Post-quantum security of cryptographic transformations in the random oracle model

Huang, Y.

Post-Quantum Security of Cryptographic Transformations in the Random Oracle Model

Yu-Hsuan Huang (黃右萱)

Post-Quantum Security of Cryptographic Transformations in the Random Oracle Model

Proefschrift

ter verkrijging van
de graad van doctor aan de Universiteit Leiden,
op gezag van rector magnificus prof.dr. S. de Rijcke,
volgens besluit van het college voor promoties
te verdedigen op woensdag 1 april 2026
klokke 10:00 uur

door

Yu-Hsuan Huang
geboren te Kaohsiung, Taiwan
in 1996

Promotores:
Prof.dr. S.O. Fehr
Prof.dr. R.J.F. Cramer

Promotiecommissie:
Prof.dr. S. Agrawal (Indian Institute of Technology)
Dr. C. Majenz (Technical University Denmark)
Prof.dr. S. Vaudenay (École Polytechnique Fédérale de Lausanne)
Prof.dr.ir. G.L.A. Derks
Prof.dr. R.M. van Luijk

The author of this thesis carried out his PhD research at the Cryptology Group of Centrum Wiskunde & Informatica (CWI) in The Netherlands, and is supported by the Dutch Research Agenda (NWA) project HAPKIDO (Project No. NWA.1215.18.002), which is financed by the Dutch Research Council (NWO).

CWI
Centrum Wiskunde & Informatica

Universiteit Leiden

Post-Quantum Security of Cryptographic Transformations in the Random Oracle Model

Yu-Hsuan Huang

If you reveal your secrets to the wind
you should not blame the wind for revealing them to the trees.

— Kahlil Gibran, Sand and Foam

Illustrations on the front and back covers are designed and drawn by the author.

Contents

List of Publications **ix**

1 Introduction **1**
1.1 Provable Security 1
1.2 Cryptographic Transformations 6
1.3 Our Contributions 12

2 Preliminaries **17**
2.1 Notations 17
2.2 Oracle Algorithms 18
2.2.1 Equivalences of Oracle Algorithms 20
2.2.2 Conditioning of Oracle Algorithms 21
2.2.3 Simulating the Write Access 21

3 Fiat-Shamir and Hash-and-Sign with Aborts **23**
3.1 Introduction 23
3.1.1 Our Contribution 24
3.2 Preliminaries 27
3.2.1 Generalized Fiat-Shamir with Aborts Signatures 27
3.2.2 A variant of the Adaptive Reprogramming Lemma . . . 29
3.3 A Gap in Prior Analyses of FSwA 30
3.4 A UF-CMA-to-UF-NMA Reduction 31
3.4.1 The Statements 32
3.4.2 Proof Strategy 33
3.4.3 From Sign to Prog 35
3.4.4 From Prog to Trans 37
3.4.5 Wrapping up the Proof of Theorem 3.5 40
3.4.6 Classical Bounds 40
3.5 Tighter Bounds in QROM 41
3.5.1 The Statements 42
3.5.2 From Prog to Trans 42
3.5.3 Wrapping up the Proof of Theorem 3.17 48

3.6 Concrete Analysis of Dilithium 49
3.6.1 Controlling the Min-Entropy via the Rank of **A** 49
3.6.2 Numerically Controlling the Rank of **A** 50
3.6.3 Plugging in the Numbers 53
3.6.4 Analytically Controlling ϵ_{sk} 54

4 BUFF Transformations **59**
4.1 Introduction 59
4.1.1 Our Contribution 60
4.1.2 Related Work 64
4.2 Preliminaries 65
4.2.1 (HILL) Entropy 65
4.2.2 Non-Resignability and Φ-Non-Malleability 66
4.2.3 BUFF Transform 68
4.3 On the Impossibility of Non-Resignability 69
4.3.1 Non-Resignability and BUFF Transform in the Plain Model 70
4.3.2 Non-Resignability and the BUFF Transform in the ROM 72
4.3.3 Φ-Non-Malleability in the ROM 74
4.4 Salted BUFF and NR-bot 76
4.4.1 Positive Results 76
4.4.2 Handling Classical Adversaries 78
4.4.3 Handling Quantum Adversaries 81
4.4.4 Negative Results with *Computational* Entropy 85
4.5 BUFF and sNR 86
4.5.1 Secret-key Non-resignability (sNR) 86
4.5.2 The Hide-and-Seek Game 88
4.5.3 BUFF via Random Oracles is sNR 89
4.5.4 Reducing sNR of BUFF to Hide-and-Seek 90
4.5.5 Hide-and-Seek for Random Oracles 94
4.5.6 Wrapping up the Proof of Theorem 4.18 98
4.6 BUFF via Iterative Hash Functions 99
4.6.1 sNR Relative to (Function-like) Oracles 99
4.6.2 Iterative Hash Functions 100
4.6.3 BUFF via Iterative Hash Functions is not sNR 101
4.7 Sandwich BUFF (sBUFF) Transformation 103
4.7.1 Sandwich BUFF via Merkle-Damgård is sNR 103
4.7.2 Reducing sNR of Sandwich BUFF to Hide-and-Seek . . 104
4.7.3 Hide-and-Seek for Merkle-Damgård 111
4.8 Positive Results with Computational Entropy 113
4.8.1 BUFF via RO is sNR, Computationally 113
4.8.2 Sandwich BUFF via MD is sNR, Computationally . . . 115

5 KEM Combiners via Split-key PRFs **117**
5.1 Introduction 117
5.1.1 Our Contributions 118
5.2 Preliminaries 121
5.3 A Generic Adaptive-to-static Compiler 122
5.3.1 Our Result 122
5.3.2 The Technical Core 123
5.3.3 Wrapping up the Proof of Theorem 5.1 126
5.3.4 Applications 126
5.4 Quantum Security of a Split-key PRF 128
5.4.1 Hybrid Security and skPRFs 128
5.4.2 Quantum-security of the skPRF 129
5.4.3 Proof of Theorem 5.7 130

Conclusions **137**

Appendix **139**
A.1 Breaking a Weakened Phi-Non-Malleability 139
A.2 A Modified Measure-and-Reprogram Lemma 140
A.3 Unsimplified Proof of Lemma 4.32 144

Bibliography **165**

Samenvatting **167**

Summary **169**

Acknowledgement **171**

Curriculum Vitae **173**

List of Publications

The content of this thesis is based on the following articles, with all (co)authors, throughout this section except [HC18, HYC20, FHA23], listed alphabetically.

[FFH25] Pouria Fallahpour, Serge Fehr, and Yu-Hsuan Huang. Tighter quantum security for Fiat-Shamir-with-aborts and hash-and-sign-with-retry signatures. Cryptology ePrint Archive, Paper 2025/985, 2025

[FHK25] Serge Fehr, Yu-Hsuan Huang, and Julia Kastner. Sandwich BUFF: Achieving non-resignability using iterative hash functions. In Benny Applebaum and Huijia (Rachel) Lin, editors, *TCC 2025, Part III*, volume 16270 of *LNCS*, pages 235–265. Springer, Cham, December 2025

[DFH+24] Jelle Don, Serge Fehr, Yu-Hsuan Huang, Jyun-Jie Liao, and Patrick Struck. Hide-and-seek and the non-resignability of the BUFF transform. In Elette Boyle and Mohammad Mahmoody, editors, *TCC 2024, Part III*, volume 15366 of *LNCS*, pages 347–370. Springer, Cham, December 2024

[DFHS24] Jelle Don, Serge Fehr, Yu-Hsuan Huang, and Patrick Struck. On the (in)security of the BUFF transform. In Leonid Reyzin and Douglas Stebila, editors, *CRYPTO 2024, Part I*, volume 14920 of *LNCS*, pages 246–275. Springer, Cham, August 2024

[BBD+23] Manuel Barbosa, Gilles Barthe, Christian Doczkal, Jelle Don, Serge Fehr, Benjamin Grégoire, Yu-Hsuan Huang, Andreas Hülsing, Yi Lee, and Xiaodi Wu. Fixing and mechanizing the security proof of Fiat-Shamir with aborts and Dilithium. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part V*, volume 14085 of *LNCS*, pages 358–389. Springer, Cham, August 2023

[DFH22] Jelle Don, Serge Fehr, and Yu-Hsuan Huang. Adaptive versus static multi-oracle algorithms, and quantum security of a split-key PRF. In Eike Kiltz and Vinod Vaikuntanathan, editors, *TCC 2022, Part I*, volume 13747 of *LNCS*, pages 33–51. Springer, Cham, November 2022

During his Ph.D. study and before, the author also (co)authored the following works, which are not included in this thesis.

[FH23] Serge Fehr and Yu-Hsuan Huang. On the quantum security of HAWK. In Thomas Johansson and Daniel Smith-Tone, editors, *Post-Quantum Cryptography - 14th International Workshop, PQCrypto 2023*, pages 405–416. Springer, Cham, August 2023

[CHH+21] Kai-Min Chung, Yao-Ching Hsieh, Mi-Ying Huang, Yu-Hsuan Huang, Tanja Lange, and Bo-Yin Yang. Isogeny-based group signatures and accountable ring signatures in QROM. Cryptology ePrint Archive, Paper 2021/1368, 2021

[CFHL21] Kai-Min Chung, Serge Fehr, Yu-Hsuan Huang, and Tai-Ning Liao. On the compressed-oracle technique, and post-quantum security of proofs of sequential work. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part II*, volume 12697 of *LNCS*, pages 598–629. Springer, Cham, October 2021

[HYC20] Yu-Hsuan Huang, Chih-Kai Yang, and Rong-Jaye Chen. Quadrangle inequality improvement for CSIDH strategy. In *Cryptology and Information Security Conference*, 2020

[HC18] Yu-Hsuan Huang and Rong-Jaye Chen. Simulating quantum algorithm by using singular value decomposition. In *Cryptology and Information Security Conference*, 2018

In addition to the above research articles, the author has contributed to the following technical documents.

[FHA23] Serge Fehr, Yu-Hsuan Huang, and Alessandro Amadori. Literature review - (quantum-safe) cryptographic combiners and hybrid security. Technical report, 2023. available at `https://hapkido.tno.nl/deliverables/literature-review-quantum-safe/`

[BBD+24] Joppe W. Bos, Olivier Bronchain, Léo Ducas, Serge Fehr, Yu-Hsuan Huang, Thomas Pornin, Eamonn W. Postlethwaite, Thomas Prest, Ludo N. Pulles, and Wessel van Woerden. HAWK. Technical report, National Institute of Standards and Technology, 2024. available at `https://csrc.nist.gov/Projects/pqc-dig-sig/round-2-additional-signatures`

Chapter 1

Introduction

1.1 Provable Security

Cryptography has a history spanning several millennia. The cryptographic techniques were originally in rudimentary forms. They mainly consisted of ad-hoc tricks, most (if not all) of which, though becoming more sophisticated over time, did not withstand the test of time: they are eventually "broken" one way or another.

The cryptographic landscape changed with the seminal work of Shannon [Sha49], who took a scientific approach through mathematically defining and proving *information-theoretic security* of a certain cryptographic scheme. The works in late 1970s and early 1980s by Diffie and Hellman [DH76], and by Goldwasser and Micali [GM82] and others, further considered the notion of *computational security* and introduced formal security definitions, respectively. This marks the beginning of modern cryptography, where security is no longer based on the mere absence of known attacks, but on the study of rigorously defined mathematical notions — what we know today as the *provable security paradigm*.

Security definition. To be able to assess whether a given cryptographic scheme is secure, the first and immediate challenge is to formalize the right definition of the intended security notion. This may be accompanied with a *security game*, specifying in which way an adversary is allowed to interact with the cryptographic scheme, and what it means to "break" the scheme.

For a signature scheme $\mathcal{S} = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vrfy})$ as an example, it might be natural to require in the security definition that it is hard for an attacker to forge a valid signature given the public key (only). This is formally captured in the $\mathsf{UF\text{-}NMA}$ security game[1] as in Fig. 1.1. The corresponding security

[1] The acronym stands for *(existential) unforgeability against no-message attacks.*

definition would then require that, for all (computationally bounded) attackers, it can only win the security game with a small probability.

