\lambda, \text{st}, \text{crs}) \\
\text{st}_{\text{or}} \leftarrow (\text{st}, \text{crs}^{\text{aux}}) & \mathbf{return} \pi \\
\mathbf{return} ((x, \text{crs}^{\text{aux}}), \text{st}_{\text{or}}) &
\end{array}$$

We first observe that, if \mathbf{P}^* always outputs $x \notin \mathcal{L}_{\mathcal{R}}$, then our prover \mathbf{P}_{or}^* always outputs $(x, \text{crs}^{\text{aux}}) \notin \mathcal{L}_{\mathcal{R}}^{\text{or}}$. This holds as the string crs^{aux} is pseudorandom such that neither condition of the or-language is satisfied. In addition, \mathbf{P}_{or}^* is efficient. Hence, \mathbf{P}_{or}^* is also an admissible attacker against non-adaptive/exclusive soundness, this time against $\mathcal{L}_{\mathcal{R}}^{\text{or}}$.

We conclude that, by the soundness of Π^{or} , the success probability of prover \mathbf{P}_{or}^* must be negligible. But because \mathbf{P}_{or}^* has the same success probability as \mathbf{P}^* in \mathbf{G}_1 it follows that the winning probability of \mathbf{P}^* in \mathbf{G}_1 must also be negligible. Since this success probability is negligibly close to the one of \mathbf{P}^* in \mathbf{G}_0 by the pseudorandomness of G , we derive that \mathbf{P}^* success probability against our derived protocol Π must be negligible.

Adaptive/Exclusive Soundness: The proof in the adaptive case follows exactly as in the non-adaptive case. Only this time \mathbf{P}^* chooses $x \notin \mathcal{L}_{\mathcal{R}}$ after seeing crs . But both the distinguisher \mathcal{D} against the pseudorandomness \mathcal{D} , as well as the prover \mathbf{P}_{or}^* against soundness, can assemble the common random string before \mathbf{P}^* selects x . It follows as before that the probability of \mathbf{P}_{or}^* against adaptive/exclusive soundness of Π^{or} and thus the one of \mathbf{P}^* against Π must be negligible.

Zero Knowledge: The simulator ZKSim works as follows: On input 1^λ it first generates $\text{crs} = \text{crs}^{\text{or}} \parallel \text{crs}^{\text{aux}}$, where $\text{crs}^{\text{or}} \xleftarrow{\$} \text{Setup}^{\text{or}}(1^\lambda)$ and crs^{aux} is sampled as

$$\text{crs}^{\text{aux}} \leftarrow G(s) \text{ xor } 0^{|s|} \parallel t$$

for s, t chosen uniformly from $\{0, 1\}^\lambda$. Note that since f is a permutation this CRS has the same distribution as a truly random string. If $t = 0^{|s|}$ then the simulator immediately aborts. Else it outputs crs as the common random string and (s, t) as state st_{ZKSim} . When receiving a (valid) theorem $x \in \mathcal{L}_{\mathcal{R}}$ the simulator runs the prover P^{or} for \mathcal{R}^{or} on input $1^\lambda, (x, \text{crs}^{\text{aux}}), \text{crs}^{\text{or}}$ and witness (s, t) to generate a proof π^{or} . The state remains unchanged.

By assumption, Π^{or} is single-theorem statistical zero knowledge (either adaptively or non-adaptively secure). Further, by Lemma 2.4 it is single-theorem statistical witness indistinguishable, and by Lemma 2.5 also multi-theorem statistical witness indistinguishable for the same level of adaptiveness. Therefore, whenever ZKSim is able to find a valid $t \neq 0^{|s|}$, the statistical distance between genuine proofs by P^{or} (for witness ω) and proofs by ZKSim resp. P^{or} (with witness (s, t)) is given by a negligible term $\epsilon(\lambda)$ for any distinguisher requesting at most q proofs. As ZKSim fails to derive $t \neq 0^{|s|}$ with probability $2^{-\lambda}$, the overall statistical distance is therefore at most $\epsilon(\lambda) + 2^{-\lambda}$ and thus negligible. Thus, $\Pi = (\text{Setup}, \text{P}, \text{V})$ is multi-theorem statistical zero knowledge. We note that the protocol inherits the notion of zero-knowledge adaptiveness from Π^{or} . \square

We remark that the transformation also preserves adaptive/culpable soundness. For this notion the distinguisher against the pseudorandom generator in the soundness part can check efficiently if the prover's choice x is in the language or not with the help of the witness ω_{guilt} which the prover needs to output, too.

4.2 Adaptive Perfect Zero-Knowledge under Expected Polynomial Time

The construction in the previous section displays a small error in the simulation, even if we would start with a perfect zero-knowledge or witness-indistinguishable argument. The reason is that our simulator may not generate a valid pair (s, t) with $t \neq 0^{|s|}$. However, to preserve perfect zero-knowledge the simulator cannot simply discard such bad pairs, else outputs of the form $G(s)$ would not be hit by the simulator (while a uniformly chosen string may actually be in the range of G).

The solution in the single-theorem case is to use the fact that the event of picking bad t 's is very unlikely, namely, $2^{-\lambda}$. We will now decrease the probability further such that we can safely search for the actual witness ω for the x part in this rare case, without violating polynomial run time on the average. For this let $p_{\mathcal{R}}$ denote the polynomial which bounds the witness length of relation \mathcal{R} . Then we use a pseudorandom generator $G(s)$ as before, but we iterate the one-way permutation f for $p_{\mathcal{R}}(\lambda)$ steps. Now the probability of picking some input $(s, t) \in \{0, 1\}^\lambda \times \{0, 1\}^{p_{\mathcal{R}}(\lambda)}$ with $t = 0^{p_{\mathcal{R}}(\lambda)}$ is $2^{-p_{\mathcal{R}}(\lambda)}$. Given that this happens we let the simulator (later, after having obtained the input x) search through all potential witnesses $w \in \{0, 1\}^{\leq p_{\mathcal{R}}(\lambda)}$ and each time check in polynomial time $q_{\mathcal{R}}(\lambda)$ if $(x, w) \in \mathcal{R}$. The run time of the simulator for the exhaustive search is then bounded from above by $2 \cdot 2^{p_{\mathcal{R}}(\lambda)} \cdot q_{\mathcal{R}}(\lambda)$. But since this step is only executed with probability at most $2^{-p_{\mathcal{R}}(\lambda)}$ the overall run time of the simulator remains polynomial in expectation.

If we assume that the original argument system Π^{or} is perfectly witness indistinguishable for non-adaptively chosen statements, then the derived protocol is perfectly zero-knowledge, with as simulator running in expected polynomial time and holding either a witness s, t for the auxiliary

part or a witness for x to compute the proof. As in the statistical case, the protocol still preserves non-adaptive/exclusive or adaptive/exclusive soundness.

The next step is to extend the above idea to multiple theorems. If we have polynomial many statements x_1, \dots, x_q then we would have to search for all witnesses to simulate the proofs if $t = 0 \dots 0$. But the time to search for all these witnesses by brute force is additive and requires at most $2q \cdot q_{\mathcal{R}}(\lambda) \cdot 2^{p_{\mathcal{R}}(\lambda)}$ many steps. Hence, the expected run time is still polynomial.

We finally remark that our simulator only attains the simple notion of expected polynomial where we average the number of steps over the randomness of the algorithm. It is not known if one can modify the simulator to achieve more robust notions, such as Levin's average-time complexity.

5 A Lattice-Based Construction

The main drawbacks of the previous constructions based on general assumptions is that they are not directly applicable to lattice-based problems because they require a one-way permutation. In this section we therefore present a multi-theorem extension in the common random string using dual-mode commitments, based on the Learning-With-Errors (LWE) assumption. Here, a dual-mode commitment scheme is a commitment scheme that can be either statistically hiding or perfectly binding, depending on how the public key is generated. Further, without the knowledge of a secret key, it is computationally indistinguishable whether a given public key belongs to the statistically hiding or to the perfectly binding version.

5.1 Dual-Mode Commitment Schemes based on Lattices

A (non-interactive) commitment scheme consists of a probabilistic polynomial-time algorithm to generate a public key and another probabilistic polynomial-time algorithm which allows to commit to a message under a public key. The scheme can be statistically-hiding (and computationally-binding), or it can be perfectly-binding (and computationally-hiding). A dual-mode scheme has now two key generation algorithms, one for the statistically-hiding and one for the perfectly-binding case. Furthermore, the output of the two key generation algorithms is computationally indistinguishable. To preserve statistical zero-knowledge we make the additional assumption that the public key output in hiding mode is close to uniform:

Definition 5.1 (Dual-mode Commitment Scheme). A non-interactive commitment scheme

$$\Gamma = (\text{Gen}_H, \text{Gen}_B, \text{Com})$$

is called a *dual-mode commitment scheme* if,

Statistically-Hiding Mode: The scheme $(\text{Gen}_H, \text{Com})$ is a statistically-hiding, computationally-binding commitment scheme. Furthermore, the output of Gen_H is statistically close to the uniform distribution.

Perfectly-Binding Mode: The scheme $(\text{Gen}_B, \text{Com})$ is a perfectly-binding, computationally-hiding commitment scheme.

Indistinguishability of Modes: The random variables Gen_H and Gen_B are computationally indistinguishable.

Note that for a dual commitment scheme, it suffices to show that the scheme is statistically-hiding in the hiding mode, perfectly-binding in the binding mode, and that the modes are computationally indistinguishable. The complementary property of the corresponding mode (with computational guarantees) follows immediately.

For the dual-mode commitments, we will use (a stripped-off version of) the two homomorphic trapdoor functions defined by Gorbunov et al. [GVW15]. As pointed out in [CH19], these two trapdoor functions give rise to a dual-mode commitment scheme. It has been shown in [CH19] that it can be used together with a non-interactive witness-indistinguishable proof system for bounded distance decoding to build non-interactive *designated-verifier* computational zero-knowledge arguments. We will describe this dual-mode commitment scheme now in detail and provide proof sketches based on the security proofs in [GVW15].

The construction of the commitment scheme in [GVW15] itself is based on the SIS problem [Ajt96], stating that for parameters $n, m = \text{poly}(n), q$ and β_{SIS} it is hard to find a short non-zero integer vector u (of length at most β_{SIS}) to a given random $n \times m$ -matrix A over \mathbb{Z}_q such that $Au = 0$. The noteworthy property is that there is also a method to generate an $n \times m$ matrix A over \mathbb{Z}_q together with a trapdoor in a secure way. This is implemented by an algorithm TrapGen , taking $1^n, 1^m$ and q as input. Furthermore, there exists an algorithm $\text{Sam}(1^m, 1^m, q)$ which outputs a “small” matrix $U \in \mathbb{Z}_q^{m \times m}$. As discussed in [GVW15] it holds that A generated by $\text{TrapGen}(1^n, 1^m, q)$ is statistically close to uniform, and that A and $A \cdot U$ (sampled according to Sam) are statistically close to A and a uniform matrix V' . The final ingredient is a fixed and easy to compute matrix $G \in \mathbb{Z}_q^{n \times m}$ for the given parameter which allows us to build the commitment scheme. We can then commit to a value $x \in \mathbb{Z}_q$ for matrix A by computing $A \cdot U + x \cdot G$. Note that since $A \cdot U$ is statistically close to a uniform matrix V' we obtain that x is statistically hidden.

We note that we do not take advantage of the trapdoor property here in our construction, but instead sample a uniform matrix A (in the hiding mode). Moreover, as pointed out in [CH19], the SIS assumption is not necessary either. The LWE assumption suffices for our purpose, since we only need that the mode switching is computationally indistinguishable. Indeed, the same could be already accomplished with Regev’s encryption scheme [Reg05] where one can alter to a lossy mode. We describe the dual-commitment scheme more formally in the following constructions:

Construction 5.2 (Hiding-Mode Commitment Scheme).

Key Generation Gen_H : We sample $A \in \mathbb{Z}_q^{n \times m}$ uniformly and set $\text{pk} \leftarrow A$.

Commitment Com : For input pk and $x \in \mathbb{Z}_q$, we sample $U \leftarrow \text{Sam}(1^m, 1^m, q)$ and return

$$\text{Com}(\text{pk}, x; U) = \text{pk} \cdot U + x \cdot G.$$

To open the commitment, we reveal x and U (or the randomness used to sample U).

Proposition 5.3. *Construction 5.2 is a statistically hiding commitment scheme.*

Proof. As shown in [GVW15], we have that the following two tuples are statistically close:

$$(\text{pk}, x, \text{pk} \cdot U + x \cdot G) \equiv_s (\text{pk}, x, V')$$

where $U \leftarrow \text{Sam}(1^m, 1^m, q)$ and $V' \leftarrow \mathbb{Z}_q^{n \times m}$, i.e., the commitment is statistically indistinguishable from a random matrix. This holds for public keys generated by TrapGen and, since that algorithm’s output is close to uniform, also for the random matrix A . \square

Next we recall from [GVW15] how we can switch to a perfectly-binding mode by assuming the hardness of LWE. This problem states that given a matrix A and $As + e$ for a small error vector e sampled from a distribution χ , recovering s is hard [Reg05].

Construction 5.4 (Binding-Mode Commitment Scheme).

Key Generation Gen_B : We sample $A' \leftarrow \mathbb{Z}_q^{(n-1) \times m}$ uniformly and $s' \xleftarrow{\$} \mathbb{Z}_q^{n-1}$ and set

$$\text{pk} \leftarrow \begin{pmatrix} A' \\ s'A' + e \end{pmatrix},$$

where e is a short “noise vector” sampled from χ .

Commitment Com: The commitment is identical to the one in Construction 5.2.

Proposition 5.5. *Construction 5.4 is a perfectly binding commitment scheme.*

Proof. To show this construction is perfectly binding, it suffices to show that we can uniquely recover x using s . Indeed, if we know s' , we can set $s = (-s', 1)$ and $z = (0, \dots, 0, r)$ and calculate

$$s(\text{pk} \cdot U + x \cdot G)G^{-1}(z) = e \cdot U \cdot G^{-1}(z) + x \cdot \langle s, z \rangle = x \cdot r + e'.$$

Note that G^{-1} is a polynomial-time algorithm whose existence is guaranteed by Lemma 2.2 in [GVW15]. For correctly chosen parameters r and e , this lets us recover x uniquely. Now, as s does not depend on x or U , if for two pairs (x, U) and (x', U')

$$\text{pk} \cdot U + x \cdot G = \text{pk} \cdot U' + x' \cdot G,$$

holds, then we have $x = x'$. □

Proposition 5.6. *Assuming the $\text{LWE}(q, \chi)$ -assumption holds, Constructions 5.2 and 5.4 together form a dual-mode commitment scheme.*

Proof.

We start by showing that the public keys of both schemes are computationally indistinguishable. First, note that all but the last column of matrix A are generated uniformly random (or statistically close to that) for both public keys. Therefore, the problem is equivalent to distinguish between $A's + e$ and v' given A' , where $v' \in \mathbb{Z}_q^n$ is a uniformly random vector and s and e are sampled as described in the scheme. However, this is exactly the decisional LWE problem. By our assumption, the two public keys are therefore indistinguishable.

We have not yet shown that Construction 5.2 is computationally binding and that Construction 5.4 is computationally hiding. However, we argue this follows directly from what we have shown already. Assume Construction 5.4 would not be computationally hiding, i.e., there exists an adversary that, given a public key pk , can distinguish between a commitment to x and x' with notable advantage. However, in this case, we can use this adversary to distinguish the public keys of the schemes, as Construction 5.2 is statistically hiding and no adversary with notable advantage can exist here.

Similarly, assume that Construction 5.2 is not computationally binding. Then, there exists an adversary that, given a public key pk , can generate a commitment c that opens to two values x and x' with non-negligible probability. However, as Construction 5.4 is perfectly binding, we can use such an adversary to distinguish between public keys of the two schemes. □

5.2 SZK-FLS-Transformation based on Lattices

We will now define our multi-theorem transformation based on the dual-mode commitment scheme in the previous section. As before, we will use the FLS-type transform, therefore we only need to define a sampling algorithm for the auxiliary CRS crs^{aux} and an augmented or-relation \mathcal{R}^{or} for this string.

The sampling algorithm $\text{Setup}^{\text{aux}}$ to generate crs^{aux} will just generate uniformly random values representing a public key pk and a commitment c :

$$\text{crs}^{\text{aux}} = (\text{pk}, c) \leftarrow U_{nmq} \times U_{nmq}.$$

Note that a random public key corresponds to the hiding-mode public key.

Technically the public key and the commitment in crs^{aux} are matrices over \mathbb{Z}_q , and not uniform strings as required by the common random string model. However, we can generate random elements in \mathbb{Z}_q from uniform strings by interpreting a random string of length $|q| + \lambda$ as an integer and mapping it to the residue mod q . The statistical distance to a uniform element from \mathbb{Z}_q is then exponentially small. We stress that we can also go “backwards” with this technique. Given a random value $v \in \mathbb{Z}_q$ we can add a random multiple $i \cdot q$ to v for $i \xleftarrow{\$} \{0, 1, \dots, 2^{\lambda-1}\}$ to get an (almost) uniform $|q| + \lambda$ bit string which would map to v again. Hence, from now on we switch between random matrices from \mathbb{Z}_q and uniformly random string whenever convenient.

Our relation will now ask for a given public key pk of the commitment scheme and commitment c , both found in the common random string, if there is a matrix $U \leftarrow \text{Sam}(1^m, 1^m, q)$ resp. randomness u such that $U = \text{Sam}(1^m, 1^m, q; u)$, such that the commitment opens to 1:

$$((\text{pk}, c), u) \in \mathcal{R}^{\text{or}} :\iff U = \text{Sam}(1^m, 1^m, q; u) \wedge c = \text{Com}(\text{pk}, 1; U).$$

Given these two properties we can now use the same construction as for the one-way permutation, only that we use the relation above and the sampler $\text{Setup}^{\text{aux}}$ to generate crs^{aux} . In fact the construction is otherwise identical to the one in Figure 4.5:

Construction 5.7 (SZK-FLS-Dual-Mode-Transformation). Let \mathcal{R} be an NP-relation. Further, let $\Gamma = (\text{Gen}_H, \text{Gen}_B, \text{Com})$ be a non-interactive dual-mode commitment scheme and suppose that $\Pi^{\text{or}} = (\text{Setup}^{\text{or}}, \text{P}^{\text{or}}, \text{V}^{\text{or}})$ be a multi-theorem non-interactive statistical witness-indistinguishable argument for the NP-relation \mathcal{R}^{or} . We construct a multi-theorem non-interactive statistical zero knowledge argument $\Pi = (\text{Setup}, \text{P}, \text{V})$ for \mathcal{R} as in Figure 4.5 with the following exception:

CRS: We define the sampling algorithm $\text{Setup}(1^\lambda)$ for the common random string crs for our construction as

$$\text{Setup}(1^\lambda) = \text{Setup}^{\text{or}}(1^\lambda) \parallel \text{Setup}^{\text{aux}}(1^\lambda).$$

The prover algorithm P and verifier algorithm V are as before.

Theorem 5.8. *Let \mathcal{R} be an NP-relation. Assuming that $\Pi^{\text{or}} = (\text{Setup}^{\text{or}}, \text{P}^{\text{or}}, \text{V}^{\text{or}})$ is a non-interactive statistical single-theorem zero-knowledge argument for \mathcal{R}^{or} and that $\Gamma = (\text{Gen}_H, \text{Gen}_B, \text{Com})$ is a dual-mode non-interactive commitment scheme, the non-interactive argument $\Pi = (\text{Setup}, \text{P}, \text{V})$ in Construction 5.7 is a multi-theorem statistical zero-knowledge argument. Furthermore, if the underlying protocol Π^{or} is (non-adaptively resp. adaptively) exclusively sound, then so is the derived protocol Π ; if Π^{or} is adaptive resp. non-adaptive zero-knowledge, then so is Π .*

Proof. The proof is very close to the one of Theorem 4.2 such that we only sketch the main differences here.

(Perfect) Completeness: It follows as in the one-way permutation case that the honest verifier accepts proofs generated by P for $x \in \mathcal{L}_{\mathcal{R}}$.

Exclusive Soundness: To show exclusive soundness (in the non-adaptive or adaptive case) we first switch the auxiliary string to a randomly sampled binding key $\text{pk} \stackrel{\$}{\leftarrow} \text{Gen}_B(1^\lambda)$ and a 0-commitment $\text{Com}(\text{pk}, 0; U)$, instead of using uniformly random values. Note that we can use two game hops to show that this is computationally indistinguishable from genuine common random strings. In the first hop we replace the random key component in crs^{aux} by a key $\text{pk} \stackrel{\$}{\leftarrow} \text{Gen}_H(1^\lambda)$, which is even statistically close. Then we replace the random commitment component in crs^{aux} by a random commitment to 0, $\text{Com}(\text{pk}, 0; U)$. This is again statistically indistinguishable.

And finally we switch to a binding key $\text{pk} \stackrel{\$}{\leftarrow} \text{Gen}_B(1^\lambda)$ and a 0-commitment under this key. This is computationally indistinguishable by the indistinguishability of the dual-mode key generation. (The additional 0-commitment can be computed easily given a hiding or binding key.) This is where we again use exclusive soundness to turn a malicious prover into a distinguisher against the dual-mode scheme, analogously to the distinguisher against the pseudorandomness of the generator in the one-way permutation case.

We now have an auxiliary string which contains a binding key and a 0-commitment, such that the or-part in the \mathcal{R}^{or} cannot be satisfied. It follows now as before that soundness of the constructed protocol follows from the soundness of the original non-interactive argument.

Zero-Knowledge: For adaptive multi-theorem zero-knowledge we remark that the simulator ZKSim can create the key part in the auxiliary string as a hiding key $\text{pk} \stackrel{\$}{\leftarrow} \text{Gen}_H(1^\lambda)$ and the commitment part as a 1-commitment under pk . Since the key pk and the 1-commitment are statistically close to a uniform strings, the simulator's string crs^{aux} is statistically close to a uniform string. For this string crs^{aux} the simulator can use the randomness of the commitment as a witness. The remaining steps in the proof are identical to the ones in the proof of Theorem 4.2. \square

6 Conclusion

We have shown how to apply the idea of the FLS transformation also for statistical zero-knowledge arguments. Our solution follows the FLS approach of using an or-language to give the simulator the leverage to compute a witness, showing that finding the right auxiliary languages in the statistical case is possible. Let us highlight two important aspects of our transformations.

First, our transformations based on one-way permutations and on lattices work in the common *random* string model and does not require any structure of the CRS. Common *reference* strings have the inherent disadvantage that they have some structure and that one needs to trust the party which generates the string. A prominent example is the discussion about the trustworthiness of the Zcash reference string and follow-up suggestions to use common random strings instead, e.g., [FMMO19]. Of course, a party generating a common *random* string may also impose some trust assumption, as our lattice-based solution with the implicit trapdoor generation algorithm shows. But several measures to thwart attacks can be implemented much easier than for structured strings.

This includes the computation of the string as the output of a hash function, or by xoring common random strings from several sources.

The other aspect we would like to emphasize that our transformations preserve adaptive security for both zero-knowledge and soundness. This does not conflict with black-box impossibility result for such statistical zero-knowledge arguments [AF07; Pas16], because in the course of showing adaptive soundness we have, in passing, encountered a possibility to bypass the impossibility results. A key observation is that one may be able to achieve adaptive soundness and zero-knowledge if one switches to the notion of exclusive soundness. This adaptive/exclusive soundness implies adaptive/culpable soundness and thus suffices for many practical applications.

Acknowledgments

We thank the anonymous reviewers for valuable comments. Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – SFB 1119 – 236615297.

References

- [AB20] Vivek Arte and Mihir Bellare. “Dual-Mode NIZKs: Possibility and Impossibility Results for Property Transfer”. In: *INDOCRYPT 2020*. Ed. by Karthikeyan Bhargavan, Elisabeth Oswald, and Manoj Prabhakaran. Vol. 12578. LNCS. Springer, Cham, Dec. 2020, pp. 859–881. DOI: [10.1007/978-3-030-65277-7_38](https://doi.org/10.1007/978-3-030-65277-7_38).
- [AF07] Masayuki Abe and Serge Fehr. “Perfect NIZK with Adaptive Soundness”. In: *TCC 2007*. Ed. by Salil P. Vadhan. Vol. 4392. LNCS. Springer, Berlin, Heidelberg, Feb. 2007, pp. 118–136. DOI: [10.1007/978-3-540-70936-7_7](https://doi.org/10.1007/978-3-540-70936-7_7).
- [Ajt96] Miklós Ajtai. “Generating Hard Instances of Lattice Problems (Extended Abstract)”. In: *28th ACM STOC*. ACM Press, May 1996, pp. 99–108. DOI: [10.1145/237814.237838](https://doi.org/10.1145/237814.237838).
- [BCC88] Gilles Brassard, David Chaum, and Claude Crépeau. “Minimum Disclosure Proofs of Knowledge”. In: *J. Comput. Syst. Sci.* 37.2 (1988), pp. 156–189. DOI: [10.1016/0022-0000\(88\)90005-0](https://doi.org/10.1016/0022-0000(88)90005-0). URL: [https://doi.org/10.1016/0022-0000\(88\)90005-0](https://doi.org/10.1016/0022-0000(88)90005-0).
- [BDMP91] Manuel Blum, Alfredo De Santis, Silvio Micali, and Giuseppe Persiano. “Noninteractive Zero-Knowledge”. In: *SIAM Journal on Computing* 20.6 (1991), pp. 1084–1118.
- [BFM88] Manuel Blum, Paul Feldman, and Silvio Micali. “Non-Interactive Zero-Knowledge and Its Applications (Extended Abstract)”. In: *20th ACM STOC*. ACM Press, May 1988, pp. 103–112. DOI: [10.1145/62212.62222](https://doi.org/10.1145/62212.62222).
- [BHK15] Mihir Bellare, Dennis Hofheinz, and Eike Kiltz. “Subtleties in the Definition of IND-CCA: When and How Should Challenge Decryption Be Disallowed?” In: *Journal of Cryptology* 28.1 (Jan. 2015), pp. 29–48. DOI: [10.1007/s00145-013-9167-4](https://doi.org/10.1007/s00145-013-9167-4).