However, an attacker in real life may fool the signer into signing certain messages. This kind of attack is not captured by the above notion of UF-NMA, and it motivates the stronger notion UF-CMA (see Fig. 1.1)[2], where an attacker is given *oracle access* to the signing procedure $\mathsf{Sign}(\mathsf{sk}, \cdot)$; in each query, it sends over a message m, and in return receives a signature $\sigma \leftarrow \mathsf{Sign}(sk, m)$ corresponding to that message. To make the attacker's task non-trivial, its goal here is to instead produce a forgery that is valid under a *fresh* message, i.e. a message that has never been queried to the oracle $\mathsf{Sign}(\mathsf{sk}, \cdot)$.

UF-NMA:	UF-CMA:
1: $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}$	1: $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}$
2: $(m^*, \sigma^*) \leftarrow \mathcal{A}(\mathsf{pk})$	2: $(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(\mathsf{sk}, \cdot)}(\mathsf{pk})$
3: **return** $\mathsf{Vrfy}(\mathsf{pk}, m^*, \sigma^*) \stackrel{?}{=} 1$	3: $\{$let m_i be the ith query to $\mathsf{Sign}(\mathsf{sk}, \cdot)\}$
	4: **return** $\mathsf{Vrfy}(\mathsf{pk}, m^*, \sigma^*) \wedge m^* \notin \{m_i\}_i$

Figure 1.1: Security games UF-NMA and UF-CMA, where the attackers are denoted by $\mathcal{A}$.

Needless to say, different security games may result in differently powerful security definitions. In the above example, any UF-CMA signature scheme is also UF-NMA, but not vice versa — the Lamport signature scheme [Lam79] satisfies UF-NMA security (assuming the underlying one-way function is secure), but does not satisfy UF-CMA in that there is an explicit attack. The bottom line is: setting up proper security definitions has always been a central question in modern cryptography, which is by no means trivial: on one hand, a definition that is too weak may not rule out all relevant attacks; on the other hand, a definition that is un-necessarily strong may end up unachievable at all.

Security reduction. Once a security definition is set-up, one can proceed to prove it for a given scheme. However, a security proof of a scheme often implies resolution of the P vs NP problem, which is out of reach from the current mathematical tools. Therefore, oftentimes security proofs are conditional, of the following form: if a computational problem $\mathcal{P}$ is hard, then the considered scheme is secure (i.e. satisfying its corresponding security definition).[3]

For example, to prove security of a signature scheme, one would show that every attacker $\mathcal{A}$ winning the UF-CMA security game with non-negligible probability can be turned into a similarly efficient algorithm $\mathcal{B}$ that solves a computational problem $\mathcal{P}$ (again with non-negligible probability/advantage depending

[2]The acronym stands for *(existential) unforgeability against chosen-message attacks*

[3]Sometimes, there are multiple computational problems whose hardness is being relied on, but for the simplicity of exposition we did not bring this up here.

on $\mathcal{P}$ being a search/decision problem) that we believe to be hard. The quality of a reduction depends on the resources taken by $\mathcal{B}$ (such as the runtime $\mathsf{TIME}(\mathcal{B})$, or the number of queries that is made to the signing oracle), and the success probability $\mathbf{Adv}^{\mathcal{P}}(\mathcal{B})$ of $\mathcal{B}$ solving $\mathcal{P}$ — the less resources $\mathcal{B}$ takes and the higher probability $\mathcal{B}$ succeeds in solving $\mathcal{P}$, the stronger security is provided by such a security reduction. This is often quantitatively discussed, and the precise form may vary, such as the following hypothetical examples:

$$\mathbf{Adv}^{\mathsf{UF\text{-}CMA}}_{\mathcal{S}}(\mathcal{A}) \leq q_S \cdot \mathbf{Adv}^{\mathcal{P}}(\mathcal{B}) \ , \quad \mathbf{Adv}^{\mathsf{UF\text{-}CMA}}_{\mathcal{S}}(\mathcal{A}) \leq \sqrt{\mathbf{Adv}^{\mathcal{P}}(\mathcal{B})}$$
$$\mathbf{Adv}^{\mathsf{UF\text{-}CMA}}_{\mathcal{S}}(\mathcal{A}) \leq \mathbf{Adv}^{\mathcal{P}}(\mathcal{B}) + q_S \cdot \epsilon \ ,$$

where $\mathsf{TIME}(\mathcal{B}) \approx \mathsf{TIME}(\mathcal{A})$, q_S is the number of chosen-message queries $\mathcal{A}$ makes, and $\epsilon > 0$ is a quantity determined by the scheme $\mathcal{S}$. In each of these examples, there may be a gap between the theoretical guarantee obtained from the reduction, and the actual security of $\mathcal{S}$. We informally refer to such a gap as the *security loss*.

The point here is that, with security reductions, one can relate the security of a cryptographic scheme, with the hardness of better-understood computational problems, and thereby increase the confidence of the considered scheme. Moreover, the smaller security loss one has in a reduction, the better security guarantee one would get.

Random oracle model. For a cryptographic scheme that relies on a hash function H, ideally one can prove security of the scheme based on a simple computational assumption on the hash function, such as collision resistance. However, in many cases, the security requires that the hash function itself behaves as if it is a random function. Namely, every hash value looks random and unpredictable unless explicitly computed. Such an intuitive requirement is much more elusive, as it cannot be formalized as a property of H in the standard model (also known as the *plain model*).

To mitigate this difficulty, H is often idealized, treated as a random function (which we call a random oracle) chosen uniformly at the beginning of each relevant security game,[4] where an adversary is allowed to make oracle queries to H in order to learn its outputs on any inputs. This is commonly known as the *random oracle model* (ROM) today. Initially, the ROM was introduced in the context of computational complexity theory, without having any hash function in mind [BG81]. Soon after, it found relevance in cryptography, both for proving security of concrete constructions [FS87], and for establishing impossibility results [IR90]. In 1993, Bellare and Rogaway further popularized the idea of proving security of cryptographic schemes in the ROM [BR93].

[4] One may also consider a more fine-grained setting, where only a certain component of the hash function is idealized.

On one hand, the ROM facilitates provable security of many more efficient cryptographic constructions. In this model, a security reduction $\mathcal{B}$ is by default in charge of simulating the random oracle that interacts with the considered attacker $\mathcal{A}$. This is of great advantage for the reduction. For instance, $\mathcal{B}$ may observe $\mathcal{A}$'s queries to H, so, intuitively, it knows which hash values $\mathcal{A}$ knows and which ones not. On the other hand, modelling H as a random oracle is, after all, still a heuristic. For certain constructions, this heuristic is not sound. Specifically, there have been contrived separating examples that are provably secure in the ROM, but insecure for every concrete instantiation of the hash function [Bar01, GK03, CGH04]. For a period of time in the past, having a security proof (only) in the ROM was considered a major drawback.

Nevertheless, experience shows that natural schemes tend to remain secure, and since the ROM typically allows (proving security of) significantly more efficient schemes, it has remained a common methodology. Nowadays, while there is still some controversy, it is fair to say that the ROM is widely accepted: many schemes rely on the ROM, and looking ahead to the post-quantum security discussion and proposals for post-quantum secure schemes, there are very few non-random-oracle alternatives, even if one would be willing to accept worse efficiency.

A very recent work [KRS25] deserves special attention, as it demonstrates an attack against a natural scheme that is provably secure in the ROM. Again, the attack applies regardless of the concrete instantiation of the hash function. Despite ongoing controversy in the community about its impact on our confidence in the random oracle methodology, this attack still contains some aspect that does not appear in typical cryptographic schemes, such as the main focus of this thesis, signature schemes.

Post-quantum cryptography and the NIST competition. Early works of the last century [Deu85, BV93, Sim94] have provided strong evidence that the quantum model of computation is strictly more powerful than its classical counterpart. Therefore, provable security against classical attackers may not hold against quantum attackers. In fact, most of the public-key cryptographic schemes we use today, including the RSA and Diffie-Hellman families of constructions, are completely broken by Shor's (quantum) algorithm [Sho94]. Amidst global efforts to develop quantum computers, the study of cryptographic schemes that remain secure against quantum attackers, also known as *post-quantum cryptography* (PQC), is thus of undeniable importance.

Already in the late 20th century, there have been candidates that seem to resist quantum attackers, which, though, was not regarded a main feature originally. This includes early works of Lamport [Lam79], McEliece [McE78], Ajtai [Ajt96], Matsumoto and Imai [MI88],[5] and Couveignes (re-uploaded version

[5]The construction proposed in [MI88] was soon broken, but it inspired an important branch of PQC based on multivariate polynomials.

available at [Cou06]). With Shor's algorithm being a clear threat, the research greatly intensified, and security against quantum attackers turned into an explicit goal.

The search for practical post-quantum schemes further intensified in 2016, when the US National Institute of Standards and Technology (NIST) initiated a competition for selecting future standards of post-quantum cryptographic schemes, including public-key encryption (PKE)[6] and digital signature schemes. In this competition, experts around the world join forces into proposing future post-quantum standards, and assessing their security. Initially there were 82 candidates submitted to the first round; until now, 5 winners — Dilithium [LDK+22], Falcon [PFH+22], SPHINCS+ [HBD+22], Kyber [SAB+22], and HQC [AAB+22] — have been selected in the 3rd and 4th rounds; in addition, there is an extra round for signature schemes [NIST22], which has now proceeded to the second stage with 14 candidates still on the table.

Quantum impacts on provable security. The presence of quantum attackers impacts provable security in the following three-fold manner.

First of all, if a reduction transforms an attacker $\mathcal{A}$ of the considered scheme to an algorithm solving a computational problem $\mathcal{P}$, then $\mathcal{P}$ itself must remain hard for quantum computers to provide a meaningful security guarantee. Since 1990s, candidates of $\mathcal{P}$ have been extensively investigated, including those based on lattices, isogenies, multi-variate polynomial, coding theory, and symmetric cryptography.

Second, the security definition may require non-trivial modifications in order to capture our intuition, beyond simply extending the quantification to all quantum attackers. A notable example is the *collision resistance*, a property of a hash function H that prevents an attacker to find two distinct inputs x_1 and x_2 such that $H(x_1) = H(x_2)$. Classically, this reflects the intuition that a hash value $H(x)$ uniquely determines its preimage x, at least against efficient attackers. However, the very same intuition fails against quantum attackers, because collision resistance per se does not necessarily prevent a quantum attacker to provide a hash value, and later offer one or another preimage (but not both) on its choice. This gap is addressed in [Unr16], which introduces a stronger security notion for H, the *collapsing* property, that is sufficient to capture our intuition in the quantum case. In fact, as later shown in [DS23], the collapsing property is (essentially) also necessary.

Last but not least, a security reduction must also work in the presence of quantum attackers. While sometimes the reduction carries over directly, this is generally not the case. As a transformation from one algorithm to another, a reduction is by default tailored to the underlying model of computation. Changing the model to allow quantum computation can compromise

[6] Technically speaking, the competition selects key-encapsulation mechanisms (KEMs), which are essentially equivalent to PKE schemes due to the modern KEM-DEM paradigm.

the reduction's validity, meaning it may no longer solve the underlying computational problem successfully, and thus fail to provide meaningful security guarantees.

One of the earliest such examples appeared in [BDF+11], and since then, more prominent cases have been identified, especially in the ROM. Indeed, a typical classical security proof in the ROM exploits that the reduction $\mathcal{B}$ can observe the queries that the attacker $\mathcal{A}$ makes to the random oracle. If $\mathcal{A}$ is quantum though, then these queries can be made "in superposition." Namely, they would be in a quantum state that, by fundamental properties of quantum mechanics, $\mathcal{B}$ cannot observe without causing any disturbance — in which case $\mathcal{A}$ may notice and then simply shut itself down. Therefore, more often than not, a classical security proof in the ROM does not (naively) carry over to the quantum setting.

Another concrete example where classical security proofs fall apart in the quantum realm is when a reduction uses *rewinding*. That is, when $\mathcal{B}$ undoes $\mathcal{A}$'s computation to an earlier point in time, in order to re-run it from there later. Classically, this is realized by copying the internal state of $\mathcal{A}$ at the time to which it is to be rewound. However, if $\mathcal{A}$ is quantum, then its internal state cannot be copied due to the *no-cloning theorem*. This barrier turns out to be intrinsic — [ARU14] shows that, relative to an oracle, there exists a family of cryptographic schemes whose classical security, proven via reductions that use rewinding, does not carry over to the quantum setting. More investigation is thus necessary. We refer interested readers to a line of work that has emerged over more than a decade [Unr12, CMSZ22, LMS22, CAD+24] to salvage the quantum rewinding.