- [BM84] Manuel Blum and Silvio Micali. “How to Generate Cryptographically Strong Sequences of Pseudo-Random Bits”. In: *SIAM J. Comput.* 13.4 (1984), pp. 850–864. DOI: [10.1137/0213053](https://doi.org/10.1137/0213053). URL: <https://doi.org/10.1137/0213053>.
- [BSMP91] Manuel Blum, Alfredo De Santis, Silvio Micali, and Giuseppe Persiano. “Noninteractive Zero-Knowledge”. In: *SIAM J. Comput.* 20.6 (1991), pp. 1084–1118. DOI: [10.1137/0220068](https://doi.org/10.1137/0220068). URL: <https://doi.org/10.1137/0220068>.
- [Can+19] Ran Canetti, Yilei Chen, Justin Holmgren, Alex Lombardi, Guy N. Rothblum, Ron D. Rothblum, and Daniel Wichs. “Fiat-Shamir: from practice to theory”. In: *51st ACM STOC*. Ed. by Moses Charikar and Edith Cohen. ACM Press, June 2019, pp. 1082–1090. DOI: [10.1145/3313276.3316380](https://doi.org/10.1145/3313276.3316380).
- [CCRR18] Ran Canetti, Yilei Chen, Leonid Reyzin, and Ron D. Rothblum. “Fiat-Shamir and Correlation Intractability from Strong KDM-Secure Encryption”. In: *EUROCRYPT 2018, Part I*. Ed. by Jesper Buus Nielsen and Vincent Rijmen. Vol. 10820. LNCS. Springer, Cham, Apr. 2018, pp. 91–122. DOI: [10.1007/978-3-319-78381-9_4](https://doi.org/10.1007/978-3-319-78381-9_4).
- [CG15] Pyrros Chaidos and Jens Groth. “Making Sigma-Protocols Non-interactive Without Random Oracles”. In: *PKC 2015*. Ed. by Jonathan Katz. Vol. 9020. LNCS. Springer, Berlin, Heidelberg, Mar. 2015, pp. 650–670. DOI: [10.1007/978-3-662-46447-2_29](https://doi.org/10.1007/978-3-662-46447-2_29).
- [CH19] Geoffroy Couteau and Dennis Hofheinz. “Designated-Verifier Pseudorandom Generators, and Their Applications”. In: *EUROCRYPT 2019, Part II*. Ed. by Yuval Ishai and Vincent Rijmen. Vol. 11477. LNCS. Springer, Cham, May 2019, pp. 562–592. DOI: [10.1007/978-3-030-17656-3_20](https://doi.org/10.1007/978-3-030-17656-3_20).
- [CLW18] Ran Canetti, Alex Lombardi, and Daniel Wichs. *Fiat-Shamir: From Practice to Theory, Part II (NIZK and Correlation Intractability from Circular-Secure FHE)*. Cryptology ePrint Archive, Report 2018/1248. 2018. URL: <https://eprint.iacr.org/2018/1248>.
- [FL16] Prastudy Fauzi and Helger Lipmaa. “Efficient Culpably Sound NIZK Shuffle Argument Without Random Oracles”. In: *CT-RSA 2016*. Ed. by Kazue Sako. Vol. 9610. LNCS. Springer, Cham, Feb. 2016, pp. 200–216. DOI: [10.1007/978-3-319-29485-8_12](https://doi.org/10.1007/978-3-319-29485-8_12).
- [FLS90] Uriel Feige, Dror Lapidot, and Adi Shamir. “Multiple Non-Interactive Zero Knowledge Proofs Based on a Single Random String (Extended Abstract)”. In: *31st FOCS*. IEEE Computer Society Press, Oct. 1990, pp. 308–317. DOI: [10.1109/FSCS.1990.89549](https://doi.org/10.1109/FSCS.1990.89549).
- [FLS99] Uriel Feige, Dror Lapidot, and Adi Shamir. “Multiple NonInteractive Zero Knowledge Proofs Under General Assumptions”. In: *SIAM J. Comput.* 29.1 (1999), pp. 1–28. DOI: [10.1137/S0097539792230010](https://doi.org/10.1137/S0097539792230010). URL: <https://doi.org/10.1137/S0097539792230010>.

- [FLSZ17] Prastudy Fauzi, Helger Lipmaa, Janno Siim, and Michal Zjajac. “An Efficient Pairing-Based Shuffle Argument”. In: *ASIACRYPT 2017, Part II*. Ed. by Tsuyoshi Takagi and Thomas Peyrin. Vol. 10625. LNCS. Springer, Cham, Dec. 2017, pp. 97–127. DOI: [10.1007/978-3-319-70697-9_4](https://doi.org/10.1007/978-3-319-70697-9_4).
- [FMMO19] Prastudy Fauzi, Sarah Meiklejohn, Rebekah Mercer, and Claudio Orlandi. “Quisquis: A New Design for Anonymous Cryptocurrencies”. In: *ASIACRYPT 2019, Part I*. Ed. by Steven D. Galbraith and Shiho Moriai. Vol. 11921. LNCS. Springer, Cham, Dec. 2019, pp. 649–678. DOI: [10.1007/978-3-030-34578-5_23](https://doi.org/10.1007/978-3-030-34578-5_23).
- [FS90] Uriel Feige and Adi Shamir. “Witness Indistinguishable and Witness Hiding Protocols”. In: *22nd ACM STOC*. ACM Press, May 1990, pp. 416–426. DOI: [10.1145/100216.100272](https://doi.org/10.1145/100216.100272).
- [GL07] Jens Groth and Steve Lu. “A Non-interactive Shuffle with Pairing Based Verifiability”. In: *ASIACRYPT 2007*. Ed. by Kaoru Kurosawa. Vol. 4833. LNCS. Springer, Berlin, Heidelberg, Dec. 2007, pp. 51–67. DOI: [10.1007/978-3-540-76900-2_4](https://doi.org/10.1007/978-3-540-76900-2_4).
- [GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. “The Knowledge Complexity of Interactive Proof Systems”. In: *SIAM J. Comput.* 18.1 (1989), pp. 186–208. DOI: [10.1137/0218012](https://doi.org/10.1137/0218012). URL: <https://doi.org/10.1137/0218012>.
- [Gol06] Oded Goldreich. *Foundations of Cryptography: Volume 1*. USA: Cambridge University Press, 2006. ISBN: 0521035368.
- [GOS06] Jens Groth, Rafail Ostrovsky, and Amit Sahai. “Perfect Non-interactive Zero Knowledge for NP”. In: *EUROCRYPT 2006*. Ed. by Serge Vaudenay. Vol. 4004. LNCS. Springer, Berlin, Heidelberg, May 2006, pp. 339–358. DOI: [10.1007/11761679_21](https://doi.org/10.1007/11761679_21).
- [GOS12] Jens Groth, Rafail Ostrovsky, and Amit Sahai. “New Techniques for Noninteractive Zero-Knowledge”. In: *J. ACM* 59.3 (2012), 11:1–11:35. DOI: [10.1145/2220357.2220358](https://doi.org/10.1145/2220357.2220358). URL: <https://doi.org/10.1145/2220357.2220358>.
- [GVW15] Sergey Gorbunov, Vinod Vaikuntanathan, and Daniel Wichs. “Leveled Fully Homomorphic Signatures from Standard Lattices”. In: *47th ACM STOC*. Ed. by Rocco A. Servedio and Ronitt Rubinfeld. ACM Press, June 2015, pp. 469–477. DOI: [10.1145/2746539.2746576](https://doi.org/10.1145/2746539.2746576).
- [HL18] Justin Holmgren and Alex Lombardi. “Cryptographic Hashing from Strong One-Way Functions (Or: One-Way Product Functions and Their Applications)”. In: *59th FOCS*. Ed. by Mikkel Thorup. IEEE Computer Society Press, Oct. 2018, pp. 850–858. DOI: [10.1109/FOCS.2018.00085](https://doi.org/10.1109/FOCS.2018.00085).
- [LNPT20] Benoît Libert, Khoa Nguyen, Alain Passelègue, and Radu Titu. “Simulation-Sound Arguments for LWE and Applications to KDM-CCA2 Security”. In: *ASIACRYPT 2020, Part I*. Ed. by Shiho Moriai and Huaxiong Wang. Vol. 12491. LNCS. Springer, Cham, Dec. 2020, pp. 128–158. DOI: [10.1007/978-3-030-64837-4_5](https://doi.org/10.1007/978-3-030-64837-4_5).

- [LPWW20] Benoît Libert, Alain Passelègue, Hoeteck Wee, and David J. Wu. “New Constructions of Statistical NIZKs: Dual-Mode DV-NIZKs and More”. 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*. Ed. by Anne Canteaut and Yuval Ishai. Vol. 12107. Lecture Notes in Computer Science. Springer, 2020, pp. 410–441. DOI: [10.1007/978-3-030-45727-3_14](https://doi.org/10.1007/978-3-030-45727-3_14). URL: https://doi.org/10.1007/978-3-030-45727-3%5C_14.
- [Pas16] Rafael Pass. “Unprovable Security of Perfect NIZK and Non-interactive Non-malleable Commitments”. In: *Comput. Complex.* 25.3 (2016), pp. 607–666. DOI: [10.1007/s00037-016-0122-2](https://doi.org/10.1007/s00037-016-0122-2). URL: <https://doi.org/10.1007/s00037-016-0122-2>.
- [Ps05] Rafael Pass and Abhi Shelat. “Unconditional Characterizations of Non-interactive Zero-Knowledge”. In: *CRYPTO 2005*. Ed. by Victor Shoup. Vol. 3621. LNCS. Aug. 2005, pp. 118–134. DOI: [10.1007/11535218_8](https://doi.org/10.1007/11535218_8).
- [PS19] Chris Peikert and Sina Shiehian. “Noninteractive Zero Knowledge for NP from (Plain) Learning with Errors”. In: *CRYPTO 2019, Part I*. Ed. by Alexandra Boldyreva and Daniele Micciancio. Vol. 11692. LNCS. Springer, Cham, Aug. 2019, pp. 89–114. DOI: [10.1007/978-3-030-26948-7_4](https://doi.org/10.1007/978-3-030-26948-7_4).
- [Reg05] Oded Regev. “On lattices, learning with errors, random linear codes, and cryptography”. In: *37th ACM STOC*. Ed. by Harold N. Gabow and Ronald Fagin. ACM Press, May 2005, pp. 84–93. DOI: [10.1145/1060590.1060603](https://doi.org/10.1145/1060590.1060603).
- [SW14] Amit Sahai and Brent Waters. “How to use indistinguishability obfuscation: deniable encryption, and more”. In: *46th ACM STOC*. Ed. by David B. Shmoys. ACM Press, May 2014, pp. 475–484. DOI: [10.1145/2591796.2591825](https://doi.org/10.1145/2591796.2591825).
- [Yao82] Andrew Chi-Chih Yao. “Theory and Applications of Trapdoor Functions (Extended Abstract)”. In: *23rd FOCS*. IEEE Computer Society Press, Nov. 1982, pp. 80–91. DOI: [10.1109/SFCS.1982.45](https://doi.org/10.1109/SFCS.1982.45).

Appendix A Multi-theorem NIPZK in the Common Reference String Model

As mentioned in the introduction, there exists a folklore transformation from single-theorem to multi-theorem non-interactive perfect zero-knowledge based on the FLS construction [FLS90; FLS99], which however requires a non-uniform common reference string. We will provide here a formal description according to our terminology for sake of completeness.

The original FLS transformation is not statistical (nor perfect) zero-knowledge, as the simulator always chooses crs to be an image of a pseudo-random generator, which can only have up to 2^n images in $\{0, 1\}^{3n}$. It is therefore not statistically close to uniformly random values. The idea of the folklore transformation is to always use an image of the pseudo-random generator as crs. Then, the crs generated by the simulator is identically distributed to the real crs. Obviously, now even for an honestly-generated crs the second condition of the augmented language (“Is crs^{aux} in the image

of the PRG?”) is always true. However, as the malicious prover is computationally bounded we can safely replace the string by a truly random string which is most likely not in the range of the generator. We will now define the setup algorithm for the scheme as well as the augmented language.

Let $G : \{0, 1\}^n \rightarrow \{0, 1\}^{3n}$ be a pseudo-random generator. The sampling algorithm $\text{Setup}^{\text{aux}}$ to generate crs^{aux} calls the PRG G on a uniformly random seed:

$$\text{crs}^{\text{aux}} \leftarrow G(U_n).$$

Note that crs is indeed not statistically close to a uniformly random string of length $3n$. Each string of length $3n$ only has a negligible probability of having a pre-image under G . It is, however, computationally indistinguishable from a uniformly random string due to the security properties of the pseudo-random generator. The augmented language is now defined identically to the one in the original FLS construction:

$$y \in \mathcal{R}^{\text{or}} : \iff \exists x \in \{0, 1\}^n : G(x) = y.$$

Construction A.1 (Folklore-SKZ-FLS). Let \mathcal{R} be an NP-relation. Further, let G be a pseudo-random generator stretching n -bit inputs to $3n$ -bit outputs for each n , and suppose that $\Pi^{\text{or}} = (\text{Setup}^{\text{or}}, \text{P}^{\text{or}}, \text{V}^{\text{or}})$ is a multi-theorem non-interactive perfect witness-indistinguishable argument for the NP-relation \mathcal{R}^{or} . We construct a multi-theorem non-interactive perfect zero knowledge argument $\Pi = (\text{Setup}, \text{P}, \text{V})$ in the common *reference* string model for \mathcal{R} as in Figure 4.5 with the following exception:

CRS: We define the sampling algorithm $\text{Setup}(1^\lambda)$ for the common *reference* string crs for our construction as

$$\text{Setup}(1^\lambda) = \text{Setup}^{\text{or}}(1^\lambda) \parallel \text{Setup}^{\text{aux}}(1^\lambda).$$

The prover algorithm P and verifier algorithm V are as before.

Theorem A.2. *Let \mathcal{R} be an NP-relation. Assuming that $\Pi^{\text{or}} = (\text{Setup}^{\text{or}}, \text{P}^{\text{or}}, \text{V}^{\text{or}})$ is a non-interactive perfect single-theorem zero-knowledge argument for \mathcal{R}^{or} and that $G : \{0, 1\}^n \rightarrow \{0, 1\}^{3n}$ is a pseudo-random generator, the non-interactive argument $\Pi = (\text{Setup}, \text{P}, \text{V})$ in Construction A.1 is a multi-theorem perfect zero-knowledge argument. Furthermore, if the underlying protocol Π^{or} is (non-adaptively resp. adaptively) exclusively sound, then so is the derived protocol Π ; if Π^{or} is adaptive resp. non-adaptive zero-knowledge, then so is Π .*

Proof. Again, the proof is very close to the one of Theorem 4.2 and of course to the original proof in [FLS90; FLS99] such that we only sketch the main differences here.

(Perfect) Completeness: It follows as in the one-way permutation case that the honest verifier accepts proofs generated by P for $x \in \mathcal{L}_{\mathcal{R}}$.

Exclusive Soundness: In the original FLS construction, the proof for soundness argues that the probability for crs to be an image of the PRG G is negligible and therefore the malicious prover cannot find a proof for it. In our case, this does not work, as we guarantee the (honestly-generated) crs to be in the domain of G . We can, however, first use a game hop to replace crs with a uniformly random string. As the malicious prover is computationally bounded, the security of the pseudo-random generator guarantees that both games are indistinguishable for the prover.

The rest of the proof is then identical to the proof of the original FLS construction. Here, if we ask for non-adaptive soundness, i.e., x is chosen before the (now completely random) crs, then the underlying protocol only needs to be non-adaptive sound as well. If we demand adaptive soundness and x is chosen after the crs, then we also require adaptive soundness of the underlying protocol.

Zero-Knowledge: The proof for (adaptive/non-adaptive) zero-knowledge is identical to the original proof of the FLS construction, with the only difference being that the crs chosen by the simulator is now by construction identically distributed to the honestly-generated one. \square

Note that non-adaptive/penalizing soundness for the protocol follows from our equivalence result in Section 3.2 for non-uniform provers.

A Random Oracle for All of Us

Marc Fischlin Felix Rohrbach Tobias Schmalz

Cryptoplexity, Technische Universität Darmstadt, Germany

www.cryptoplexity.de

{[marc.fischlin](mailto:marc.fischlin@cryptoplexity.de), [felix.rohrbach](mailto:felix.rohrbach@cryptoplexity.de), [tobias.schmalz](mailto:tobias.schmalz@cryptoplexity.de)}@cryptoplexity.de

Abstract

We introduce the notion of a universal random oracle. Analogously to a classical random oracle it idealizes hash functions as random functions. However, as opposed to a classical random oracle which is created freshly and independently for each adversary, the universal random oracle should provide security of a cryptographic protocol against *all* adversaries simultaneously. This should even hold if the adversary now depends on the random function. This reflects better the idea that the strong hash functions like SHA-2 and SHA-3 are fixed before the adversary decides upon the attack strategy.

Besides formalizing the notion of the universal random oracle model we show that the model is asymptotically equivalent to Unruh’s auxiliary-input random oracle model (Crypto 2007). In Unruh’s model the adversary receives some inefficiently computed information about the random oracle as extra input. Noteworthy, while security in the universal random oracle model implies security in the auxiliary-input random oracle model tightly, the converse implication introduces an inevitable security loss. This implies that the universal random oracle model provides stronger guarantees in terms of concrete security. Validating the model we finally show, via a direct proof with concrete security, that a universal random oracle is one-way.

1 Introduction

The random oracle methodology [FS87; BR93] has turned out to be a useful tool to design cryptographic protocols with practical efficiency, while allowing security proofs if one assumes that the hash function behaves ideally. That is, in the security proof one assumes that the involved hash function acts optimally like a random function. The underlying assumption is that, if one later uses a strong hash function like SHA-2 or SHA-3 in practice, then any attack against the protocol must be due to unexpected weakness in the hash function. While the soundness of this approach has been disputed, e.g., [Nie02; GK03; BBP04; CGH04], we have not yet experienced practical schemes showing such weaknesses (i.e., without incorporating an obvious structural shortcoming in this regard).

1.1 The Universal Random Oracle Model

Security in the random oracle model considers executions where the random oracle is chosen when the attack starts, independently of the adversary and its strategy. If one goes back to the original idea of later plugging in SHA-2 or SHA-3, however, the order compared to practical settings is in fact reversed: These hash functions have been designed first and are already available, such that the adversary may actually take advantage of this a-priori knowledge in the attack. In contrast, common security games first fix the adversary and then initialize the random oracle.

At first it seems as if the idea of making the adversary depend on the random oracle would refute the idea of eliminating the presence of any structural weakness of the hash function. But recall that we still consider the random oracle to be a random function, only that the adversary may now depend on this random function. In a sense, the adversary still cannot exploit functional properties of the hash functions, but it may take into account that the actual hash function in the protocol is fixed before the protocol is attacked, or even designed. We call this a *universal* random oracle, because the same random oracle should work against all adversaries.

On a technical level the difference between the two approaches, the classical random oracle model and the universal one, becomes apparent through the usage of the Borel-Cantelli lemma, as done for example in the famous work by Impagliazzo and Rudich [IR89]. The Borel-Cantelli lemma allows to reverse the order of quantifiers in the sense that if for any adversary the probability of an attack for a random oracle is negligible, then there exists an oracle which works against all adversaries. In fact, a random oracle will work almost surely against all adversaries.

Hence, while one could use in principle the Borel-Cantelli lemma to switch from the random oracle model to the universal one, the lemma comes with two disadvantages. First, the final step in the argument, namely that a random function works against all adversaries, only works against the countable class of uniform adversaries (unless one makes further restrictions on the security reduction itself [BLN09; BN13]), and thus excludes non-uniform adversaries. The second disadvantage is that the asymptotic statement of the Borel-Cantelli lemma infringes with the notion of concrete security. But the latter is important for schemes aiming at practicality. Hence, using the Borel-Cantelli lemma in principle indeed allows to go from classical random oracles to universal ones, but comes at a price.

1.2 Defining Universal Random Oracles

Defining the universal random oracle model (UROM) is more challenging than one would envision. A straightforward approach would be to demand that for a random oracle \mathcal{O} no adversary A can win the corresponding security experiment Game (with more than negligible probability ε in the security parameter λ):

$$\mathbb{P}_{\mathcal{O}} \left[\forall A \exists \varepsilon \in \text{negl} \forall \lambda : \mathbb{P}_{\text{Game}} \left[\text{Game}^{A, \mathcal{O}}(\lambda) \leq \varepsilon(\lambda) \right] = 1. \right]$$

However, as we argue this definition appears to be too liberal: We provide an experiment which was secure in this version of the UROM, although the experiment is both intuitively insecure and also provably breakable in the ordinary random oracle model. This would violate our intuition that the UROM provides stronger security guarantees than the ordinary ROM.

The above mismatch also motivates our actual definition of the UROM. As in the random oracle model we aim for security for a given security parameter, such that the quantification over λ appears

outside of the probability over the random oracle:

$$\forall s \in \text{poly} \exists \epsilon \in \text{negl} \forall \lambda : \mathbb{P}_{\mathcal{O}} \left[\forall A \in \text{SIZE}(s(\lambda)) : \mathbb{P}_{\text{Game}} \left[\text{Game}^{A, \mathcal{O}}(\lambda) \leq \epsilon(\lambda) \right] \geq 1 - \epsilon(\lambda) \right].$$

To let the adversary A depend on the random oracle we quantify over all adversaries in the probability for \mathcal{O} , and only use a size bound $s(\lambda)$ on the outside. We give more details about this choice within. Another justification for the correctness of our approach is by relating this model to existing definitions, especially to the auxiliary-input random oracle model.

1.3 Relationship to Auxiliary-Input Random Oracles

Unruh [Unr07] defined the auxiliary-input random oracle model (AI-ROM) as an extension of the classical random oracle model. In this model the adversary A receives as input some information about the previously sampled random oracle \mathcal{O} , provided by some unbounded algorithm $z^{\mathcal{O}}$ with oracle access to the random oracle. This can, for example, be a collision found in exponential time such that random oracles are not collision-resistant in this model.

Unruh’s main technical result, called lazy sampling with auxiliary input [Unr07, Theorem 2], is to relate the statistical distance of outputs for adversaries receiving auxiliary input $z^{\mathcal{O}}$ for random oracle \mathcal{O} , to the one when instead having access to a fresh random oracle (but which is consistent on some fixed values with the original oracle). He shows that the statistical distance between the two settings is of order $\mathcal{O}\left(\sqrt{ST/P}\right)$ where S is the bit size of auxiliary information, T is the number of random oracle queries of the adversary, and P is the number of coordinates to be fixed. The bound was subsequently improved to $\mathcal{O}(ST/P)$ by Coretti et al. [CDGS18], matching a lower bound of Dodis et al. [DGK17].

Here we show that the two models, AI-ROM and UROM, are equivalent. That is, if a game is secure in one model, then it is also secure in the other model. Remarkably, there is a security degradation when going from AI-ROM to UROM: If a game is ϵ -secure in the AI-ROM, then it is “only” $\sqrt{\epsilon}$ -secure in the UROM. We also show that this quadratic loss is inevitable in general. The other direction holds tightly, i.e., ϵ -security in the UROM implies ϵ -security in the AI-ROM. In this sense, the UROM gives stronger security guarantees for concrete bounds.

Another interesting aspect of UROM is that the separation of the game’s randomness from the randomness of choosing the random oracle allows for more freedom in setting the security bounds. The above equivalence of UROM and AI-ROM holds for negligible bounds for both the game’s and the random oracle’s randomness, but the UROM model also allows for notions where the game should have a negligible success probability for any adversary, with all but exponentially-small probability in the selection of the random oracle.

1.4 Relationship to Global Random Oracles

Canetti et al. [CJS14], and later Camenisch et al. [CDGLN18], considered the notion of global random oracles in the Universal Composability (UC) framework [Can01]. The starting point of the global random oracle model is the observation that, even if multiple components of a cryptographic protocol are proven UC secure in the (standard) random oracle model, their composition is not necessarily secure if the random oracle is replaced by the same hash function in all components. The global random oracle model now says that a single, global random oracle functionality is available to

all parties. A security proof in the model allows for one random oracle to be used in all components, and therefore also to be replaced with the same hash function everywhere.

The global random oracle model and the UROM are close in spirit in light of the idea that the same hash function may be used at several places, but are technically somewhat orthogonal. The global random oracle model is investigating cross-effects of hash function deployments between different protocol executions (but a fresh random oracle instance is chosen when considering an attack on the composed setting). In contrast, the UROM is concerned with dependencies of the adversary with respect to the universally available random oracle within an abstract security game. This abstract security game may be compound of several protocols but is not cast in a simulation-based setting like the UC framework.

1.5 Proving Security in the UROM

We finally give an example of how to show security in the UROM, by showing that one-wayness exists in the UROM. Of course, we can immediately transfer any security result from the AI-ROM via the equivalence, however, this example lets us use the above mentioned flexibility in choosing our security bounds to show that one-wayness exists for all but exponentially few (universal) random oracles.

The proof of one-wayness follows the compression technique of Gennaro and Trevisan [GT00]. Similar approaches have also been given for the AI-ROM [DGK17; CDGS18]; our goal here is to exercise security arguments in the UROM model. The line of reasoning is as follows. If there was a successful adversary against the one-wayness of the UROM, then we can compress the random oracle with the help of the adversary. The important point here is that the adversary may depend on the random oracle for this compression, making the approach employable in the UROM. If the adversary is too successful then we can actually compress beyond information-theoretic lower bounds. Of course, we state the latter fact in terms of concrete security in the UROM.

2 Preliminaries

In this section we present the basic notions of negligible functions, security games, and the (classical) random oracle model, and one-wayness. The notions of the UROM and AI-ROM are given in the subsequent sections.

2.1 Negligible Functions

We use the standard of notion of negligible function and state some very basic but useful properties afterwards:

Definition 2.1 (Negligible Functions). A function $\varepsilon : \mathbb{N} \rightarrow \mathbb{R}$ is called *negligible* if for any polynomial $p : \mathbb{N} \rightarrow \mathbb{R}^+$ there exists $\Lambda \in \mathbb{N}$ such that

$$\forall \lambda \geq \Lambda : \varepsilon(\lambda) \leq \frac{1}{p(\lambda)}.$$

We denote the set of all negligible functions by negl and the set of all functions which are not negligible by non-negl .

Note that we allow negligible functions ε to be negative at some inputs. When quantifying over all negligible functions we can restrict ourselves to non-negative functions by considering the pointwise maximum $\max\{0, \varepsilon(\lambda)\}$. If and only if ε is negligible so is this maximum. Analogously, we can always presume $\varepsilon(\lambda) \leq 1$ when quantifying over all negligible functions. This follows by considering the pointwise minimum $\min\{1, \varepsilon(\lambda)\}$.

When considering definitions we sometimes bound success probabilities for a sequence of events E_λ (e.g., an adversary winning a security game for parameter λ), by negligible functions:

$$\exists \varepsilon \in \text{negl} \forall \lambda \in \mathbb{N} : \mathbb{P}[E_\lambda] \leq \varepsilon(\lambda).$$

Note that we can quantify over all λ since we can change the negligible function ε at finitely many points. When negating this statement, e.g., when describing that there exists a successful adversary, we get

$$\forall \varepsilon \in \text{negl} \exists \lambda \in \mathbb{N} : \mathbb{P}[E_\lambda] > \varepsilon(\lambda). \quad (2.1)$$

This means that for any (negligible) bound we find a security parameter where the probability of the event, e.g., the adversary winning, exceeds this bound. When showing our relationship of the UROM to the AI-ROM we use that the above holds if and only if

$$\forall \varepsilon \in \text{negl} \exists \lambda \in \mathbb{N} : \mathbb{P}[E_\lambda] > \varepsilon^2(\lambda). \quad (2.2)$$

To see this note that we may only consider non-negative functions ε bounded from above by 1. But then the function $\delta(\lambda) := \sqrt{\varepsilon(\lambda)}$ is well defined and it holds that $\delta(\lambda) \geq \varepsilon(\lambda)$ for all security parameters. Furthermore, δ is negligible if and only if ε is.

Hence, when quantifying over all negligible functions we can always switch between ε and δ and get the desired bound. More precisely, assume that statement (2.1) holds. Take some arbitrary negligible function ε . Our goal is to show that there exists some λ such that $\mathbb{P}[E_\lambda] > \varepsilon^2(\lambda)$. But this follows straightforwardly from the first statement since $\varepsilon(\lambda) \geq \varepsilon^2(\lambda)$. For the converse note that, if statement (2.2) holds, then for any given negligible function ε we can consider the function $\delta(\lambda) = \sqrt{\varepsilon(\lambda)}$ in the second statement. By assumption there exists some λ where the probability exceeds $\delta^2(\lambda) = \varepsilon(\lambda)$. This shows that the first statement holds in this case as well for any negligible function.

2.2 Security Games

We consider abstract security games involving the adversary A and an oracle \mathcal{O} . We denote by $\text{Game}^{A, \mathcal{O}}(\lambda)$ the binary outcome of executing the security game. We often emphasize the dependency of algorithms and functions by using subscripts, e.g., we write $A_{\mathcal{O}}$ to denote the fact that the adversary may depend on oracle \mathcal{O} , or $\varepsilon_{A, \mathcal{O}}(\lambda)$ to indicate that the function ε depends on both the adversary and the oracle. We use the terms security game and experiment interchangeably.

Further, by $A^{\mathcal{O}}$, we denote that the adversary A has oracle-access to \mathcal{O} . We view A as a (family of) circuit(s) that have a special oracle gate, which allows A to query the oracle. We capture adversaries with an upper bound $s(\lambda)$ of the size for non-uniform adversaries resp. run time for uniform adversaries via a set $\text{SIZE}(s(\lambda))$. The set of efficient adversaries is given by $\text{SIZE}(\text{poly})$.