The bottom line is: even with quantum computational hardness in place, there still are many cases where a security proof does not (trivially) carry over to the quantum setting.

1.2 Cryptographic Transformations

A security proof of a cryptographic scheme is typically built up from a sequence of reductions, where one first considers a simpler but weaker scheme, and then strengthens it into a more sophisticated variant with stronger security. Sometimes, certain parts of the sequence can be obtained via applying generic cryptographic transformations that are not tailored to the specific scheme at hand. For instance, one of the earliest such transformations, known as the *Goldreich-Levin construction* [GL89], allows one to enhance a public-key encryption (PKE) scheme into achieving semantic security (IND-CPA). Later works such as the *Fujisaki-Okamoto transformation* [FO99, FO13] is capable of achieving stronger security guarantee (IND-CCA) while being much more efficient.

In the context of signature schemes, most practically relevant constructions

that achieve the standard UF-CMA security follow some variant of either the Fiat-Shamir transformation, or the hash-and-sign design principle. In the context of *key-encapsulation mechanisms* (KEMs) and others, there have recently been increasing attention toward designing more conservative schemes. In that regard, cryptographic combiners may also come in handy. We will describe the above in more detail, for the rest of this section.

Hash-and-sign design principle The hash-and-sign (H&S) design principle is widely used for constructing signature schemes from a trapdoor one-way function f, while the specific requirements of f may differ. To sign a message m, one produces (with a secret trapdoor of f) a preimage $x \in f^{-1}(y)$ of the hash $y := H(m)$ of m, which can then be efficiently verified via checking $f(x) = y$.

Security of H&S has been studied in the ROM, with the earliest classical analysis traced back to the full-domain hash (FDH) signature scheme [BR93], where f is a permutation based on RSA. More than a decade afterward, [GPV08] introduce and analyze a generic framework for constructing H&S signature schemes, in which f, a preimage sampleable function as they call it, only needs to satisfy certain weaker properties. The earliest quantum analysis is treated in [BDF+11], assuming f is collision resistant, which is further relaxed by [Zha12] assuming only the one-way security of f.

To facilitates security proofs, H&S is often made probabilistic, where the message is hashed via $y := H(m, r)$ with a randomized salt r instead, and then r is appended as an additional part of the signature. As summarized in [KX24, Table 1], the quantum analyses of [BDF+11, Zha12] carries over to this probabilistic variant. Moreover, it enjoys tighter security reductions [KX24], as well as more freedom to choose the function f [CD20].

Fiat-Shamir transformation. The *Fiat-Shamir transformation*, proposed by Amos Fiat and Adi Shamir [FS87] in 1986, plays an important role in building signature schemes.[7] Consider a 3-round (public-coin) interactive proof system $\Sigma = (\mathcal{P}, \mathcal{V})$, commonly known as a Sigma protocol. In this protocol, a prover $\mathcal{P}$ tries to convince a verifier $\mathcal{V}$ that it knows some secret witness w of a statement x, i.e. $(x, w) \in R$ for some well-defined relation $R \subseteq \{0,1\}^* \times \{0,1\}^*$. This protocol proceeds as spelled out in Fig. 1.2: First, $\mathcal{P}$ sends over a message r. Then $\mathcal{V}$ poses to $\mathcal{P}$ its challenge $y \leftarrow \mathcal{Y}$ which is sampled uniformly random from a finite set $\mathcal{Y}$. Then $\mathcal{P}$ responds to the challenge with z. Finally, the verifier outputs a bit $b := \mathcal{V}(x, r, y, z)$ indicating whether or not it is convinced.

[7] More generally, the Fiat-Shamir transformation can also be used to build non-interactive zero-knowledge proof (NIZK).

$\Sigma(x, w)$:
1: $(r, \mathsf{st}) \leftarrow \mathcal{P}(x, w)$
2: $y \leftarrow \mathcal{Y}$
3: $z \leftarrow \mathcal{P}(y, \mathsf{st})$
4: **return** $\mathcal{V}(x, r, y, z)$

Figure 1.2: An honest execution of the Sigma protocol $\Sigma = (\mathcal{P}, \mathcal{V})$

Every such interactive proof can be made non-interactive, by replacing the challenge y with the hash of the first message $y := H(r)$ for a hash function H; this can be turned into a signature by hashing the message m as well, i.e. setting $y := H(r, m)$. To construct a proper signature scheme, though, one additionally needs to be able to efficiently sample a "hard instance" (x, w) in R, where computing w given x is hard. Formally, we specify such a signature scheme $\mathsf{FS}[\Sigma, H]$ in Fig. 1.3, assuming that there is indeed an efficiently sampleable distribution $D_{\mathcal{R}}$ generating a hard instance (x, w) in R.

KGen:	$\mathsf{Sign}(\mathsf{sk} = (x, w), m)$:	$\mathsf{Vrfy}(\mathsf{pk} = x, \sigma = (r, z), m)$:
1: $(x, w) \leftarrow \mathcal{D}_R$	1: $(r, \mathsf{st}) \leftarrow \mathcal{P}(x, w)$	1: $y := H(r, m)$
2: $\mathsf{sk} := (x, w)$	2: $y := H(r, m)$	2: **return** $\mathcal{V}(x, r, y, z)$
3: $\mathsf{pk} := x$	3: $z \leftarrow \mathcal{P}(y, \mathsf{st})$	
4: **return** $(\mathsf{sk}, \mathsf{pk})$	4: **return** $\sigma := (r, z)$	

Figure 1.3: The Fiat-Shamir signature scheme $\mathsf{FS}[\Sigma, H] = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vrfy})$ constructed from a Sigma protocol $\Sigma = (\mathcal{P}, \mathcal{V})$, where H is a hash function and $\mathcal{D}_R$ efficiently samples a hard instance in R.

Security properties of the signature scheme $\mathsf{FS}[\Sigma, H]$ are typically proven in the ROM. The earliest formal analysis goes back to [PS96]. However, establishing quantum security turns out significantly more challenging. The community was facing an (improper) impossibility result [DFG13], as well as quantum attacks [ARU14] breaking certain artificial choice of Σ, which carries over to the Fiat-Shamir scheme $\mathsf{FS}[\Sigma, H]$. Facing these negative results, Dominique Unruh proposed and analyzed a variant known as the *Unruh transformation* [Unr15], and studied the plain Fiat-Shamir transformation when given statistical soundness of Σ in [Unr17]. It took several decades from the first classical analysis of the Fiat-Shamir transformation, before the generic quantum security is formally proven [DFMS19, LZ19] under a slightly stronger (computational) assumption of Σ, known as the collapsing property,[8] than what is necessary for achieving classical security.

[8] As its name suggests, the definition of Σ's collapsing property is inspired by that of a hash function.

In retrospect, it has become clear how the security properties of $\mathsf{FS}[\Sigma, H]$ in the ROM follow from those of Σ. If Σ is *knowledge sound*, which prevents $\mathcal{P}$ to convince $\mathcal{V}$ without knowing the witness w, then $\mathsf{FS}[\Sigma, H]$ is $\mathsf{UF\text{-}NMA}$ against classical attackers. In addition, whenever Σ satisfy the collapsing property, the above holds against quantum attackers as well. If Σ (1) satisfies the *honest-verifier zero-knowledge* property (HVZK), which prevents $\mathcal{V}$ to learn anything about w in an honest execution, and (2) contains high min-entropy in its first message,[9] then there is an $\mathsf{UF\text{-}CMA\text{-}to\text{-}UF\text{-}NMA}$ security reduction, i.e. $\mathsf{FS}[\Sigma, H]$ is $\mathsf{UF\text{-}CMA}$ as long as it is $\mathsf{UF\text{-}NMA}$, covering both classical and quantum attackers. Indeed, the security proof of $\mathsf{FS}[\Sigma, H]$ applies regardless of the concrete instantiation of Σ, provided the aforementioned premises are satisfied.

Fiat-Shamir with aborts. The *Fiat-Shamir with aborts* (FSwA) transformation is adapted from the above Fiat-Shamir transformation by Vadim Lyubashevsky [Lyu09] in 2009. It is especially useful for constructing lattice-based and isogeny-based signature schemes, many of which are relevant to the NIST PQC competition.

The main difference between FSwA and the original Fiat-Shamir transformation stems from the fact that, for many Sigma protocols $\Sigma = (\mathcal{P}, \mathcal{V})$ relevant to the post-quantum setting, the prover $\mathcal{P}$ aborts with a certain probability. Typically, the abort by $\mathcal{P}$ is introduced, because otherwise completing the protocol would leak the witness w to the verifier $\mathcal{V}$. For $\mathcal{P}(x, w)$ to convince $\mathcal{V}(x)$ that (x, w) is a valid instance in R, the entire protocol must then be repeated until $\mathcal{P}$ does not abort. This is sometimes referred to as *rejection sampling*, and it carries over when constructing an FSwA signature scheme $\mathcal{S}$, which simply replaces $\mathcal{V}$'s challenge in each repetition with a suitable hash value.

Similar to the case of the plain Fiat-Shamir transformation, analyses of FSwA are typically performed in the ROM. The first classical analysis of FSwA was treated in [Lyu09], and since then FSwA has been used in many concrete signature schemes. This includes Lyubashevsky's signature scheme [Lyu09, Lyu12], GLP [GLP12], TESLA [ABB+20], Dilithium [DKL+18], SeaSign [DG19], and HAETAE [CCD+24]. Among these, TESLA was the earliest analyzed in the quantum setting, though the proof was tailored to the specifics of the scheme. Toward achieving full-fledged quantum security of FSwA, the authors of [KLS18] provided a generic $\mathsf{UF\text{-}CMA\text{-}to\text{-}UF\text{-}NMA}$ security reduction, which, however, is shown to be flawed in our later works (see Section 1.3). They also showed that FSwA satisfies $\mathsf{UF\text{-}NMA}$ under a more stringent condition on Σ than one would ideally require. This condition was later relaxed by [DFMS19] through a generic $\mathsf{UF\text{-}NMA}$ security proof that holds for Fiat-Shamir transformations, both with and without aborts.

[9] The second requirement is mild, and easily enforceable.

Hash-and-sign with retry/aborts. Just as the original Fiat-Shamir transformation can be generalized to FSwA, the H&S design principle can likewise be extended. In particular, many post-quantum signature schemes, especially those based on coding theory and multivariate polynomials, relies on a variant known as the *hash-and-sign with retry/aborts* (HSwA) design principle. As its name suggests, in a HSwA signature scheme, the main body of the signing procedure may abort with a certain probability, and thus have to be repeated several times until it does not abort. This new design principle is adopted by various constructions, many of which are relevant to the NIST PQC competition. Concrete examples include the Hidden Field Equation (HFE) scheme [Pat96], the Unbalanced Oil and Vinegar (UOV) scheme [KPG99], the Courtois-Finiasz-Sendrier (CFS) scheme [CFS01, Dal08], GeMSS [CFMR+17], Wave [DST19], MAYO [Beu21], QR-UOV [FIKT21], and Falcon+ [GJK24] (an updated version of Falcon).

BUFF transformations. The next transformation we want to briefly introduce here is the BUFF transformation.[10] It is relatively niche compared to, e.g. the Fiat-Shamir transformation, in that its purpose is to provide certain non-standard security properties that are not relevant in typical applications of digital signature schemes. However, it has recently gained quite some attention since the call for additional post-quantum signature candidate schemes by NIST [NIST22] explicitly refers to these non-standard security properties as being "desirable," and many of the recent signature candidates refer to it. The BUFF transformation additionally gained attention due to our work which shows that the original understanding of what the BUFF transform achieves, has to be revised (see Section 1.3 for more details).

In more detail, in [CDF+21], Cremers, Düzlü, Fiedler, Fischlin, and Janson proposed the first formal definition of the aforementioned additional desirable security properties, including *exclusive ownership* [PS05], *message-bound signatures*, and *non-resignability* [JCCS19]. They act as a second layer of protection against atypical misuses of a signature scheme under a higher-level protocol — indeed, there have been real-life attacks that exploit the lack of these properties, as discussed in [CDF+21]. The authors of [CDF+21] then proposed the *BUFF transformation* as a simple and efficient generic compiler that transforms any signature scheme, into another one that (is claimed to) achieve all these additional security properties. As will be elaborated in Fig. 4.3, it simply performs an additional pre-hash to the message with the public key, signs the hash, and appends the hash as an additional part of the signature, with verification done in the obvious way.