We write $\mathbb{P}_{\text{Game}} \left[\text{Game}^{A_{\mathcal{O}}, \mathcal{O}}(\lambda) \right]$ for the probability that adversary A with access to random oracle \mathcal{O} wins in the security game Game . Here, the probability is over all random choices in the game,

including the randomness of the adversary. The random oracle, however, is fixed at this point and the adversary may depend on \mathcal{O} . The oracle is usually chosen “outside” of the game.

2.3 The Random Oracle Model

A random oracle \mathcal{O} is an oracle that gives access to a truly random function. We assume that for every security parameter λ , oracle \mathcal{O} maps inputs from $\{0, 1\}^*$ or from $\{0, 1\}^{\leq d(\lambda)}$ to outputs from $\{0, 1\}^\lambda$, where $\{0, 1\}^{\leq d(\lambda)}$ denotes the set of strings of bit length at most $d(\lambda)$. For so-called length-preserving random oracles the domain for every security parameter λ is simply $\{0, 1\}^\lambda$, i.e., inputs are of length $d(\lambda) = \lambda$ exactly. In the classical random oracle model we pick the oracle \mathcal{O} as part of the game. In this case we usually write $\mathbb{P}_{\text{Game}, \mathcal{O}}[\text{Game}^{A, \mathcal{O}}(\lambda)]$ for the probability of A winning the corresponding game. Note that here now A usually does not depend on \mathcal{O} beyond the oracle access.

With the above notation we can phrase the (classical) random oracle model as follows.

Definition 2.2 (ROM). An experiment **Game** is secure in the ROM iff

$$\forall A \in \text{SIZE}(\text{poly}) \exists \varepsilon \in \text{negl} \forall \lambda \in \mathbb{N} : \mathbb{P}_{\mathcal{O}, \text{Game}}[\text{Game}^{A, \mathcal{O}}(\lambda)] \leq \varepsilon(\lambda).$$

2.4 One-Wayness in the Random Oracle Model

To define that a random oracle \mathcal{O} immediately gives a one-way function we simply state the security game $\text{Game}_{\text{OW}}^{A, \mathcal{O}}(\lambda)$ as follows. Run $A^{\mathcal{O}}(1^\lambda, \mathcal{O}(x))$ for $x \xleftarrow{\$} \{0, 1\}^\lambda$ to obtain a value x^* . Let the game output 1 if and only if $\mathcal{O}(x) = \mathcal{O}(x^*)$. If we assume length-preserving random oracles, as in Section 5, we necessarily have $x^* \in \{0, 1\}^\lambda$ then:

$$\begin{array}{l} \text{Game}_{\text{OW}}^{A, \mathcal{O}}(\lambda) \\ \hline 1 : x \xleftarrow{\$} \{0, 1\}^\lambda \\ 2 : x^* \xleftarrow{\$} A^{\mathcal{O}}(1^\lambda, \mathcal{O}(x)) \\ 3 : \text{return } 1 \text{ if } \mathcal{O}(x^*) = \mathcal{O}(x) \text{ else } 0 \end{array}$$

Note that if we later switch to the UROM, then the one-wayness security game does not change, only the oracle setting.

3 The Universal Random Oracle Model UROM

A straightforward formalization of the universal ROM may now be to demand that, with probability 1, a random oracle \mathcal{O} is good for all adversaries A . That is, for each adversary the success probability is negligible for the given random oracle:

$$\mathbb{P}_{\mathcal{O}} \left[\begin{array}{l} \forall A_{\mathcal{O}} \in \text{SIZE}(\text{poly}(\lambda)) \\ \exists \varepsilon_{A, \mathcal{O}} \in \text{negl} \forall \lambda \in \mathbb{N} \end{array} : \mathbb{P}_{\text{Game}}[\text{Game}^{A_{\mathcal{O}}, \mathcal{O}}(\lambda)] \leq \varepsilon_{A, \mathcal{O}}(\lambda) \right] = 1.$$

The issue with this approach is that it identifies a case which is both intuitively insecure and also provably insecure in the (plain) random oracle, to be secure in this version of the UROM, as we discuss in Appendix A.

We next present the, in our view, right definition of the UROM. The essence from the above failed definitional attempt is that we have to pull out the quantification of the security parameter

from the outer probability, and therefore all preceding quantifiers. But moving out the quantification over all adversaries would infringe with our idea to make the adversary depend on the random oracle. To re-install this idea we set a bound on the adversarial success probability and run time resp. size, and define “good” random oracles for all adversaries within such bounds:

Definition 3.1 (UROM). A security game Game is secure in the UROM if

$$\forall s \in \text{poly} \exists \varepsilon_s \in \text{negl} \forall \lambda \in \mathbb{N} : \mathbb{P}_{\mathcal{O}} \left[\forall A_{\mathcal{O}} \in \text{SIZE}(s(\lambda)) : \mathbb{P}_{\text{Game}} \left[\text{Game}^{A_{\mathcal{O}}, \mathcal{O}}(\lambda) \right] \leq \varepsilon_s(\lambda) \right] \geq 1 - \varepsilon_s(\lambda).$$

Note that with the “outer” negligible error function ε_s we account for “bad” random oracles, for which the security game may be easy to win, e.g., breaking one-wayness for the all-zero oracle \mathcal{O} . Since we may consider finite domains and ranges for \mathcal{O} for the fixed-size adversaries (for given security parameter λ), such bad random oracles may have a non-zero probability. The negligible function expresses that such oracles are very sparse. As the weakness of the random oracle may depend on the adversarial size, e.g., many hardcoded preimages in the one-wayness experiment may affect the security, we make the negligible function also depend on the size s . The “inner” probability then captures that no adversary (of the given size) can win the security game with high probability, but here the probability is only over all the random choices of the game and adversary.

Both the inner and outer negligible error terms are based on the same function. We could have chosen different negligible functions ε_s (for the inner game probability) and δ_s for the outer oracle probability, and quantify over the existence of such negligible functions ($\exists \varepsilon_s, \delta_s \in \text{negl}$). But the pointwise maximum, $\gamma_s(\lambda) := \max\{\varepsilon_s(\lambda), \delta_s(\lambda)\}$, would also be negligible and satisfy the bounds:

- For $\gamma_s(\lambda) = \delta_s(\lambda) \geq \varepsilon_s(\lambda)$ we have

$$\begin{aligned} 1 - \delta_s(\lambda) &\leq \mathbb{P}_{\mathcal{O}} \left[\forall A_{\mathcal{O}} \in \text{SIZE}(s(\lambda)) : \mathbb{P}_{\text{Game}} \left[\text{Game}^{A_{\mathcal{O}}, \mathcal{O}}(\lambda) \right] \leq \varepsilon_s(\lambda) \right] \\ &\leq \mathbb{P}_{\mathcal{O}} \left[\forall A_{\mathcal{O}} \in \text{SIZE}(s(\lambda)) : \mathbb{P}_{\text{Game}} \left[\text{Game}^{A_{\mathcal{O}}, \mathcal{O}}(\lambda) \right] \leq \delta_s(\lambda) \right]. \end{aligned}$$

- For $\gamma_s(\lambda) = \varepsilon_s(\lambda) \geq \delta_s(\lambda)$ we have

$$\begin{aligned} \mathbb{P}_{\mathcal{O}} \left[\forall A_{\mathcal{O}} \in \text{SIZE}(s(\lambda)) : \mathbb{P}_{\text{Game}} \left[\text{Game}^{A_{\mathcal{O}}, \mathcal{O}}(\lambda) \right] \leq \varepsilon_s(\lambda) \right] &\geq 1 - \delta_s(\lambda) \\ &\geq 1 - \varepsilon_s(\lambda). \end{aligned}$$

It thus suffices to consider a single negligible function.

4 UROM vs. AI-ROM

In this section we show that UROM and AI-ROM are equivalent, although not tightly related. We first present the AI-ROM and then show both directions of the equivalence.

4.1 AI-ROM

The auxiliary-input random oracle model AI-ROM [Unr07] allows a preprocessing through an unbounded oracle algorithm $z^{(\cdot)}$ which on input the security parameter outputs a polynomial-size string. This string is then given as auxiliary information about the random oracle to the adversary A . Experiments in which the adversary needs to find a collision in the (unkeyed) random oracle

are for example insecure in the AI-ROM: The function $z^{\mathcal{O}}(1^\lambda)$ exhaustively searches for a collision $x \neq x'$ of complexity λ and outputs this pair; adversary A simply outputs the collision. This is in contrast to security in the regular ROM in which no efficient A is able to find such a collision with non-negligible probability.

Definition 4.1 (AI-ROM). An experiment **Game** is secure in the AI-ROM iff

$$\forall A, z^{(\cdot)} \exists \varepsilon \in \text{negl} \forall \lambda : \mathbb{P}_{\mathcal{O}, \text{Game}} \left[\text{Game}^{A^{\mathcal{O}}, \mathcal{O}}(\lambda) \right] \leq \varepsilon(\lambda).$$

We will now show the equivalence of the two notions. Section 4.2 will show that AI-ROM implies UROM, and Section 4.3 will show the reverse implication.

4.2 AI-ROM implies UROM

Theorem 4.2 (AI-ROM \Rightarrow UROM). *If **Game** is secure in the AI-ROM then it is also secure in the UROM.*

Proof. Assume an experiment **Game** is insecure in the UROM, i.e., negating the security requirement we have

$$\exists s \in \text{poly} \forall \varepsilon \in \text{negl} \exists \lambda \in \mathbb{N} : \mathbb{P}_{\mathcal{O}} \left[\forall A_{\mathcal{O}} \in \text{SIZE}(s(\lambda)) : \mathbb{P}_{\text{Game}} \left[\text{Game}^{A_{\mathcal{O}}, \mathcal{O}}(\lambda) \right] \leq \varepsilon(\lambda) \right] < 1 - \varepsilon(\lambda).$$

We will show that the experiment is also insecure in the AI-ROM, by constructing an adversary pair (A_{AI}, z) against the auxiliary-input setting. First, to get a better intuition we switch to the complementary event of the outer probability for our successful attack against the UROM:

$$\exists s \in \text{poly} \forall \varepsilon \in \text{negl} \exists \lambda \in \mathbb{N} : \mathbb{P}_{\mathcal{O}} \left[\exists A_{\mathcal{O}} \in \text{SIZE}(s(\lambda)) : \mathbb{P}_{\text{Game}} \left[\text{Game}^{A_{\mathcal{O}}, \mathcal{O}}(\lambda) \right] > \varepsilon(\lambda) \right] \geq \varepsilon(\lambda).$$

This formula now states the the fraction of “bad” random oracles \mathcal{O} for which there exists a successful adversary, exceeding the bound $\varepsilon(\lambda)$ for some parameter λ , cannot be upper bounded by any negligible function $\varepsilon(\lambda)$. We can capture this set $\Omega = \Omega_{s, \varepsilon, \lambda}$ of bad random oracles as

$$\Omega := \left\{ \mathcal{O} \mid \exists A_{\mathcal{O}} \in \text{SIZE}(s(\lambda)) : \mathbb{P}_{\text{Game}} \left[\text{Game}^{A_{\mathcal{O}}, \mathcal{O}}(\lambda) \right] > \varepsilon(\lambda) \right\}$$

Note that all parameters s, ε, λ are fixed via the quantifiers when defining this set. This is especially true for the security parameter λ , so the condition is *not* a statement over all security parameters. To emphasize this we can also write the definition of Ω implicitly as

$$\exists s \in \text{poly} \forall \varepsilon \in \text{negl} \exists \lambda \in \mathbb{N} \forall \mathcal{O} \in \Omega \exists A_{\mathcal{O}} \in \text{SIZE}(s(\lambda)) : \mathbb{P}_{\text{Game}} \left[\text{Game}^{A_{\mathcal{O}}, \mathcal{O}}(\lambda) \right] > \varepsilon(\lambda).$$

Note that for the fixed values s, ε, λ we have $\mathbb{P}_{\mathcal{O}}[\mathcal{O} \in \Omega] \geq \varepsilon(\lambda)$ and therefore

$$\mathbb{P}_{\mathcal{O}, \text{Game}} \left[\text{Game}^{A_{\mathcal{O}}, \mathcal{O}}(\lambda) \right] \geq \mathbb{P}_{\text{Game}} \left[\text{Game}^{A_{\mathcal{O}}, \mathcal{O}}(\lambda) \mid \mathcal{O} \in \Omega \right] \cdot \mathbb{P}_{\mathcal{O}}[\mathcal{O} \in \Omega] > \varepsilon^2(\lambda).$$

Next we define an inefficient function $z_s^{(\cdot)}$ that, given any oracle \mathcal{O} and security parameter λ , outputs (a circuit description of) the adversary $A_{\mathcal{O}}$ of size at most $s(\lambda)$ with the highest success probability against the game. This adversary will win the game with probability more than $\varepsilon(\lambda)$ for any oracle in Ω according to the definition. Further, we define A_{AI} , which is the universal circuit that will interpret the circuit returned by z_s . Note that the universal circuit A_{AI} has polynomial

size, since the size of $A_{\mathcal{O}}$ is bounded by the (fixed) polynomial $s(\lambda)$ and the execution of a circuit can be done efficiently. Therefore we can conclude that there exists a pair (A_{AI}, z_s) , where A_{AI} is polynomially bounded. This pair is successful in the given game for any bad oracle from the set Ω :

$$\exists(A_{\text{AI}}, z_s^{(\cdot)}) \forall \varepsilon \in \text{negl} \exists \lambda \in \mathbb{N} \forall \mathcal{O} \in \Omega : \mathbb{P}_{\text{Game}} \left[\text{Game}^{A_{\text{AI}}^{\mathcal{O}}(z_s^{\mathcal{O}}), \mathcal{O}}(\lambda) \right] > \varepsilon(\lambda).$$

It remains to show that a similar bound also holds when we go back to picking \mathcal{O} at random from *all* random oracles instead of the set Ω . To this end we use our assumption that the probability of an oracle being in Ω is at least $\varepsilon(\lambda)$:

$$\exists(A_{\text{AI}}, z_s^{(\cdot)}) \forall \varepsilon \in \text{negl} \exists \lambda \in \mathbb{N} : \mathbb{P}_{\mathcal{O}, \text{Game}} \left[\text{Game}^{A_{\text{AI}}^{\mathcal{O}}(z_s^{\mathcal{O}}), \mathcal{O}}(\lambda) \right] > \varepsilon^2(\lambda).$$

According to the discussion about negligible functions we can rewrite this as

$$\exists(A_{\text{AI}}, z_s^{(\cdot)}) \forall \varepsilon \in \text{negl} \exists \lambda \in \mathbb{N} : \mathbb{P}_{\mathcal{O}, \text{Game}} \left[\text{Game}^{A_{\text{AI}}^{\mathcal{O}}(z_s^{\mathcal{O}}), \mathcal{O}}(\lambda) \right] > \varepsilon(\lambda).$$

But this means that the protocol cannot be secure in the AI-ROM. \square

In terms of exact security, the derived adversary A_{AI} has roughly the same running time as A . But its success probability drops from $\varepsilon(\lambda)$ (for A) to $\varepsilon^2(\lambda)$. In general this is inevitable, though. For any $k = \omega(\lambda)$ consider the game $\text{Game}^{A, \mathcal{O}}(\lambda)$ which returns 1 if the leading k bits of $\mathcal{O}(0^\lambda)$ are all 0 and if, in addition, k random coin flips also land all on 0. Then the probability that any pair (A_{AI}, z) wins in the AI-ROM setting is at most 2^{-2k} . But in the UROM setting we can set $\varepsilon(\lambda)$ to be 2^{-k} , because for all “good” oracles \mathcal{O} with the leading k bits of $\mathcal{O}(0^\lambda)$ being different from 0 no adversary can win the game.