Candidate signature schemes in the extra round, such as Squirrels [ENST23], Racoon [dEK+23], HAWK [BBD+24], PROV [GCF+23], Vox [PCF+23], and eMLE [LZ23] have referred to BUFF in their proposal — some have incorpo-

[10] The acronym BUFF stands for Beyond-UnForgeability-Feature.

rated BUFF as a part of their design, while some others mentioned the possibility of applying BUFF to their schemes. BUFF is also relevant to two of the winners in round 3. As argued in [CDF+21], Dilithium has implicitly applied BUFF, inheriting generic security guarantees of BUFF, while Falcon does not. Nevertheless, the Falcon team has announced that they will also apply BUFF [FHK+22] to achieve the additional security properties.

Cryptographic combiners. A cryptographic combiner transforms multiple cryptographic schemes into a single (hybrid) scheme with the same or similar functionality, such that the resulting scheme remains secure as long as at least one of the component schemes is secure.

Combiners are particularly relevant in the context of post-quantum cryptography, where the security of post-quantum schemes often relies on relatively new computational assumptions, compared to well-established pre-quantum schemes based on RSA, Diffie-Hellman, and elliptic-curve cryptography. Combining a post-quantum scheme with a pre-quantum one thus provides a conservative transition to gain quantum security while retaining the classical one. This is not merely of theoretical interest — indeed, combiners can be used in standardized protocols such as Internet Key Exchange Protocol version 2 (IKEv2) [TTB+23] and Transport Layer Security 1.3 (TLS 1.3) [SFG25].[11] In the Netherlands, a nationwide initiative involving industry and academia, *Hybrid Approach for quantum-safe Public Key Infrastructure Development for Organisations* (HAPKIDO), has also been studying both practical and theoretical aspects of hybrid PKI as well as combiners. We note that, as observed in a recent literature survey [FHA23], different parties may have different opinions toward this kind of hybrid approach. Some government agencies in Europe publicly endorsed the use of combiners, such as the French Cybersecurity Agency (ANSSI) [ANSS22] and the German Federal Office for Information Security (BSI) [EHH+22], while some others hold a rather negative opinion about it, like the American National Security Agency (NSA) [NSA24].

Needless to say, the precise security notion of a combiner depends on that of the primitive being combined. In some cases, there is a straightforward construction: for instance, concatenating signatures produced by the component schemes trivially yields a combiner that preserves the standard $\mathsf{UF\text{-}CMA}$ security. In some other cases, however, a more sophisticated approach is required.

Relevant to this thesis, notable examples are combiners for *key-encapsulation mechanisms* (KEMs). In a nutshell, a KEM is a public-key primitive where, anyone with a long-term public key pk can (produce and) *encapsulate* a session key k in a ciphertext c that can later be *decapsulated* to the same k by the holder of the long-term secret key sk corresponding to pk. Securely combining KEMs is far from obvious. Indeed, the standard security of a KEM

[11] The use of combiners is not standardized in TLS 1.3, but it is supported by the IETF informational document [SFG25].

is IND-CCA, which prevents a valid ciphertext being mauled into another one that is decapsulated to the same session key. This property is in particular not preserved by the naive KEM combiner that concatenates all ciphertexts and takes the xor of the session keys, as the combined ciphertext and the combined session key respectively. Moreover, [GHP18] shows that for the similar reason, another natural construction called the xor-then-PRF combiner does not preserve IND-CCA security either.

On the positive side, constructing KEM combiners has been a topic under active research, with various concrete constructions shown to be secure. This includes combiners that are based on PRF-then-xor [GHP18], split-key PRF (skPRF) [GHP18], xor-then-MAC (XtM) [BBF+19], dual PRF [BBF+19], and the FO transformation [HV21]. Looking ahead at our contributions, our work in [DFH22], on which this thesis is based, helps establish (quantum) security of a particularly efficient construction based on skPRF. For the sake of efficiency, recent works also look into concrete constructions of hybrid KEMs [BCD+24], and KEM combiners that are not full-fledged since they rely on additional properties of the component schemes [ABK25].

1.3 Our Contributions

In this thesis, we (re)consider and study the following cryptographic transformations: the hash-and-sign and Fiat-Shamir (with Aborts), which will be covered in Chapter 3; the BUFF transformation, which will be covered in Chapter 4; and a particularly efficient KEM combiner, which will be covered in Chapter 5. For the first three cases (HSwA, FSwA, and BUFF), our results significantly contribute to the security understanding of the transformations, both when considering classical and quantum attacks. In particular, we show that there were incorrect understandings of them prior to our results, and we re-establish security (to the extent possible). For the KEM combiner, on the other hand, we provide the first quantum security proof.

Security of FSwA and HSwA. As have been discussed in the prior section, FSwA was first introduced in [Lyu09] as a variant of the Fiat-Shamir transformation, with quantum security analyzed in [KLS18, DFMS19], and is often used for constructing post-quantum signature schemes. Unexpectedly, though, there turns out to be a crucial but subtle flaw in [KLS18] and all prior UF-CMA-to-UF-NMA security reductions of FSwA signature schemes. The flaw applies both classically and quantum, and was discovered during the course of a formal verification project in [BBD+23]. It was also independently and concurrently discovered by [DFPS23], and reappeared in the context of HSwA signature schemes [KX24]. As a consequence, security proofs of a long list of (potentially hundreds of) works based on FSwA or HSwA are invalidated, and there is no known easy fix. Specifically, the upcoming NIST PQC standard,

Dilithium [DKL+18], is left with no valid security proof. We will describe this flaw in Section 3.3.

The main technical contribution of Chapter 3 is to restore confidence toward FSwA and HSwA signature schemes from the aforementioned flaw. Below, we briefly explain how we achieve this.

To begin with, as a conceptual contribution, we notice the structural resemblance of FSwA and HSwA, and put together (in Section 3.2.1) a unified framework capturing both, which we refer to as *generalized Fiat-Shamir with aborts* signature schemes. In Section 3.4, we then provide a new, fixed, and generic UF-CMA-to-UF-NMA security proof in the ROM, which applies both classically and quantum, for all such (generalized) FSwA signature schemes $\mathcal{S}$. Concretely, we show that as long as $\mathcal{S}$ satisfies what we call the *accepting honest-verifier zero-knowledge* (acHVZK) property, together with a mild requirement about min-entropy, and if it is UF-NMA, then it is also UF-CMA. The acHVZK property that we premise is essentially the minimal: it ensures that an "honestly generated signature" leaks nothing about the secret key. In particular, it is weaker than the naHVZK property premised in the prior flawed security proof by [KLS18].

More formally, in Section 3.4, based on our work in [BBD+23], we show that every UF-CMA attacker $\mathcal{A}$ making at most q_H quantum queries to the random oracle H, and at most q_S classical queries to the signing oracle, can be transformed into a similarly efficient UF-NMA attacker $\mathcal{B}$ such that

$$\mathbf{Adv}^{\mathsf{UF\text{-}CMA}}_{\mathcal{S}}(\mathcal{A}) \leq \mathbf{Adv}^{\mathsf{UF\text{-}NMA}}_{\mathcal{S}}(\mathcal{B}) + O\left(\frac{q_H\sqrt{q_S}}{1-p} + \frac{q_S\sqrt{q_H}}{(1-p)^2}\right)\sqrt{\epsilon} + q_S\zeta \ ,$$

for parameters p, ϵ, ζ dependent on the scheme,[12] where $\epsilon, \zeta > 0$ are negligibly small, and $0 \leq p < 1$, which we call the "abort rate," is not too close to 1. In the case where queries to H are restricted to be classical, we have a correspondingly tighter bound

$$\mathbf{Adv}^{\mathsf{UF\text{-}CMA}}_{\mathcal{S}}(\mathcal{A}) \leq \mathbf{Adv}^{\mathsf{UF\text{-}NMA}}_{\mathcal{S}}(\mathcal{B}) + O\left(\frac{q_S q_H}{1-p} + \frac{q_S^2}{(1-p)^2}\right)\epsilon + q_S\zeta \ .$$

In Section 3.5, based on our follow-up work in [FFH25], we further improve the generic quantum bound to

$$\mathbf{Adv}^{\mathsf{UF\text{-}CMA}}_{\mathcal{S}}(\mathcal{A}) \leq \mathbf{Adv}^{\mathsf{UF\text{-}NMA}}_{\mathcal{S}}(\mathcal{B}) + O\left(\frac{q_S\sqrt{q_H}}{1-p}\right)\sqrt{\epsilon} + q_S\zeta \ .$$

Note that, most of the time q_H is much larger than q_S, with $q_S\zeta$ being non-dominating. In this case, the quantum security loss is brought down from $O\left(\frac{q_H\sqrt{q_S}}{1-p}\right)\sqrt{\epsilon}$ to $O\left(\frac{q_S\sqrt{q_H}}{1-p}\right)\sqrt{\epsilon}$. Concretely, this amounts to a factor of

[12] In general, these parameters may be key-dependent, but we ignore it here for simplicity.

roughly $\sqrt{q_H/q_S} \approx 2^{32}$, for the NIST security level 2 where q_S and q_H are up to 2^{64} and 2^{128} respectively. This, as well as the classical bound, is (to some extent) as tight as one can hope for. Specifically, in all relevant regimes,[13] the obtained additive security loss, up to a constant factor, matches that of the standard, well-studied Fiat-Shamir transformation without aborts (see [GHHM21, Theorem 3]).[14] Therefore, any further improvements, should they exist, also need to carry over to this well-understood setting without aborts.

Naively applying our generic bound to Dilithium, however, does not provide sufficient concrete security. This is mainly due to our generic bounds still being worse compared to those of the flawed analyses in [KLS18]. The bounds in [KLS18] are independent of q_S and q_H, while our losses scale along with q_S and q_H. Therefore, we provide a more elaborated concrete analysis of Dilithium in Section 3.6, which is partially computer-aided, and recover its full security level.

(In)security of BUFF. Recall that, the BUFF transformation is introduced in [CDF+21], and is claimed to achieve several additional security properties that they formally defined: exclusive ownership, message-bound signatures, and non-resignability (NR). However, in Chapter 4, which is based on our works in [DFHS24, DFH+24, FHK25], we show that the actual situation for the case of NR is much more subtle.

In Section 4.3, we show that the NR property, as defined in [CDF+21], is essentially impossible to achieve. More precisely, we show that, when considering a signature scheme that has sufficient computational min-entropy (HILL entropy) within a randomly chosen message given its signature, there is an explicit attack breaking NR, both in the plain model and the ROM. In particular, for every signature scheme obtained via applying the BUFF transformation, there is a concrete attack breaking its NR security, provided that the underlying hash function is sufficiently compressing, as is typically the case. At the opposite extreme, if a message can be efficiently recovered from its signature, then as noted by [CDF+21], the scheme is trivially not NR. While there is still a small, artificial gap that is not covered, all natural schemes fall into either of the above two categories.

Our generic attack against NR contradicts and invalidates the claimed security of BUFF in [CDF+21], for which we then identify the crucial flawed step in the security proof. We observe that the flawed analysis reduces the NR security of the BUFF transformation, to a particular security of the underlying hash function called Φ-*non-malleability* (Φ-NM), where the prefix Φ specifies a suitable class of functions of relevance to the security definition. Then, to complete the full security justification of BUFF, a claim in [BFS11] that the

[13] If the abort rate p is very close to 1, which practically never happens, then our security bounds degrade slightly.

[14] In the (atypical) regime where q_S is much larger than q_H, our quantum bound is even tighter.

random oracle satisfy Φ-NM is recycled. Unfortunately, this particular claim is false. As shown in Section 4.3, for any hash function that is sufficiently compressing (which is typically so), there is an explicit attack breaking its Φ-non-malleability.

We observe, however, that there are certain aspects of our attack that do not appear in real-world scenarios. Therefore, rather than being viewed as a practical threat, our attack shows that the definition of NR security as in [CDF+21] is not the "right" one. To recover from this negative state of affairs, it is necessary to explore alternative definitions of NR, and to investigate whether they can be achieved. That is exactly what we do for the rest of Chapter 4.

In Section 4.4, we introduce a slightly weaker yet still meaningful definition of non-resignability in the ROM, which we call $\mathsf{NR}^{H,\perp}$. With $\mathsf{NR}^{H,\perp}$ in place, the generic attack no longer applies. Whether or not BUFF satisfies $\mathsf{NR}^{H,\perp}$ is far from obvious. Therefore, to begin with, we introduce a salted variant of BUFF, \$-BUFF, and show that it satisfies $\mathsf{NR}^{H,\perp}$, covering both classical and quantum attackers, if the entropy requirement in the definition of $\mathsf{NR}^{H,\perp}$ is statistical. On the other hand, if the entropy requirement is computational, then there is a counter example, for which applying \$-BUFF (or BUFF) would result in a scheme that does not satisfy $\mathsf{NR}^{H,\perp}$. Again, the attack here still contains a certain level of contrivedness, which does not seem very realistic either.

In Section 4.5, we introduce yet another variant of NR in the ROM, which we call $\mathsf{sNR}^{H,\perp}$. One motivation doing so is to mitigate the negative result for $\mathsf{NR}^{H,\perp}$ under the computational entropy requirement, which no longer applies to $\mathsf{sNR}^{H,\perp}$. Moreover, it also serves as a proxy for analyzing the $\mathsf{NR}^{H,\perp}$ property of the original, unsalted BUFF, since under the statistical entropy requirement, any scheme that satisfies $\mathsf{sNR}^{H,\perp}$ also satisfies $\mathsf{NR}^{H,\perp}$.[15] As the main technical contribution of this section, we then show that the (unsalted) BUFF satisfies $\mathsf{sNR}^{H,\perp}$, which implies that it also satisfies $\mathsf{NR}^{H,\perp}$. This confirms our intuition that BUFF indeed satisfies a meaningful notion of NR, when the underlying hash function is idealized as a random oracle.

In Section 4.6, we further investigate the security of BUFF, when using a typical iterative hash function, e.g. SHA-2, SHA-3, or SHAKE. Specifically, we consider a more fine-grained setting, where, instead of idealizing the entire hash function as a random oracle, only the round function is idealized. To our surprise, in this fine-grained setting, the BUFF transformation is no longer secure! We show that, if the underlying hash function of BUFF is an iterative hash function, then there is an explicit attack breaking the (fine-grained version of) $\mathsf{sNR}^{H,\perp}$ for any scheme obtained from applying the BUFF transformation.

To recover from the above (fine-grained) negative result, we introduce the third variant of BUFF in Section 4.7, which we call the Sandwich BUFF trans-

[15] On the other hand, under the computational entropy requirement, $\mathsf{sNR}^{H,\perp}$ and $\mathsf{NR}^{H,\perp}$ become incomparable.

formation (sBUFF). Recall that the original BUFF works via hashing the message with the public key, signing the hash, and then appending the hash to the signature. As its name suggests, the Sandwich BUFF is almost identical, except that in the hashing step, the public key is sandwiched between two copies of the message, rather than concatenated after it. We then show that, if the Sandwich BUFF transformation $\mathsf{sBUFF}[\mathcal{S}, \mathsf{MD}]$ uses the Merkle-Damgård hash function MD^H, where the compression function is modelled as a random oracle H, then it satisfies $\mathsf{sNR}^{H,\perp}$.[16]

Finally, we conclude Chapter 4 with Section 4.8, where we show that all positive results in terms of $\mathsf{sNR}^{H,\perp}$ (and its fine-grained version) carries over, when the underlying entropy requirements are computational.

Quantum security of a KEM combiner In [GHP18], Giacon, Heuer and Poettering showed that any *split-key PRF* (skPRF) gives rise to a secure KEM combiner. In more detail, they show that if an skPRF is used in the (rather) obvious way as a key-derivation function in a KEM combiner, then the resulting hybrid KEM is IND-CCA secure if at least one of the component KEMs is IND-CCA secure. They also suggest a few candidates for skPRFs. The most efficient of the proposed constructions is a hash-based skPRF, which is proven secure in [GHP18] in the random-oracle model, considering classical attackers. However, in the context of a quantum attack, which is in particular relevant in the above example application of a combiner, it is crucial to prove security when quantum attackers are in place, i.e. when attacker can query the random oracle in quantum superposition. In Chapter 5, which is based on our work in [DFH22], we close this gap by proving post-quantum security of the hash-based skPRF construction as mentioned above.

[16] For simplicity, we do not cover the case of sBUFF using a Sponge function, but it is treated in our work [FHK25].

Chapter 2

Preliminaries

Throughout this thesis, we assume readers' familiarity with general mathematics and a basic understanding of algorithms, computational complexity, and quantum information science. In this chapter, though, we set out our notation and terminology, and discuss several aspects that are particularly relevant.

2.1 Notations

Basic mathematics. Throughout the thesis, we take the convention that $\ln(x)$ and $\log(x)$ are the natural and binary logarithm respectively, and for every positive integer n we denote by $[n] := \{1, \ldots, n\}$.

The security parameter. In cryptography, it is a standard convention that all algorithms and adversaries (and likewise, functions) take as input the unary representation 1^λ of a security parameter λ. Throughout this thesis, we adopt this convention and keep λ implicit unless necessary. Whenever the considered algorithms or functions are associated with a mathematical object, such as their input/output space or an oracle they query, it is to be understood that the object is implicitly parameterized by λ as well.

The arrow notation. For a finite set $\mathcal{X}$, we will denote a random variable x being sampled uniformly from the set by $x \leftarrow \mathcal{X}$. For a distribution $\mathcal{D}$ that is not necessarily uniform, though, we will write $x \leftarrow \mathcal{D}$ to indicate that x is drawn from $\mathcal{D}$. Similarly, since an algorithm may in general have a randomized output, we denote by $x \leftarrow \mathcal{A}$ that x is an output generated by running $\mathcal{A}$. Sometimes we are interested in the probability that $x = x^\circ$ for a certain fixed value x°, in which case we often use the following short-hand notation

$$\Pr\left[\, x^\circ \leftarrow \mathcal{A} \,\right] := \Pr_{x \leftarrow \mathcal{A}}\left[\, x = x^\circ \,\right] .$$

Signature schemes. Following the standard definition, a *signature scheme* $\mathcal{S}$ consists of a key generation algorithms KGen, a signing algorithm Sign, and a verification algorithm Vrfy, all of which are PPT algorithms. By default, we denote by $\mathcal{PK}$ the space of public keys, by $\mathcal{SK}$ the space of secret keys, by $\mathcal{M}$ the message space and by $\mathcal{SGN}$ the space of signatures. For simplicity, we assume throughout that $\mathcal{S}$ is perfectly correct, i.e. $\mathsf{Vrfy}\big(\mathsf{pk}, m, \mathsf{Sign}(\mathsf{sk}, m)\big) = 1$ with certainty, and that the public key can be efficiently derived from the secret key.

Statistical distance. For any two random variables x_1, x_2 over a finite set $\mathcal{X}$, we denote their *statistical distance* by

$$SD(x_1, x_2) := \frac{1}{2} \sum_{x^\circ \in \mathcal{X}} \big| \Pr[\, x_1 = x^\circ \,] - \Pr[\, x_2 = x^\circ \,] \big| \,,$$

which is known to equal $\max_V(\Pr[\, V(x_1) = 1 \,] - \Pr[\, V(x_2) = 1 \,])$ with the maximization over all predicates $V : \mathcal{X} \to \{0, 1\}$.

Guessing probability and min-entropy. For a random variable y over a finite set $\mathcal{Y}$, the *guessing probability* and the *min-entropy* are respectively defined as

$$\mathsf{guess}(y) := \max_{y^\circ} \Pr\big[\, y = y^\circ \big] \qquad \text{and} \qquad \mathsf{H}_\infty(y) := -\log\big(\mathsf{guess}(y)\big) \,.$$

For random variables x and y over respective finite sets $\mathcal{X}$ and $\mathcal{Y}$, the *conditional* guessing probability is defined as

$$\mathsf{guess}(y \mid x) := \sum_{x^\circ} \Pr[x = x^\circ] \cdot \max_{y^\circ} \Pr\big[\, y = y^\circ \;\big|\; x = x^\circ \big] \,.$$

It is well known that $\mathsf{guess}(y \mid x) = \max_f \Pr[y = f(x)]$, where the maximization is over all (deterministic or randomized) functions $f : \mathcal{X} \to \mathcal{Y}$. In line with the unconditional case above, the *conditional* min-entropy is then given by $\mathsf{H}_\infty(y \mid x) := -\log\big(\mathsf{guess}(y \mid x)\big)$.

2.2 Oracle Algorithms

Throughout this thesis, we frequently consider the random oracle model (ROM), where parties/algorithms have oracle access to a uniformly random function $H : \mathcal{X} \to \mathcal{Y}$, with suitably chosen finite domain $\mathcal{X}$ and range $\mathcal{Y}$. We also consider quantum algorithms, which can then query H "in superposition," i.e. in the form of the unitary $|x, y\rangle \mapsto |x, y + H(x)\rangle$. We write $\mathcal{A}^H$ to denote a (classical or quantum) algorithm with query access to H. Similarly, we write $\mathcal{A}^{H,\bullet}$

in case $\mathcal{A}$ may additionally make queries to another, possibly unspecified oracle, hinted at with a placeholder "$\bullet$" (or $\blacklozenge$, $\blacksquare$, $\blacktriangle$ etc.). Furthermore, in case of such an oracle algorithm $\mathcal{A}^{H,\bullet}$, we write $\mathcal{A}^{H,\mathcal{O}}$ to express that the unspecified oracle is instantiated with the particular algorithm $\mathcal{O}$. We note that such an instantiation may also have access to H, i.e., the oracle may be instantiated with an algorithm $\mathcal{O}^H$, in which case we then naturally write $\mathcal{A}^{H,\mathcal{O}^H}$.

We stress that we use the same notation for classical and quantum algorithms with respective classical or quantum access to the random oracle H, while the query access to any other oracle is always assumed to be classical in this work, and thus so is any oracle instantiation $\mathcal{O}$ we consider. Furthermore, by default we assume any oracle instantiation $\mathcal{O}$ to be at most *statically stateful*, meaning that the state (if any) is chosen at the beginning and then remains fixed.[1]

Regarding language and notation, since in the random oracle model by default all algorithms have access to H, we reserve the terminology *oracle* algorithm for those that have access to one (or more) additional oracle(s). Also, once we have specified that an oracle algorithm is of the form, say, $\mathcal{A}^{H,\bullet}$, i.e., with access to H and to another, unspecified oracle, we may then simply write $\mathcal{A}$ in later occurrences, taking the considered form as understood.

We note that even though an oracle algorithm $\mathcal{A}^{H,\bullet}$ typically expects a particular instantiation $\mathcal{O}$ of the oracle, we may consider a run of $\mathcal{A}^{H,\mathcal{O}'}$ for any instantiation $\mathcal{O}'$. We may also consider a run of $\mathcal{A}$ where different calls to the oracle are answered by different instantiations. We then write

$$\mathcal{A}^{H,[\mathcal{O}_1^{i_1},\mathcal{O}_2^{i_2},\ldots]}$$

to capture that the first i_1 oracle queries are answered by $\mathcal{O}_1$, the following i_2 queries by $\mathcal{O}_2$, etc. We note that in-between these oracle queries, there may be multiple queries to H.

For proof-technical reasons, we will also consider the case where an (oracle) algorithm may make *write* (a.k.a. *reprogramming*) *queries* to the random oracle H. On input (x, y), such a write query will redefine the function value of $H(x)$ to y; we will capture this by the command $H(x) := y$. We only consider *classical* write queries, and we will write $\mathcal{A}^{\bar{H}}, \mathcal{A}^{\bar{H},\bullet}, \mathcal{A}^{\bar{H},\mathcal{O}^{\bar{H}}}$ etc. to indicate that $\mathcal{A}$ (and $\mathcal{O}$) may also make classical write queries to H, next to the ordinary (classical or quantum) read queries.