4.3 UROM implies AI-ROM

Theorem 4.3 (UROM \Rightarrow AI-ROM, tightly). *If Game is secure in the UROM then it is also secure in the AI-ROM.*

For the proof we use the so-called splitting lemma [PS00] which allows to relate the probability of events over a product space $X \times Y$ to the ones when the X -part is fixed:

Lemma 4.4 (Splitting Lemma [PS00]). *Let $\mathcal{D} = \mathcal{D}_X \times \mathcal{D}_Y$ be some product distributions over $X \times Y$. Let $Z \subseteq X \times Y$ be such that $\mathbb{P}_{\mathcal{D}}[(x, y) \in Z] > \varepsilon$. For any $\alpha < \varepsilon$ call $x \in X$ to be α -good if*

$$\mathbb{P}_{y \leftarrow \mathcal{D}_Y} [(x, y) \in Z] > \varepsilon - \alpha.$$

Then we have $\mathbb{P}_{x \leftarrow \mathcal{D}_X} [x \text{ is } \alpha\text{-good}] \geq \alpha$.

Proof (of Theorem 4.3). Assume that we have a successful attacker in the AI-ROM:

$$\exists(A_{\text{AI}}, z^{(\cdot)}) \forall \varepsilon \in \text{negl} \exists \lambda \in \mathbb{N} : \mathbb{P}_{\mathcal{O}, \text{Game}} \left[\text{Game}^{A_{\text{AI}}^{\mathcal{O}}(z^{\mathcal{O}}), \mathcal{O}}(\lambda) \right] > \varepsilon(\lambda).$$

We show that we can build a successful adversary A in the UROM model. We first apply the splitting lemma (Lemma 4.4) for fixed $A_{\text{AI}}, z, \varepsilon, \lambda$. We will only consider such choices which exceed the bound $\varepsilon(\lambda)$. We define the distribution \mathcal{D}_X as the choice of a random oracle \mathcal{O} , and \mathcal{D}_Y as the randomness in the game (for both the game and the adversary), as well as Z as the events in which (A_{AI}, z) wins

the game for the random oracle. This happens with probability at least $\varepsilon(\lambda)$ by assumption. If we now choose α to be $\frac{1}{2}\varepsilon$ then we get that $\mathbb{P}_{\mathcal{O}}[\mathcal{O} \text{ is } \alpha\text{-good}] \geq \frac{1}{2}\varepsilon$. Therefore,

$$\exists(A_{\text{AI}}, z) \forall \varepsilon \in \text{negl} \exists \lambda \in \mathbb{N} : \mathbb{P}_{\mathcal{O}} \left[\mathbb{P}_{\text{Game}} \left[\text{Game}^{A_{\text{AI}}^{\mathcal{O}}(z^{\mathcal{O}}), \mathcal{O}}(\lambda) \right] > \frac{1}{2}\varepsilon \right] \geq \frac{1}{2}\varepsilon$$

As $\frac{1}{2}\varepsilon$ is negligible iff ε is, and since we quantify over all negligible functions, we get

$$\exists(A_{\text{AI}}, z) \forall \varepsilon \in \text{negl} \exists \lambda \in \mathbb{N} : \mathbb{P}_{\mathcal{O}} \left[\mathbb{P}_{\text{Game}} \left[\text{Game}^{A_{\text{AI}}^{\mathcal{O}}(z^{\mathcal{O}}), \mathcal{O}}(\lambda) \right] > \varepsilon(\lambda) \right] \geq \varepsilon(\lambda).$$

Since A_{AI} is polynomially bounded, and $z^{(\cdot)}$ only returns a polynomial-size string, we can view A_{AI} as a circuit of polynomial size $s(\lambda)$. But then we can interpret the AI-ROM adversary pair as consisting of an oracle-dependent component, namely the polynomial-size string $z^{\mathcal{O}}$, and a general part A_{AI} . If we hardcode the string $z^{\mathcal{O}}$ we can write this as a single oracle-dependent adversary $A_{\mathcal{O}}$ of polynomial size $s(\lambda)$. Moving this oracle-dependent algorithm inside the outer probability we obtain:

$$\exists s(\lambda) \in \text{poly} \forall \varepsilon \in \text{negl} \exists \lambda \in \mathbb{N} : \mathbb{P}_{\mathcal{O}} \left[\exists A_{\mathcal{O}} \in \text{SIZE}(s(\lambda)) : \mathbb{P}_{\text{Game}} \left[\text{Game}^{A_{\mathcal{O}}^{\mathcal{O}}, \mathcal{O}}(\lambda) \right] > \varepsilon(\lambda) \right] \geq \varepsilon(\lambda).$$

This shows that there have a successful adversary against the UROM. \square

Remarkably, the reduction here is tight. If we have an adversary A_{AI} and z against the AI-ROM, then we get a successful adversary A against UROM with the same running time (as AI-ROM) and, except for a factor $\frac{1}{2}$, the same success probability. This shows that the AI-ROM and UROM model are qualitatively equivalent. Yet, quantitatively, a security bound in the AI-ROM may be significantly looser than in the UROM (see the discussion after Theorem 4.2). This means that a direct proof in the UROM may yield tighter bounds.

4.4 Advantages of UROM

While AI-ROM and UROM are equivalent as shown in the last two sections, we argue that UROM has some advantages over AI-ROM, as it provides more flexibility in choosing security bounds. By having separate bounds for the selection of the random oracle and the success probability of an adversary in the security game, we can for example demand that a game might only be won with negligible probability, for all but an exponential fraction of random oracles. Or, conversely, we could show that a game is secure for every second random oracle, if we can be reasonably sure that we can use one of the good oracles, while in the AI-ROM, a proof might not be possible at all.

For the former, we will give an example in the next section, showing that UROM is a one-way function for nearly all oracles.

5 Universal Random Oracles are One-way Functions

In this chapter, we will show that random oracles exist in the UROM (more specifically, that the oracle itself is a one-way function). This result serves as an example how to prove security in the UROM, and how to show that a game is secure for all but an exponential fraction of random oracles. Our proof will use the compression technique introduced by Gennaro and Trevisan [GT00], although our notation is closer to the argument by Haitner et al. [HRS15].

We note that similar results exist for the AI-ROM [DGK17; CDGS18], which shows that in the AI-ROM, a one-way function exists which no adversary can invert with probability higher than

$\frac{AT}{2^\lambda} + \frac{T}{2^\lambda}$, where A denotes the size of the non-uniform advice the adversary gets about the oracle, and T denotes the number of queries to the oracle. Obviously, their result could be translated to a security bound in the UROM due to the equivalence of the two notions, but the goal here is to present a proof that directly works in the UROM.

The idea of the proof is that, if we have a successful adversary against the random oracle \mathcal{O} , then we can use this (specific) adversary to compress the oracle \mathcal{O} into a smaller description, contradicting lower bounds for the description size of random oracles. The reason that this works in the UROM is that the compression can of course depend on the random oracle, such that the adversary, too, can depend on \mathcal{O} .

For simplicity reasons, we will assume for this chapter that the random oracle \mathcal{O} is always length-preserving (i.e., $d(\lambda) = \lambda$). Note that the existence of length-preserving one-way functions is equivalent to the existence of general one-way functions [Gol01], so this assumption does not influence the result.

We state the result in terms of exact security, using the general UROM approach where we have different probabilities for the inner and outer probability (for the game hardness resp. for the random oracle):

Theorem 5.1. *Let S be the maximum size of an adversary. Then, in the Universal Random Oracle Model, the random oracle is a one-way function with security bounds $\frac{1}{P}$ and $2^{-\lambda}$ for the inner and outer probability, under the condition that $P \cdot S \leq 2^{\lambda/4}$ and $\lambda > 55$:*

$$\mathbb{P}_{\mathcal{O}} \left[\exists A_{\mathcal{O}} \in \text{SIZE}(S) : \mathbb{P}_{x \leftarrow \{0,1\}^\lambda} [A_{\mathcal{O}}^{\mathcal{O}}(1^\lambda, \mathcal{O}(x)) \in \mathcal{O}^{-1}(\mathcal{O}(x))] > \frac{1}{P} \right] < 2^{-\lambda}$$

The asymptotic version follows as an easy corollary:

Corollary 5.2. *UROM is a one-way function in the UROM model: For every polynomial $s(\lambda)$ bounding the size of an adversary, there exists a negligible function $\epsilon_s(\lambda)$ such that for all security parameters λ ,*

$$\mathbb{P}_{\mathcal{O}} \left[\exists A_{\mathcal{O}} \in \text{SIZE}(s(\lambda)) : \mathbb{P}_{x \leftarrow \{0,1\}^\lambda} [A_{\mathcal{O}}^{\mathcal{O}}(1^\lambda, \mathcal{O}(x)) \in \mathcal{O}^{-1}(\mathcal{O}(x))] > \epsilon_s(\lambda) \right] < 2^{-\lambda}$$

Our compression argument will work as follows: Assuming that the random oracle \mathcal{O} in the UROM is not a one-way function, we will show that we can describe the oracle \mathcal{O} with less bits than should be required for a truly random function. For this we assume that A is deterministic; if it is not, then we can make it deterministic by hard-coding the best randomness. We also assume that A needs to output a preimage of size λ and thus only makes queries of this size; any other queries do not help to find a preimage of λ bits and could be easily answered randomly by A itself. Both of these assumptions only increase the size of A at most by a small, constant factor which does not affect our proof.

We give an encoder algorithm which encodes the entire UROM-oracle \mathcal{O} using the successful adversary A , as well as a decoder algorithm which reconstructs \mathcal{O} without access to the oracle itself, using the shorter output of the encoder only. The code for both algorithms is given in Figure 5.1. The encoder starts by defining the set I of all images y on which $A_{\mathcal{O}}$ is able to find some preimage x . Note that, as $A_{\mathcal{O}}$ is deterministic, for given \mathcal{O} , we can indeed specify if $A_{\mathcal{O}}$ is successful on some input y or not. Further, the encoder creates two initially empty sets: Y , which will contain all the y 's for which we reply on $A_{\mathcal{O}}$ to recover one of y 's preimages (and which we therefore do not have to

Encoder $^{\mathcal{O}}(1^\lambda, A_{\mathcal{O}})$	Decoder $(1^\lambda, Y, Z, A_{\mathcal{O}})$
1 : $I \leftarrow \{y \in \{0, 1\}^\lambda \mid A_{\mathcal{O}}^{\mathcal{O}}(y) \text{ successful}\}$ 2 : $Y, Z \leftarrow \emptyset$ 3 : while $I \neq \emptyset$: 4 : $y \leftarrow \min I$ 5 : $Y \leftarrow Y \cup \{y\}, I \leftarrow I \setminus \{y\}$ 6 : Emulate $A_{\mathcal{O}}^{\mathcal{O}}(y)$: 7 : On \mathcal{O} -query x : 8 : if $\mathcal{O}(x) = y$: 9 : abort emulation with result x 10 : if $\mathcal{O}(x) \in I$: 11 : $I \leftarrow I \setminus \{\mathcal{O}(x)\}$ 12 : return $\mathcal{O}(x)$ 13 : $x \leftarrow A_{\mathcal{O}}^{\mathcal{O}}(y)$ // or x result of abort 14 : $Z \leftarrow Z \cup \{(x', y) \mid \mathcal{O}(x') = y, x' \neq x\}$ 15 : endwhile 16 : $Z \leftarrow Z \cup \{(x', y') \mid y' \notin Y, \mathcal{O}(x') = y'\}$ 17 : return $(Y, Z, A_{\mathcal{O}})$	1 : $\mathcal{O} \leftarrow$ Initialize with Z 2 : while $Y \neq \emptyset$: 3 : $y \leftarrow \min Y$ 4 : Emulate $A_{\mathcal{O}}^{\mathcal{O}}(y)$: 5 : On \mathcal{O} -query x : 6 : if $\mathcal{O}(x) \neq \perp$: 7 : return $\mathcal{O}(x)$ 8 : else 9 : abort emulation with x 10 : $x \leftarrow A_{\mathcal{O}}^{\mathcal{O}}(y)$ // or x in abort 11 : $\mathcal{O}(x) \leftarrow y$ 12 : endwhile 13 : return \mathcal{O}

FIGURE 5.1 — Encoder and Decoder for UROM-oracle \mathcal{O} .

save explicitly); and Z , which will contain all full pairs (x, y) with $\mathcal{O}(x) = y$. Therefore, Y denotes the set for which values we actually compress (by not saving the corresponding x -values).