For an oracle algorithm $\mathcal{A}^{\bar{H},\bullet}$ and a specific instantiation of the oracle, say, as $\mathcal{O}^{\bar{H}}$, we write

$$\mathcal{B}^{\bar{H}} := \mathcal{A}^{\bar{H},\mathcal{O}^{\bar{H}}}$$

[1] For comparison, the random oracle H with read queries only is statically stateful when instantiated/implemented (inefficiently) by choosing a random function at the beginning, while it is adaptively stateful when done using lazy sampling. The random oracle $\bar{H}$ with write queries (see below) is adaptively stateful no matter how it is implemented.

to specify that the algorithm $\mathcal{B}$ runs $\mathcal{A}$ and answers all oracle queries by means of running $\mathcal{O}^{\bar{H}}$ internally, while forwarding all random oracle queries of $\mathcal{A}$ and $\mathcal{O}$ to H. This indeed makes $\mathcal{B}$ an algorithm of the form $\mathcal{B}^{\bar{H}}$.

Pushing this a bit further, $\mathcal{B}^{\bar{H},\bullet} := \mathcal{A}^{\bar{H},[\mathcal{O}^{i-1},\bullet,\mathcal{P}^{\infty}]}$ then denotes the oracle algorithm that runs $\mathcal{A}$, answers the first $i-1$ oracle queries with $\mathcal{O}$, forwards the ith query to $\mathcal{B}$'s oracle, and answers all the remaining ones with $\mathcal{P}$. In case $\mathcal{A}$ makes at most q oracle queries, we could just as well write $\mathcal{P}^{q-i}$ instead of $\mathcal{P}^{\infty}$.

2.2.1 Equivalences of Oracle Algorithms

Different (oracle) algorithms may "behave the same way". We want to formally capture the possible meanings of the latter that will be important for us.

We say that two algorithms $\mathcal{B}^{\bar{H}}$ and $\mathcal{C}^{\bar{H}}$ are *semantically equal*, written as equality $\mathcal{B}^{\bar{H}} = \mathcal{C}^{\bar{H}}$, if the *joint distribution* of: (1) the output of the considered algorithm, and (2) the (possibly reprogrammed) random oracle H at the end of the execution of the algorithm, are the same, *for any initial choice* of H and any joint input. This is for instance satisfied when $\mathcal{C}$ is a purely syntactic rewriting of the defining code of $\mathcal{B}$.

Another natural, but weaker, equivalence relation, which we call *output equivalent*, is $\mathcal{B}^{\bar{H}} \simeq \mathcal{C}^{\bar{H}}$, which (by our definition) expresses that the two distributions of the respective *outputs only* are equal, *on average* over the random choice of H and for any joint input.

We stress that the assignment $\mathcal{B}^{\bar{H}} := \mathcal{A}^{\bar{H},\mathcal{O}^{\bar{H}}}$ implies semantical equivalence $\mathcal{B}^{\bar{H}} = \mathcal{A}^{\bar{H},\mathcal{O}^{\bar{H}}}$. Furthermore,

$$\mathcal{O}^{\bar{H}} = \mathcal{P}^{\bar{H}} \implies \mathcal{A}^{\bar{H},\mathcal{O}^{\bar{H}}} = \mathcal{A}^{\bar{H},\mathcal{P}^{\bar{H}}}$$

for any $\mathcal{A}^{\bar{H},\bullet}$, while $\mathcal{O}^{\bar{H}} \simeq \mathcal{P}^{\bar{H}}$ is in general not sufficient to conclude $\mathcal{A}^{\bar{H},\mathcal{O}^{\bar{H}}} \simeq \mathcal{A}^{\bar{H},\mathcal{P}^{\bar{H}}}$.[2]

Output equivalence sometimes only holds approximately, informally denoted as $\mathcal{B}^{\bar{H}} \approx \mathcal{C}^{\bar{H}}$ then. This is commonly quantified via the *distinguishing advantage*

$$\left|\Pr\left[1 \leftarrow \mathcal{B}^{\bar{H}}\right] - \Pr\left[1 \leftarrow \mathcal{C}^{\bar{H}}\right]\right|,$$

where $\mathcal{B}$ and $\mathcal{C}$ are then typically assumed to have a binary output, and where the probability is over the randomness of the algorithms ($\mathcal{B}$ and $\mathcal{C}$) and the random choice of the random oracle H. We note that for simplicity, we consider here algorithms with no input, but the same quantities can also be considered for any choice of input, of course. In case of binary outputs, the above *distinguishing advantage* coincides with the statistical distance $SD\left(\mathcal{B}^{\bar{H}}, \mathcal{C}^{\bar{H}}\right)$ between the two output distributions, and so we then use the two expressions

[2] Even if $\mathcal{O}$ and $\mathcal{P}$ are restricted to read access only, still $\mathcal{O}^{H} \simeq \mathcal{P}^{H} \not\Rightarrow \mathcal{A}^{\bar{H},\mathcal{O}^{H}} \simeq \mathcal{A}^{\bar{H},\mathcal{P}^{H}}$.

interchangeably.[3] In case of non-binary outputs, the statistical distance upper bounds the distinguishing advantage.

2.2.2 Conditioning of Oracle Algorithms

The run of an algorithm $\mathcal{A}^{\bar{H}}$ defines the (distribution of the) output of the algorithm, but also various internal variables, like the local randomness of $\mathcal{A}$ in case of a classical algorithm or the measurement outcomes of measurements performed by $\mathcal{A}$ in case of a quantum algorithm, the (classical) write queries to H, etc. For any event Λ defined by these random variables, we can then consider a run of $\mathcal{A}^{\bar{H}}$ conditioned on Λ, which we will denote by $\mathcal{A}^{\bar{H}}[\Lambda]$. We may use the variation $\mathcal{A}[\Lambda]^{\bar{H}}$ on the notation to indicate that Λ does not depend on H. We mainly use the latter in case of a classical algorithm $\mathcal{A}$, where it then means that Λ is determined by $\mathcal{A}$'s local randomness (and its input). In this case, the assignment $\mathcal{B}^{\bar{H}} := \mathcal{A}[\Lambda]^{\bar{H}}$ is well-defined as the oracle algorithm that runs $\mathcal{A}$ but samples the local randomness conditioned on Λ.

This notation extends naturally to $\mathcal{A}^{\bar{H},\mathcal{O}^{\bar{H}}}[\Lambda]$ for any oracle algorithm $\mathcal{A}^{\bar{H},\bullet}$ and instantiation $\mathcal{O}^{\bar{H}}$, as well as to the variations $\mathcal{A}[\Lambda]^{\bar{H},\bullet}$ and $\mathcal{A}^{\bar{H}}[\Lambda]^{\bullet}$, when Λ does not depend on (H and) the oracle.

The above notation will mainly be useful in the following context. Consider two algorithms $\mathcal{B}^{\bar{H}}$ and $\mathcal{C}^{\bar{H}}$, as well as an event Λ that is well-defined for either of the two executions. Then the equivalences $\mathcal{B}^{\bar{H}}[\Lambda] = \mathcal{C}^{\bar{H}}[\Lambda]$ and $\mathcal{B}^{\bar{H}}[\Lambda] \simeq \mathcal{C}^{\bar{H}}[\Lambda]$ are naturally defined by means of the equality of the respective *conditional* distributions. Furthermore, if Λ has the same probability $\Pr[\Lambda]$ in a run of $\mathcal{B}^{\bar{H}}$ and in a run of $\mathcal{C}^{\bar{H}}$, then

$$SD\big(\mathcal{B}^{\bar{H}}, \mathcal{C}^{\bar{H}}\big) \leq \Pr[\Lambda] \cdot SD\big(\mathcal{B}^{\bar{H}}[\Lambda], \mathcal{C}^{\bar{H}}[\Lambda]\big) + \Pr[\neg\Lambda] \cdot SD\big(\mathcal{B}^{\bar{H}}[\neg\Lambda], \mathcal{C}^{\bar{H}}[\neg\Lambda]\big)\,.$$

In particular, if $\mathcal{B}^{\bar{H}}[\Lambda] \simeq \mathcal{C}^{\bar{H}}[\Lambda]$ then $SD\big(\mathcal{B}^{\bar{H}}, \mathcal{C}^{\bar{H}}\big) \leq \Pr[\neg\Lambda]$.

We conclude by noting that in case of an oracle algorithm $\mathcal{A}^{\bar{H},\bullet}$, an instantiation $\mathcal{O}^{\bar{H}}$ of the oracle, and an event Λ defined by the local randomness of $\mathcal{O}$ (for any fixed input to $\mathcal{O}$), if $\mathcal{A}^{\bar{H},\bullet}$ is promised to make precisely *one* query to the oracle then the event Λ is also well-defined by a run of $\mathcal{A}^{\bar{H},\mathcal{O}^{\bar{H}}}$, and $\mathcal{A}^{\bar{H},\mathcal{O}[\Lambda]^{\bar{H}}} = \mathcal{A}^{\bar{H},\mathcal{O}^{\bar{H}}}[\Lambda]$.

2.2.3 Simulating the Write Access

It is not too hard to see that write queries to H can always be simulated internally, in the sense that there exists an (adaptively stateful) algorithm $\mathcal{S}^{H}$, which makes one H-query whenever it is invoked with a read request (and no query upon any write request), and such that $\mathcal{A}^{\bar{H}} \simeq \mathcal{A}^{\mathcal{S}^{H}}$ for every $\mathcal{A}^{\bar{H}}$.

[3] The former is more common in the cryptography literature, while the advantage of the latter is its succinctness.

$\mathcal{S}$ simply simulates any write query to H by bookkeeping the reprogramming requests and properly adjusting the future read queries to H, which it relays. It is obvious how this works in case $\mathcal{A}$ makes *classical* read queries to H, but it can also be done in case of quantum read queries, where $\mathcal{S}$ then is a suitable quantum algorithm.

Lemma 2.1. *There exists a stateful oracle algorithm $\mathcal{S}^H$, which makes 1 read query per each read-query invocation, and such that $\mathcal{A}^{\bar{H}} = \mathcal{A}^{\mathcal{S}^H}$ for every $\mathcal{A}^{\bar{H}}$.*

Proof. The oracle algorithm $\mathcal{S}^H$ operates by bookkeeping a list of classical write queries of $\mathcal{A}$. Upon receiving a write query (x, y), it first removes any occurrences of the form $(x, \cdot)$ from the list (if there exists any) and then inserts (x, y) into the list. Answering a read query then can be performed by $\mathcal{S}^H$, by invoking one single query to H, as follows. Given a list $\{(x_1, y_1), \ldots, (x_L, y_L)\} \subset \mathcal{X} \times \mathcal{Y}$ of classical write queries with pairwise distinct x_i's, $\mathcal{S}^H$ (efficiently) computes the unitary $|x, y\rangle \mapsto |x, y + H'(x)\rangle$ by means of a quantum algorithm that makes a single quantum query to H, i.e. to the unitary $|x, y\rangle \mapsto |x, y + H(x)\rangle$, for H' defined as

$$H'(x) = \begin{cases} y_i & \text{if } \exists\, i : x = x_i\,, \\ H(x) & \text{otherwise}\,. \end{cases}$$

This is trivial if $\mathcal{S}$ is given *control* access to H, i.e., access to the unitary $|b, x, y\rangle \mapsto |b, x, y + bH(x)\rangle$. Furthermore, control access to H can be simulated using a single ordinary quantum access to H as follows: $\mathcal{S}$ first prepares a uniform superposition of elements in $\mathcal{Y}$ in an auxiliary register $|z\rangle$, applies a control swap to $|y, z\rangle$ with $|b\rangle$ being the control bit, makes the quantum query to H, and finally applies a control swap again. One can conclude by noting that $\mathcal{S}^H$ perfectly simulates the view of the attacker $\mathcal{A}$. □

By default, we then write $\bar{\mathcal{A}}^H := \mathcal{A}^{\mathcal{S}^H}$ for the algorithm that simulates $\mathcal{A}$'s write queries internally, and thus is such that $\bar{\mathcal{A}}^H \simeq \mathcal{A}^{\bar{H}}$.[4] We note that in case of an oracle algorithm $\mathcal{A}^{\bar{H},\bullet}$ and setting $\bar{\mathcal{A}}^{H,\bullet} := \mathcal{A}^{\mathcal{S}^H,\bullet}$, the equality $\bar{\mathcal{A}}^{H,\mathcal{O}^{\bar{H}}} \simeq \mathcal{A}^{\bar{H},\mathcal{O}^{\bar{H}}}$ of the output distributions holds (only) *conditioned* on the event that $\mathcal{O}$ makes no (read or write) query to H on a point that has previously been reprogrammed by $\mathcal{A}$. Thus, $\bar{\mathcal{A}}^{H,\mathcal{O}^{\bar{H}}}[\Omega] \simeq \mathcal{A}^{\bar{H},\mathcal{O}^{\bar{H}}}[\Omega]$ with Ω being the said event.