As long as the set I of invertible images still contains values, the encoder takes the lexicographically smallest value y out of I and adds it to Y . We simply write $\min I$ for this element (line 4). Now, the encoder emulates a run of $A_{\mathcal{O}}$ with y as input and checks for all queries. There are two types of queries we need to take care of: The first one are hitting queries, i.e., queries to \mathcal{O} which return y (line 8). In this case, however, A has already given us the preimage, therefore, we abort the simulation at this point. The second type of queries we have to handle are queries that return values y which are still in I (line 10). To make sure we have no circular dependencies between these values, we remove these values from the set I . After the execution of $A_{\mathcal{O}}$ finishes and found a preimage x , we add all further preimages $x' \neq x$ of y that $A_{\mathcal{O}}$ did not return to Z and continue (line 14). Finally, after the set I has become empty, we add all preimages of $y \notin Y$ (as pairs with the image) to Z (line 16). The encoder eventually returns the sets Y, Z , plus a description of $A_{\mathcal{O}}$.

The decoder, on input Y, Z and $A_{\mathcal{O}}$, starts by initializing \mathcal{O} with all the preimage-image-pairs in Z (line 1). Now, similar to the encoder, the decoder goes through all values in Y in lexicographical order and emulates a run of $A_{\mathcal{O}}$ using the partial definition of \mathcal{O} . Note that at this point, we already have a partial description of \mathcal{O} that consists of all value-image-pairs we got via Z as well as all preimages we reconstructed in previous steps. Therefore, for each query x , the adversary $A_{\mathcal{O}}$ makes to the oracle, we first check if $\mathcal{O}(x) \neq \perp$, i.e., if \mathcal{O} is already defined on that value (line 6). If this is the case, we just return the value saved in \mathcal{O} . However, if this is not the case, we know that the call to \mathcal{O} is a hitting query. The reason is that the encoder would have recognized this case and made sure that the value would have been saved in Z (by potentially removing it from I , see line 11) – except for the case where that query is a hitting query. Therefore, in this case, we can already abort

the simulation with result x (line 9). If none of the queries is a hitting query and we therefore do not abort, then we eventually obtain x from the adversary (line 10), since the encoder has only put y into Y because the adversary is successful for y . Finally the decoder sets $\mathcal{O}(x)$ to y .

Note that the lexicographic order here is rather arbitrary – the important part is that the encoder always knows exactly which partial information the decoder will have when it will try to decode a specific y , so any fixed order on the images is fine.

The decoder will always return the original oracle \mathcal{O} when given the information the encoder returns. However, we still need to argue that the information returned by encoder is actually smaller than a straightforward description of \mathcal{O} .

Lemma 5.3. *Let $A_{\mathcal{O}}$ be a deterministic adversary against the one-wayness of \mathcal{O} of size $S \leq s(\lambda)$. Further, let $A_{\mathcal{O}}$ be successful on a fraction of $\frac{1}{P}$ of all input challenges $x \in \{0, 1\}^{\lambda}$. With probability $1 - 2^{-\lambda-1}$ the encoder algorithm describes \mathcal{O} using at most*

$$2 \log \binom{2^{\lambda}}{a} + (2^{\lambda} - a)\lambda + S$$

bits, where a is defined as $a = \frac{2^{\lambda}}{n^2 \cdot PS}$.

Proof. First note that with probability $1 - 2^{-\lambda}$, oracle \mathcal{O} will have no y such that y has more than λ^2 preimages. To show this we start with the probability that a specific y has more than λ^2 preimages. For this, we model each of the 2^n inputs x as a random variable X_i such that $X_i = 1$ iff this x maps to y . Then the number of preimages is the sum of all X_i , denoted by X . Now, we can use the Chernoff bound for a binomial distribution $B(n, p)$ to bound the probability of y having too many preimages:

$$\mathbb{P}[X \geq (1 + \delta)np] \leq \left[\frac{e^{\delta}}{(1 + \delta)^{(1 + \delta)}} \right]^{np}.$$

Using $n = 2^{\lambda}$, $p = 2^{-\lambda}$ and $\delta = \lambda^2$, we get

$$\mathbb{P}_{\mathcal{O}}[|\mathcal{O}^{-1}(y)| > \lambda^2] \leq \frac{e^{\lambda^2}}{(1 + \lambda^2)^{1 + \lambda^2}} \leq 2^{-\lambda^2}$$

for $\lambda \geq 3$. Therefore, the probability that each value $y \in \{0, 1\}^{\lambda}$ has at most λ^2 preimages is

$$\mathbb{P}_{\mathcal{O}}[\forall y, |\mathcal{O}^{-1}(y)| \leq \lambda^2] \geq 1 - 2^{\lambda}(2^{-\lambda^2}) \geq 1 - 2^{-\lambda-1}.$$

Now that we can assume that the number of preimages of all y is bounded by λ^2 , we know that I , the set of all y on which A is successful, has at least size $\frac{2^{\lambda}}{\lambda^2 \cdot P}$, where $\frac{1}{P}$ is the success probability of $A_{\mathcal{O}}$. Furthermore, $A_{\mathcal{O}}$ makes at most S queries on any input. Hence, for each y the encoder adds to Y , it removes at most S values from I . Therefore, Y has at least size $\frac{2^{\lambda}}{\lambda^2 \cdot PS}$.

We will now encode Y by giving the positions of the values in Y in $\{0, 1\}^{\lambda}$. For this we need $\log \binom{2^{\lambda}}{|Y|}$ bits, since we have at most $\binom{2^{\lambda}}{|Y|}$ such sets of size $|Y|$. Similarly, we can encode the corresponding preimages x in $\{0, 1\}^{\lambda}$ which the encoder found for each $y \in Y$ with the same amount of bits. Denote this set as X . Note that these positions of the x 's in X enables a shorter presentation of the set Z of pairs (x', y') with $y' \notin Y$, which the encoder also outputs. Instead of storing the pairs we only need to go through the values $x' \in \{0, 1\}^{\lambda}$ in lexicographic order, skipping over the values in X , and only store the corresponding values y' in this order. This allows us to recover the

pairs in Z with the help of the positions in X , but now we only to store $(2^\lambda - |Y|)\lambda$ extra bits to represent Z , plus the $\log \binom{2^\lambda}{|Y|}$ bits to encode X . Finally, we need S bits for the description of $A_{\mathcal{O}}$.

Now, the above size corresponds to the size if the adversary has a success probability of exactly $\frac{1}{P}$ and makes exactly S queries for each input. However, for any sensible parameters, this should yield an upper bound on the size of the description for any adversary that makes less queries and is successful on a larger fraction of images (if this is not the case, we can of course always adjust our en- and decoder to initialize I with exactly a P -fraction of all y s and always remove exactly S items from I for every y we add to Y). \square

Proof (for Theorem 5.1). To prove Theorem 5.1, we have to show that for parameters S, P and λ ,

$$\mathbb{P}_{\mathcal{O}} \left[\exists A_{\mathcal{O}} \in \text{SIZE}(S) : \mathbb{P}_{x \leftarrow \{0,1\}^\lambda} [A_{\mathcal{O}}^{\mathcal{O}}(1^\lambda, \mathcal{O}(x)) \in \mathcal{O}^{-1}(\mathcal{O}(x))] > \frac{1}{P} \right] < 2^{-\lambda}.$$

Lemma 5.3 tells us two things: First, that at most a $2^{-\lambda-1}$ fraction of the oracles \mathcal{O} has more than λ^2 preimages for some y . Further, we know that for those oracles with at most λ^2 preimages for each value, if $A_{\mathcal{O}}$ is successful, we can encode the oracle using at most

$$2 \log \binom{2^\lambda}{a} + (2^\lambda - a)\lambda + S$$

bits with $a = \frac{2^\lambda}{\lambda^2 P S}$. Now, however, this means that the number of oracles that can be encoded in this way is at most

$$\binom{2^\lambda}{a}^2 \cdot 2^{(2^\lambda - a)\lambda} \cdot 2^S.$$

Therefore, using $P, S \leq 2^{\lambda/4}$ and $\lambda^2 P S \geq 2$ one can encode only a fraction of

$$\begin{aligned} \frac{\binom{2^\lambda}{a}^2 2^{(2^\lambda - a)\lambda} 2^S}{2^{\lambda 2^\lambda}} &< \frac{\left(\frac{e 2^\lambda}{a}\right)^{2a} 2^S}{2^{a\lambda}} = \frac{e^{2a} 2^{\lambda a} 2^S}{a^{2a}} = \frac{e^{2a} 2^S (\lambda^2 P S)^{2a}}{(2^\lambda)^a} \\ &= 2^S \left(\frac{e^2 \lambda^4 P^2 S^2}{2^\lambda} \right)^{\frac{2^\lambda}{\lambda^2 P S}} < 2^S \left(\frac{e^2 \lambda^4 P^2 S^2}{2^\lambda} \right)^{2^{\frac{\lambda}{2}}} \\ &< 2^{\frac{\lambda}{4}} \left(\frac{e^2 \lambda^4 2^{\frac{\lambda}{2}}}{2^\lambda} \right)^{2^{\lambda/2}} < \left(\frac{8 \lambda^4 2^{2^{-\lambda/4}} 2^{\frac{\lambda}{2}}}{2^\lambda} \right)^{2^{\frac{\lambda}{2}}} < \left(\frac{8 \lambda^4 2^{\frac{\lambda}{2} + 1}}{2^\lambda} \right)^{2^{\frac{\lambda}{2}}} \\ &< 2^{-\lambda-1} \text{ for } \lambda \geq 55. \end{aligned}$$

of all $2^{\lambda 2^\lambda}$ oracles.

In summary, $A_{\mathcal{O}}$ can invert either those oracles \mathcal{O} that have some y with more than λ^2 preimages (which happens with probability at most $2^{-\lambda-1}$), or those that can be encoded as above (which is a fraction of $2^{-\lambda-1}$). Note that both bounds are independent of the choice of S and P . Therefore, the probability that a random oracle \mathcal{O} is invertible with more than probability $\frac{1}{P}$ is bounded by $2^{-\lambda}$:

$$\begin{aligned} \mathbb{P}_{\mathcal{O}} \left[\exists A_{\mathcal{O}} \in \text{SIZE}(s(\lambda)) : \mathbb{P}_{x \leftarrow \{0,1\}^\lambda} [A_{\mathcal{O}}^{\mathcal{O}}(1^\lambda, \mathcal{O}(x)) \in \mathcal{O}^{-1}(\mathcal{O}(x))] > \frac{1}{P} \right] \\ < 2^{-\lambda-1} + 2^{-\lambda-1} = 2^{-\lambda}. \end{aligned}$$

This proves the theorem. \square

6 Conclusion

In our paper we have presented an alternative approach to define security for idealized hash functions. Whereas the classical random oracle model assumes that the idealized hash function is specific for each adversary, the UROM model allows arbitrary dependencies of the adversary on the random oracle. This appears to be a natural and necessary generalization of the ROM when instantiating the random oracle with known hash functions like SHA-2 or SHA-3. Our UROM has been defined in light of this idea.

Once we had carved out our model, we could evaluate it. We thus related our definition to Unruh’s auxiliary-input random oracle model. There, the dependency of the adversary on the random oracle is defined by an unbounded preprocessing stage, giving a polynomial-sized advice to the adversary. We then proved our security notion equivalent to AI-ROM which further solidifies the validity of our UROM definition and, vice versa, also means that the AI-ROM provides strong security guarantees. Remarkably, the security bounds are not tightly related.

One of the differences between the UROM and the AI-ROM, and potentially one of the advantages of the UROM, is that our model allows for more flexibility concerning the sources of insecurities. Specifically, in our model one can separately fine-tune the probabilities for the random oracle and the random choices of the adversary. For instance, one could go so far and simply ask for a non-zero probability for a good random oracle, still stipulating a negligible success probability for the adversary. One could then argue, or hope, that SHA-2 or SHA-3 is indeed one of these good random oracles to provide strong security against all adversaries.

An interesting aspect may be to transfer the UROM or the AI-ROM to the UC setting and the global random oracle model. As mentioned before, the global random oracle model and the idea of having an adversarial dependency on the random oracle (as in UROM and AI-ROM) are incomparable. In principle, however, it should be possible to consider a universal random oracle in the global UC setting as well. Given the subtleties in the simpler game-based setting for defining the UROM, we expect this to be far from trivial, though.

Acknowledgments

We thank the anonymous reviewers for valuable comments. Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – SFB 1119 – 236615297 and by the German Federal Ministry of Education and Research and the Hessian Ministry of Higher Education, Research, Science and the Arts within their joint support of the National Research Center for Applied Cybersecurity ATHENE.

References

- [BBP04] Mihir Bellare, Alexandra Boldyreva, and Adriana Palacio. “An Uninstantiable Random-Oracle-Model Scheme for a Hybrid-Encryption Problem”. In: *EUROCRYPT 2004*. Ed. by Christian Cachin and Jan Camenisch. Vol. 3027. LNCS. Springer, Berlin, Heidelberg, May 2004, pp. 171–188. DOI: [10.1007/978-3-540-24676-3_11](https://doi.org/10.1007/978-3-540-24676-3_11).