[4] Here the respective output distributions of the two algorithms are actually equal *for any choice of H*; thus, we have an equivalence that lies in-between $=$ and $\simeq$, but this is not important to us.

Chapter 3

Fiat-Shamir and Hash-and-Sign with Aborts

3.1 Introduction

Fiat-Shamir-with-aborts (FSwA) and *probabilistic hash-and-sign with retry/abort* (HSwA) are important design principles for digital signature schemes (in the random oracle model), in particular for constructing quantum-secure signature schemes. Both have in common that the signature generation may require several trials until a "good" signature is obtained; informally speaking, the retrying is typically necessary in order to not leak unwanted information about the secret key via a "bad" choice of the signature. Examples of signature schemes that follow one or the other are Lyubashevsky's signature [Lyu09, Lyu12], GLP [GLP12], TESLA [ABB+17], Dilithium [DKL+18], SeaSign [DFG19a], and HAETAE [CCD+24] in the FSwA paradigm, and Hidden Field Equation (HFE) signatures [Pat96], Unbalanced Oil and Vinegar (UOV) [KPG99], the Courtois-Finiasz-Sendrier (CFS) signature [CFS01, Dal08], GeMSS [CFMR+17], Wave [DST19], MAYO [Beu21], and QR-UOV [FIKT21] in the HSwA paradigm.

The two design principles, FSwA and HSwA, also have in common that security is typically proven in two steps: first, the specific instantiation is exploited in order to show UF-NMA security, and then an argument that is generic for the design principle (and only requires some mild additional properties from the instantiation) is used to conclude full-fledged (strong or ordinary) UF-CMA security, i.e., security against chosen-message attacks.

Quite recently, a subtle but crucial flaw was independently and concurrently discovered by [BBD+23] and [DFPS23] in all prior UF-CMA-to-UF-NMA reductions of FSwA. The flaw applies both classically and quantum, and invalidates all prior security proofs of FSwA signature schemes, most notably leaving the future NIST PQC standard, Dilithium, without a valid proof. Moving on to

HSwA signature schemes, the authors of [KX24] followed up on the observation from [CDP23] that the original security reduction of HSwA in [SSH11] contains a similar flaw, and they provide the first correct such reduction in the ROM, covering both classical and quantum attacks. However, as will be seen later, their quantum analysis suffers a rather non-ideal security loss.

3.1.1 Our Contribution

Facing the aforementioned flaw, we re-establish the security analyses of FSwA and HSwA signature schemes in this chapter, to the extent possible. Specifically, we consider a unified, abstract class of digital signature schemes, which we call *generalized Fiat-Shamir with aborts* signature schemes, covering both FSwA and HSwA (see Section 3.2.1). We then describe the flaw in Section 3.3 under this abstract formalism, and provide a new generic UF-CMA-to-UF-NMA reduction, accompanied with various bounds that covers both classical and quantum attacks in the random oracle model.

A new security proof. In Section 3.4, we provide a new security proof in the ROM from scratch, reducing the UF-CMA security of a generalized FSwA scheme $\mathcal{S}$ to the UF-NMA security. On a very high level, our reduction relies on a simulator Sim that can efficiently simulate the signing procedure Sign without knowing the secret key. This may seem contradictory to the very definition of the UF-CMA security, but the catch here is that, as a part of our reduction, Sim is allowed to reprogram the random oracle $H(r, m) := y$, once it produces a suitable simulated transcript (r, y, z). With such a simulator in place, any UF-CMA attacker $\mathcal{A}$ can be transformed into a similarly efficient UF-NMA attacker $\mathcal{B}$, via forwarding each signing queries made by $\mathcal{A}$ to the simulator.

For this approach to work, it is sufficient to show that Sign and Sim are indistinguishable, even if one can query the random oracle H. Moreover, the UF-CMA-to-UF-NMA security loss is roughly upperbounded by the distinguishing advantage, i.e.

$$\mathbf{Adv}^{\mathsf{UF\text{-}CMA}}_{\mathcal{S}}(\mathcal{A}) - \mathbf{Adv}^{\mathsf{UF\text{-}NMA}}_{\mathcal{S}}(\mathcal{B}) \leq SD(\mathcal{A}^{H,\mathsf{Sign}}, \mathcal{A}^{H,\mathsf{Sim}}) \ ,$$

assuming $\mathcal{A}$ makes a few more queries for technical reasons. To prove this indistinguishability, our analyses proceed through a sophisticated hybrid argument, detailed in Section 3.4.2, with two additional intermediate oracles Prog and Trans come into the play. The main technical challenge is to show that $\mathcal{A}^{H,\mathsf{Sign}} \approx \mathcal{A}^{H,\mathsf{Prog}} \approx \mathcal{A}^{H,\mathsf{Trans}}$, while the closeness $\mathcal{A}^{H,\mathsf{Trans}} \approx \mathcal{A}^{H,\mathsf{Sim}}$ follows straightforwardly with advantage bounded by $q_S\zeta$, where q_S is the number of signing queries $\mathcal{A}$ can make, and ζ is a scheme-dependent parameter that is close to zero.

In the general case where $\mathcal{A}$ is allowed to make quantum queries, we obtain

$SD\left(\mathcal{A}^{H,\mathsf{Sign}}, \mathcal{A}^{H,\mathsf{Prog}}\right) \leq O\left(\frac{q_S\sqrt{q_H\epsilon}}{1-p}\right)$ and $SD\left(\mathcal{A}^{H,\mathsf{Prog}}, \mathcal{A}^{H,\mathsf{Trans}}\right) \leq O\left(q_H\sqrt{\frac{q_S\epsilon}{1-p}}\right)$, where q_H is the number of $\mathcal{A}$'s queries to the random oracle H, and p and ϵ are scheme-dependent parameters with $p \in [0,1)$ being not too close to 1, and ϵ being close-to-zero.[1] This leads to the following security loss

$$\mathbf{Adv}^{\mathsf{UF\text{-}CMA}}_{\mathcal{S}}(\mathcal{A}) - \mathbf{Adv}^{\mathsf{UF\text{-}NMA}}_{\mathcal{S}}(\mathcal{B}) \leq O\left(q_H\sqrt{\frac{q_S\epsilon}{1-p}} + \frac{q_S}{1-p}\sqrt{q_H\epsilon}\right) + q_S\zeta\ .$$

In the case where $\mathcal{A}$ is restricted to making classical queries only, via replacing relevant parts of the quantum analysis with a classical bound, we obtain a correspondingly tighter security loss

$$\mathbf{Adv}^{\mathsf{UF\text{-}CMA}}_{\mathcal{S}}(\mathcal{A}) - \mathbf{Adv}^{\mathsf{UF\text{-}NMA}}_{\mathcal{S}}(\mathcal{B}) \leq O\left(\frac{q_Hq_S\epsilon}{1-p} + \frac{q_S^2\epsilon}{(1-p)^2}\right) + q_S\zeta\ .$$

We note that, typically q_H is much bigger than q_S, e.g. in the NIST-2 security level, (q_S, q_H) can be all the way up to $(2^{64}, 2^{128})$. Therefore, the above quantum and classical bounds are dominated by $O\left(q_H\sqrt{\frac{q_S}{1-p}\epsilon}\right)$ and $O\left(\frac{q_Hq_S\epsilon}{1-p}\right)$ respectively (assuming $q_S\zeta$ is not too big). For the classical bound there is a matching attack where $SD(\mathcal{A}^{H,\mathsf{Sign}}, \mathcal{A}^{H,\mathsf{Sim}}) \geq \Omega(q_Hq_S\epsilon)$. However, (in the case where $q_S = 1$)[2] the best known quantum attack as presented in [GHHM21] only yields $SD(\mathcal{A}^{H,\mathsf{Sign}}, \mathcal{A}^{H,\mathsf{Sim}}) \geq \Omega(\sqrt{q_H\epsilon})$, with the q_H term inside of the square-root, suggesting that the quantum bound as stated above may still be suboptimal. This seemingly suboptimal loss also appears in prior analyses of HSwA [KX24] in the quantum setting.

Improving the quantum analysis. In Section 3.5, we further improve the hybrid step $\mathcal{A}^{H,\mathsf{Prog}} \approx \mathcal{A}^{H,\mathsf{Trans}}$ to $SD(\mathcal{A}^{H,\mathsf{Prog}}, \mathcal{A}^{H,\mathsf{Trans}}) \leq O\left(\frac{q_S}{1-p}\sqrt{q_H\epsilon}\right)$, which then leads to a tighter overall security loss in the quantum setting:

$$\mathbf{Adv}^{\mathsf{UF\text{-}CMA}}(\mathcal{A}) - \mathbf{Adv}^{\mathsf{UF\text{-}NMA}}(\mathcal{B}) \leq O\left(\frac{q_S}{1-p}\sqrt{q_H\epsilon}\right) + q_S\zeta\ .$$

Plugging in the numbers with $(q_S, q_H) = (2^{64}, 2^{128})$, and suppressing less relevant terms, we obtain $q_S\sqrt{q_H\epsilon} = 2^{128}\sqrt{\epsilon}$, an improvement against $q_H\sqrt{q_S\epsilon} = 2^{160}\sqrt{\epsilon}$ by a multiplicative factor of 2^{32}. This improved quantum bound as well as the above classical bound, matches the well-studied analyses of the original Fiat-Shamir signature schemes (without aborts).[3] Though it remains

[1] As a matter of fact, we allow these parameters to be key-dependent, which is omitted in this introduction for simplicity.

[2] The situation when q_S scales up remains unclear.

[3] In the atypical regime where q_S is much bigger than q_H, our quantum bound of order $O(q_S\sqrt{q_H\epsilon})$ even outperforms the standard bound of order $O(q_S\sqrt{(q_S+q_H)\epsilon})$.

open whether the quantum bound is optimal, any further improvement would also need to carry over to the better-understood setting without aborts.

At the core of our improvement is the following technical challenge (somewhat simplified here for the ease of exposition). Let $\mathcal{D}$ be a distribution over a set $\mathcal{R}$ with the promise that $\Pr[r = r_\circ] \leq \epsilon$ for any $r_\circ \in \mathcal{R}$ and $r \leftarrow \mathcal{D}$, and let f be an arbitrary (randomized or deterministic) function with domain $\mathcal{R} \times \mathcal{Y}$ and with a special symbol $\perp$ in its co-domain. Consider an arbitrary quantum algorithm $\mathcal{A}^H$ that gets a sample produced by one or the other of the following two procedures:

1: $r \leftarrow \mathcal{D}$		1: $r \leftarrow \mathcal{D}$
2: $H(r) := y \leftarrow \mathcal{Y}$		2: $y \leftarrow \mathcal{Y}$
3: $z \leftarrow f(r, y)$		3: $z \leftarrow f(r, y)$
4:	or	4: **if** $z \neq \perp$ **then** $H(r) := y$
5: **if** $z = \perp$ **then return** $\perp$		5: **if** $z = \perp$ **then return** $\perp$
6: **else return** (r, z)		6: **else return** (r, z)

and that can make superposition queries to the random oracle H *before* and *after* it gets the sample, say q_H in total. The goal now is to show that it is hard for $\mathcal{A}^H$ to decide from which of the two it got the sample.