- [BLN09] Ahto Buldas, Sven Laur, and Margus Niitsoo. “Oracle Separation in the Non-uniform Model”. In: *ProvSec 2009*. Ed. by Josef Pieprzyk and Fangguo Zhang. Vol. 5848. LNCS. Springer, Berlin, Heidelberg, Nov. 2009, pp. 230–244. DOI: [10.1007/978-3-642-04642-1_19](https://doi.org/10.1007/978-3-642-04642-1_19).
- [BN13] Ahto Buldas and Margus Niitsoo. “Black-Box Separations and Their Adaptability to the Non-uniform Model”. In: *ACISP 13*. Ed. by Colin Boyd and Leonie Simpson. Vol. 7959. LNCS. Springer, Berlin, Heidelberg, July 2013, pp. 152–167. DOI: [10.1007/978-3-642-39059-3_11](https://doi.org/10.1007/978-3-642-39059-3_11).
- [BR93] Mihir Bellare and Phillip Rogaway. “Random Oracles are Practical: A Paradigm for Designing Efficient Protocols”. In: *ACM CCS 93*. Ed. by Dorothy E. Denning, Raymond Pyle, Ravi Ganesan, Ravi S. Sandhu, and Victoria Ashby. ACM Press, Nov. 1993, pp. 62–73. DOI: [10.1145/168588.168596](https://doi.org/10.1145/168588.168596).
- [Can01] Ran Canetti. “Universally Composable Security: A New Paradigm for Cryptographic Protocols”. In: *42nd FOCS*. IEEE Computer Society Press, Oct. 2001, pp. 136–145. DOI: [10.1109/SFCS.2001.959888](https://doi.org/10.1109/SFCS.2001.959888).
- [CDGLN18] Jan Camenisch, Manu Drijvers, Tommaso Gagliardoni, Anja Lehmann, and Gregory Neven. “The Wonderful World of Global Random Oracles”. In: *EUROCRYPT 2018, Part I*. Ed. by Jesper Buus Nielsen and Vincent Rijmen. Vol. 10820. LNCS. Springer, Cham, Apr. 2018, pp. 280–312. DOI: [10.1007/978-3-319-78381-9_11](https://doi.org/10.1007/978-3-319-78381-9_11).
- [CDGS18] Sandro Coretti, Yevgeniy Dodis, Siyao Guo, and John P. Steinberger. “Random Oracles and Non-uniformity”. In: *EUROCRYPT 2018, Part I*. Ed. by Jesper Buus Nielsen and Vincent Rijmen. Vol. 10820. LNCS. Springer, Cham, Apr. 2018, pp. 227–258. DOI: [10.1007/978-3-319-78381-9_9](https://doi.org/10.1007/978-3-319-78381-9_9).
- [CGH04] Ran Canetti, Oded Goldreich, and Shai Halevi. “The Random Oracle Methodology, Revisited”. In: *J. ACM* 51.4 (July 2004), pp. 557–594. ISSN: 0004-5411. DOI: [10.1145/1008731.1008734](https://doi.org/10.1145/1008731.1008734). URL: <https://doi.org/10.1145/1008731.1008734>.
- [CJS14] Ran Canetti, Abhishek Jain, and Alessandra Scafuro. “Practical UC security with a Global Random Oracle”. In: *ACM CCS 2014*. Ed. by Gail-Joon Ahn, Moti Yung, and Ninghui Li. ACM Press, Nov. 2014, pp. 597–608. DOI: [10.1145/2660267.2660374](https://doi.org/10.1145/2660267.2660374).
- [DGK17] Yevgeniy Dodis, Siyao Guo, and Jonathan Katz. “Fixing Cracks in the Concrete: Random Oracles with Auxiliary Input, Revisited”. In: *EUROCRYPT 2017, Part II*. Ed. by Jean-Sébastien Coron and Jesper Buus Nielsen. Vol. 10211. LNCS. Springer, Cham, Apr. 2017, pp. 473–495. DOI: [10.1007/978-3-319-56614-6_16](https://doi.org/10.1007/978-3-319-56614-6_16).
- [FS87] Amos Fiat and Adi Shamir. “How to Prove Yourself: Practical Solutions to Identification and Signature Problems”. In: *CRYPTO’86*. Ed. by Andrew M. Odlyzko. Vol. 263. LNCS. Springer, Berlin, Heidelberg, Aug. 1987, pp. 186–194. DOI: [10.1007/3-540-47721-7_12](https://doi.org/10.1007/3-540-47721-7_12).
- [GK03] Shafi Goldwasser and Yael Tauman Kalai. “On the (In)security of the Fiat-Shamir Paradigm”. In: *44th FOCS*. IEEE Computer Society Press, Oct. 2003, pp. 102–115. DOI: [10.1109/SFCS.2003.1238185](https://doi.org/10.1109/SFCS.2003.1238185).

- [Gol01] Oded Goldreich. *Foundations of Cryptography: Basic Tools*. Vol. 1. Cambridge, UK: Cambridge University Press, 2001, pp. xix + 372. ISBN: 978-0511546891. DOI: [10.1017/CB09780511546891](https://doi.org/10.1017/CB09780511546891).
- [GT00] Rosario Gennaro and Luca Trevisan. “Lower Bounds on the Efficiency of Generic Cryptographic Constructions”. In: *41st FOCS*. IEEE Computer Society Press, Nov. 2000, pp. 305–313. DOI: [10.1109/SFCS.2000.892119](https://doi.org/10.1109/SFCS.2000.892119).
- [HHR15] Iftach Haitner, Jonathan J. Hoch, Omer Reingold, and Gil Segev. “Finding Collisions in Interactive Protocols - Tight Lower Bounds on the Round and Communication Complexities of Statistically Hiding Commitments”. In: *SIAM J. Comput.* 44.1 (2015), pp. 193–242. URL: <https://doi.org/10.1137/130938438>.
- [IR89] Russell Impagliazzo and Steven Rudich. “Limits on the Provable Consequences of One-Way Permutations”. In: *21st ACM STOC*. ACM Press, May 1989, pp. 44–61. DOI: [10.1145/73007.73012](https://doi.org/10.1145/73007.73012).
- [Nie02] Jesper Buus Nielsen. “Separating Random Oracle Proofs from Complexity Theoretic Proofs: The Non-committing Encryption Case”. In: *CRYPTO 2002*. Ed. by Moti Yung. Vol. 2442. LNCS. Springer, Berlin, Heidelberg, Aug. 2002, pp. 111–126. DOI: [10.1007/3-540-45708-9_8](https://doi.org/10.1007/3-540-45708-9_8).
- [PS00] David Pointcheval and Jacques Stern. “Security Arguments for Digital Signatures and Blind Signatures”. In: *Journal of Cryptology* 13.3 (June 2000), pp. 361–396. DOI: [10.1007/s001450010003](https://doi.org/10.1007/s001450010003).
- [Unr07] Dominique Unruh. “Random Oracles and Auxiliary Input”. In: *CRYPTO 2007*. Ed. by Alfred Menezes. Vol. 4622. LNCS. Springer, Berlin, Heidelberg, Aug. 2007, pp. 205–223. DOI: [10.1007/978-3-540-74143-5_12](https://doi.org/10.1007/978-3-540-74143-5_12).

Appendix A Defining Universal Random Oracles

In this section we present an alternative definition for UROM and argue why it is inappropriate, motivating also our definition of UROM (Definition 3.1 on page 205).

The Naive Approach. We start with the straightforward adoption of the idea to make the adversary depend on the random oracle by splitting the success probabilities for the experiment **Game** and the random oracle \mathcal{O} , stating that the random oracle should work for all adversaries:

A security game **Game** is secure in the *naive UROM* if

$$\mathbb{P}_{\mathcal{O}} \left[\begin{array}{l} \forall A_{\mathcal{O}} \in \text{SIZE}(\text{poly}(\lambda)) \\ \exists \varepsilon_{A, \mathcal{O}} \in \text{negl} \forall \lambda \end{array} : \mathbb{P}_{\text{Game}} \left[\text{Game}^{A_{\mathcal{O}}, \mathcal{O}}(\lambda) \right] \leq \varepsilon_{A, \mathcal{O}}(\lambda) \right] = 1.$$

We next argue that there is a game which is trivially insecure when considered in the plain random oracle model, but provably secure according to naive UROM. This is counterintuitive because we expect universal random oracles to provide stronger security guarantees compared to the classical ROM. Let \mathcal{O} be length-preserving and the domain size of the random oracle be $d(\lambda) = \lambda$. The game is defined as:

$$\text{Game}^{A_{\mathcal{O}}, \mathcal{O}}(\lambda) = 1 \quad : \iff \quad \mathcal{O}(0^\lambda) \equiv 0 \pmod{\lambda^2},$$

where we interpret the λ -bits output of $\mathcal{O}(0^\lambda)$ as an integer between 0 and $2^\lambda - 1$. We ignore here for simplicity that this integer reduced $\text{mod } \lambda^2$ is only statistically close to a random number between 0 and $\lambda^2 - 1$, and from now on calculate with a probability of $\frac{1}{\lambda^2}$ that the experiment **Game** returns 1 and the adversary wins.

First note that this experiment **Game** is insecure in the standard random oracle mode (Definition 2.2 on page 204), because the trivial adversary who does nothing wins with non-negligible probability has a success probability of at least $\frac{1}{\lambda^2}$, where the probability is over the choice of \mathcal{O} only. We next show that it is secure in the naive **UROM**, though. To this end we first negate the security statement of the naive **UROM** and consider the complementary probability. That is, we have to show:

$$\mathbb{P}_{\mathcal{O}} \left[\begin{array}{l} \exists A_{\mathcal{O}} \in \text{SIZE}(\text{poly}(\lambda)) \\ \forall \varepsilon_{A, \mathcal{O}} \in \text{negl} \exists \lambda \end{array} : \mathbb{P}_{\text{Game}} \left[\text{Game}^{A_{\mathcal{O}}, \mathcal{O}}(\lambda) \right] > \varepsilon_{A, \mathcal{O}}(\lambda) \right] = 0.$$

We first note that the experiment is independent of the adversary, such that we can simplify the statement to:

$$\mathbb{P}_{\mathcal{O}} \left[\forall \varepsilon_{\mathcal{O}} \in \text{negl} \exists \lambda : \mathbb{P}_{\text{Game}} \left[\text{Game}^{\mathcal{O}}(\lambda) \right] > \varepsilon_{\mathcal{O}}(\lambda) \right] = 0.$$

Next observe that the experiment is deterministic, once \mathcal{O} is chosen randomly “on the outside”. This means that we can restrict ourselves to negligible functions $\varepsilon_{\mathcal{O}}$ which only take on values 0 and 1, and also drop the probability over **Game** and instead use the output of the game directly:

$$\mathbb{P}_{\mathcal{O}} \left[\forall \varepsilon_{\mathcal{O}} \in \text{negl}, \varepsilon_{\mathcal{O}} : \mathbb{N} \rightarrow \{0, 1\} \exists \lambda : \text{Game}^{\mathcal{O}}(\lambda) > \varepsilon_{\mathcal{O}}(\lambda) \right] = 0.$$

It suffices now to show that, with probability 0 over the choice of \mathcal{O} , experiment **Game** outputs 1 for infinitely many security parameters λ . If the game only outputs 1 finitely often for a fixed oracle \mathcal{O} , say, up to a bound $\Lambda \in \mathbb{N}$, then we can consider the binary-valued negligible function $\varepsilon_{\mathcal{O}}^{\Lambda}(\lambda) = 1$ if $\lambda \leq \Lambda$, and 0 elsewhere. For this function the game’s output would not exceed the bound $\varepsilon_{\mathcal{O}}^{\Lambda}(\lambda)$ for any λ . In other words, it suffices to show that the (deterministic) experiment **Game** outputs 1 for infinitely many security parameters:

$$\mathbb{P}_{\mathcal{O}} \left[\text{for infinitely many } \lambda \in \mathbb{N} : \text{Game}^{\mathcal{O}}(\lambda) = 1 \right] = 0.$$

We next apply the Borel-Cantelli lemma to show that this is indeed the case. Let E_{λ} describe the event that the game is won for security parameter λ . Then $\mathbb{P}_{\mathcal{O}}[E_{\lambda}] = \frac{1}{\lambda^2}$ over the choice of the random oracle \mathcal{O} . Therefore, since the hyperharmonic series converges,

$$\sum_{\lambda=1}^{\infty} \mathbb{P}[E_{\lambda}] < \infty.$$

The Borel-Cantelli lemma now tells us that the probability that infinitely many E_{λ} happen is 0. Therefore, the game is indeed secure in the naive **UROM**.

Towards the sophisticated UROM. Let us recap what goes wrong with the naive approach above. Borel-Cantelli tells us that for a random oracle \mathcal{O} the probabilities of **Game** outputting 1 become small such that the adversary will only be successful on finitely many security parameters (with probability 1). This yields a fundamental, yet from a cryptographic perspective somewhat counterintuitive property of adversaries: *An adversary might be only successful on finitely many*

security parameters (except with probability 0), even though the adversary has a polynomial success probability for each individual security parameter!

The difference to the ordinary random oracle model is that, there, we rather state security in reverse order, i.e., for a given security parameter λ the probability of an adversary breaking the game for random oracle \mathcal{O} is negligible. We would like to resurrect this behavior while preserving the idea of having a universal random oracle. The approach is basically to move out the quantification over all security parameters ($\forall\lambda$) out of the probability for oracle \mathcal{O} . This, however, means that the preceding quantification over the adversary and the negligible function ($\forall A\exists\epsilon\forall\lambda$) needs to be moved outside of $\mathbb{P}_{\mathcal{O}}[\cdot]$ as well. But this infringes with our idea of the universal random oracle model where the adversary may depend on \mathcal{O} . To re-install this property we only move out a bound $s(\lambda)$ on the adversary's size, and still quantify over all adversaries of this maximal size $s(\lambda)$. This yields our definition of the universal random oracle model (Definition 3.1):

$$\forall s \in \text{poly} \exists \epsilon_s \in \text{negl} \forall \lambda \in \mathbb{N} : \mathbb{P}_{\mathcal{O}} \left[\forall A_{\mathcal{O}} \in \text{SIZE}(s(\lambda)) : \mathbb{P}_{\text{Game}} \left[\text{Game}^{A_{\mathcal{O}}, \mathcal{O}}(\lambda) \right] \leq \epsilon_s(\lambda) \right] \geq 1 - \epsilon_s(\lambda).$$

The outer negligible function $\epsilon_s(\lambda)$ now becomes necessary since for fixed λ we only consider oracle \mathcal{O} of restricted input and output size, determined by the size bound of the adversary and the fixed game.

Besides the equivalence to the auxiliary-input random oracle model and the immediate implication that security in this version of the UROM implies security for ordinary random oracles, we can also discuss directly why our counter example for the naive approach is also labeled as insecure. Recall that $\text{Game}^{\mathcal{O}}(\lambda)$ outputs 1 if $\mathcal{O}(0^\lambda) \equiv 0 \pmod{\lambda^2}$. Then for any given parameter λ we have $\mathbb{P}_{\mathcal{O}} \left[\text{Game}^{\mathcal{O}}(\lambda) = 0 \right] \leq 1 - \frac{1}{\lambda^2}$. It follows that there is no negligible bound $\epsilon_s(\lambda)$ such that this probability is at least $1 - \epsilon_s(\lambda)$.