We note that the only difference between the two is that the first procedure reprograms $H(r)$ to y no matter what, while the latter does so only in case of a non-$\perp$ output. Thus, intuitively it is clear that it is hard for $\mathcal{A}^H$ to distinguish the two: it can notice the difference only when the procedure outputs $\perp$ *and* $\mathcal{A}^H$ queries $H(r)$ after having received the sample (thus $\perp$); but the latter is unlikely to happen then due to the high entropy in r. However, making this a rigorous argument in the case of quantum queries results in a hybrid argument over all the queries to H (after having received the sample), where (for the sake of the argument) one would then measure for each query if the query is to r or not. This then leads to a distinguishing advantage of $q_H\sqrt{\epsilon}$, where the square-root comes from the Gentle-Measurement Lemma, used to argue that the measurements cause little disturbance, and the factor q_H is by quantifying over all queries. Thus, the real challenge is to prove a bound on the distinguishing advantage that scales as $\sqrt{q_H\epsilon}$ instead; this is what we aim for and achieve in this work.[4]

Concrete analyses of Dilithium. In Section 3.6, we perform a more elaborated concrete analysis on the NIST PQC standard Dilithium, where the parameter ϵ is better controlled. This is achieved both via a computer-aided

[4]One might also be tempted to argue that the two can only be distinguished by $\mathcal{A}^H$ if $\mathcal{A}^H$ has queried $H(r)$ *before* it receives the sample (and then use compressed-oracle techniques to get the q_H inside the square-root) — but then one falls for the same trap as earlier, faulty FSwA proofs: due to the *conditional* reprogramming of H in the second procedure, H may become non-uniformly random there. One expects this non-uniformity to be negligible and hard to notice for $\mathcal{A}^H$, but giving a concrete (and sufficiently good) bound appears difficult.

approach that yields better numerical results, and via a pen-and-paper version that is easier to verify, but with a slightly worse concrete bound.

The technical challenge that we address in order to control ϵ for Dilithium, is a rather innocent-looking mathematical problem. We consider a block diagonal square matrix $A^{\square} := D_1 \oplus \cdots \oplus D_n$ with each block $D_i \leftarrow \mathbb{F}_q^{\ell \times \ell}$ being an ℓ-by-ℓ random matrix over a finite field of order q, where concretely speaking $q \approx 2^{23}$ and $n\ell \in (1000, 2000)$. The goal here is to show that, with a cryptographically close-to-1 probability, the rank of $A^{\square}$ is very close to $n\ell$. We note that this is not as straightforward as it may seem. For instance, one may be tempted to argue that $A^{\square}$ is invertible with a probability close to 1, but this probability is in fact roughly $1 - n/q \approx 1 - n \cdot 2^{-23}$, which is not cryptographically close to 1.

3.2 Preliminaries

3.2.1 Generalized Fiat-Shamir with Aborts Signatures

Let $\mathcal{M}, \mathcal{R}, \mathcal{Y}$ and $\mathcal{Z}$ be arbitrary non-empty finite sets, and fix the domain and range of the random oracle to be $\mathcal{R} \times \mathcal{M}$ and $\mathcal{Y}$, respectively, i.e. $H : \mathcal{R} \times \mathcal{M} \to \mathcal{Y}$. Furthermore, let $\perp$ be some special symbol not contained in $\mathcal{Z}$. These sets may depend on a security parameter, but we leave this dependency — and the security parameter itself — implicit throughout most of the document.

We consider signature schemes $\left(\mathsf{KGen}, \mathsf{Sign}^H, \mathsf{Vrfy}^H\right)$ in the random oracle model of the following form. On input the security parameter, the key-generation algorithm KGen produces a key pair $(\mathsf{sk}, \mathsf{pk})$, which in turn specifies a distribution $\mathcal{D}$ over a set $\mathcal{R}$, an ensemble $\{f(r, y)\}_{r \in \mathcal{R}, y \in \mathcal{Y}}$ of distributions over $\mathcal{Z} \cup \{\perp\}$, and a predicate $\mathcal{V} : \mathcal{R} \times \mathcal{Y} \times \mathcal{Z} \to \{0, 1\}$. Signing of a message $m \in \mathcal{M}$ works as specified in Fig. 3.1 below, and the verification Vrfy, given a claimed signature $\sigma = (r, z)$ for a message $m \in \mathcal{M}$, accept if and only if $\mathcal{V}(r, H(r, m), z) = 1$.

Remark 3.1. *How the predicate $\mathcal{V}(r, y, z)$ works is irrelevant for this chapter, but a natural way is to check that z falls in the support of the distribution $f(r, y)$.*

$\mathsf{Sign}^H(\mathsf{sk}, m)$:
1: **repeat**
2: $\quad r \leftarrow \mathcal{D}$
3: $\quad y := H(r, m)$
4: $\quad z \leftarrow f(r, y)$
5: **until** $z \neq \perp$
6: **return** (r, z)

Figure 3.1: The signing procedure, where the hash function H in step 3 is modelled as a random oracle. The dependency of $\mathcal{D}$ and f on sk is left implicit.

Obviously, for signing to be efficient, it is necessary that $\mathcal{D}$ is efficiently sampleable when given the secret key sk, and $f(r, y)$ is efficiently sampleable (for any r and y) when given sk and the randomness used to sample $r \leftarrow \mathcal{D}$. For verification to be efficient, it needs to be efficiently testable given the public key (only) if z is in the support of $f(r, y)$, for any r, y, z. Similarly, for the scheme to be secure (against key-only attacks) it is necessary that signing should be computationally hard when only given the public key pk. However, these (in)efficiency aspects are not our concern; our security reductions also apply for non-efficient or insecure schemes (though they become somewhat pointless then). Therefore, we keep the public key pk and the secret key sk implicit — unless specified otherwise, we consider them arbitrary but fixed — and so we also keep the dependency of $\mathcal{D}$ and f on $(\mathsf{pk}, \mathsf{sk})$, implicit; the same for the parameters p and ϵ below.

Two important parameters for such a signature scheme are

$$p := \Pr_{\substack{r \leftarrow \mathcal{D}, y \leftarrow \mathcal{Y} \\ z \leftarrow f(r,y)}} [\, z = \perp \,], \tag{3.1}$$

referred to as the *abort probability*, and

$$\epsilon := \mathsf{guess}(r) := \max_{r^\circ} \Pr_{r \leftarrow \mathcal{D}} [\, r = r^\circ \,], \tag{3.2}$$

which is the *guessing probability* of the distribution $\mathcal{D}$.

The above abstract signature scheme design is well motivated by the fact that it covers both *Fiat-Shamir with aborts* (FSwA) signature schemes, as well as *probabilistic hash-and-sign with retry/abort* (HSwA) signature schemes. In the case of FSwA signatures, f is usually a deterministic function, which can be efficiently computed given the secret key and the randomness used to sample r (the "first message" in the Σ-protocol); in the case of HSwA signatures, $\mathcal{D}$ is typically the uniform distribution over strings of a certain length, and f is the preimage-sampling algorithm of a so-called weak preimage-sampleable function.

As a matter of fact, a HSwA signature can be understood as the FSwA signature obtained from the Σ-protocol that chooses a random bit string r as the "first message", and samples the response z as a preimage of the (random) challenge under the weak preimage-sampleable function. We therefore call the general class of signature schemes considered here (covering FSwA and HSwA signatures) *generalized* FSwA signature schemes.

To formally capture that such an honestly generated signature leaks nothing about the secret key, we define the following *accepting honest-verifier zero-knowledge* (acHVZK) property.

Definition 3.2 (Accepting (Statistical) Honest-Verifier Zero-Knowledge)**.** *Let $\zeta > 0$ and $T \in \mathbb{Z}_{\geq 0}$. A generalized FSwA signature scheme $\Sigma = (\mathsf{KGen}, \mathcal{D}, f, \mathcal{V})$ is called (ζ, T)-acHVZK if there exists an algorithm acSim with runtime T, such that on input a public key pk it outputs a triple $(\hat{r}, \hat{y}, \hat{z})$ that (on average over the choice of pk generated by KGen) is ζ-close in statistical distance to (r, y, z) conditioned on $z \neq \bot$, where $r \leftarrow \mathcal{D}$, $y \leftarrow \mathcal{Y}$ and $z \leftarrow f(r, y)$.*

Remark 3.3. *In the above definition, we consider the statistical distance for fixed pk, then average it over the randomness of pk.*

3.2.2 A variant of the Adaptive Reprogramming Lemma

We consider the following, slightly extended variant of the Adaptive Reprogramming Lemma from [GHHM21]. It differs from the original variant in that, next to the quantum read queries, we allow the distinguisher to make (classical) write queries to H (with a bound on the *expected* number of write queries).

Lemma 3.4. *Let $\epsilon > 0$, and let $\{D_i\}_{i \in \mathcal{I}}$ be a family of distributions over $\mathcal{X}$ indexed by a finite set $\mathcal{I}$, such that*

$$\underset{x \leftarrow D_i}{\mathsf{guess}}(x) := \max_{x^\circ \in \mathcal{X}} \Pr_{x \leftarrow D_i}[x = x^\circ] \leq \epsilon$$

for all $i \in \mathcal{I}$. Let $\mathcal{A}^{\bar{H}, \blacksquare}$ be an oracle algorithm that makes one query to an oracle ($\blacksquare$), which is to be instantiated by $\mathcal{O}_0^H$ or $\mathcal{O}_1^{\bar{H}}$ as specified in Fig. 3.2; furthermore, prior to that query, $\mathcal{A}$ makes at most q_r quantum read queries to H, and in expectation *at most q_w classical write queries to H, for given positive numbers $q_r, q_w \in \mathbb{Z}$.[5] Then*

$$\left| \Pr\left[1 \leftarrow \mathcal{A}^{\bar{H}, \mathcal{O}_0^H}\right] - \Pr\left[1 \leftarrow \mathcal{A}^{\bar{H}, \mathcal{O}_1^{\bar{H}}}\right] \right| \leq \left(2q_w + \frac{q_r}{2}\right)\epsilon + \sqrt{q_r \epsilon}\,.$$

[5] We allow arbitrary many read/write H-queries *after* the query to $\mathcal{O}_0$ or $\mathcal{O}_1$.

$\mathcal{O}_0^H(i)$:	$\mathcal{O}_1^{\bar{H}}(i)$:
1: $x \leftarrow \mathcal{D}_i$	1: $x \leftarrow \mathcal{D}_i$
2: $y := H(x)$	2: $H(x) := y \leftarrow \mathcal{Y}$
3: **return** (x, y)	3: **return** (x, y)

Figure 3.2: Reprogramming or not reprogramming, that is the question.

The proof for this extended variant is a quite simple reduction to the original version [GHHM21], exploiting that we can simulate the write queries.

Proof. Consider $\bar{\mathcal{A}}^{H,\mathcal{O}_0^H} = \mathcal{A}^{S^H,\mathcal{O}_0^H}$ and $\bar{\mathcal{A}}^{H,\mathcal{O}_1^{\bar{H}}} = \mathcal{A}^{S^H,\mathcal{O}_1^{\bar{H}}}$, which run $\mathcal{A}$ and simulate the write queries locally, and let Ω be the event that $\mathcal{A}$ has not made a write query (prior to the oracle call) for the point s sampled by the oracle then. This event is well defined and has the same probability in any of the executions of $\mathcal{A}^{\bar{H},\mathcal{O}_0}, \bar{\mathcal{A}}^{H,\mathcal{O}_0}, \bar{\mathcal{A}}^{H,\mathcal{O}_1}, \mathcal{A}^{\bar{H},\mathcal{O}_1}$. Furthermore,

$$\bar{\mathcal{A}}^{H,\mathcal{O}_0^H}[\Omega] \simeq \mathcal{A}^{\bar{H},\mathcal{O}_0^H}[\Omega] \qquad \text{and} \qquad \bar{\mathcal{A}}^{H,\mathcal{O}_1^{\bar{H}}}[\Omega] \simeq \mathcal{A}^{\bar{H},\mathcal{O}_1^{\bar{H}}}[\Omega]\,.$$

It thus follows that

$$\begin{aligned}
&SD(\mathcal{A}^{\bar{H},\mathcal{O}_0^H}, \mathcal{A}^{\bar{H},\mathcal{O}_1^{\bar{H}}}) \\
&\leq SD(\mathcal{A}^{\bar{H},\mathcal{O}_0^H}, \bar{\mathcal{A}}^{H,\mathcal{O}_0^H}) + SD(\bar{\mathcal{A}}^{H,\mathcal{O}_0^H}, \bar{\mathcal{A}}^{H,\mathcal{O}_1^{\bar{H}}}) + SD(\bar{\mathcal{A}}^{H,\mathcal{O}_1^{\bar{H}}}, \mathcal{A}^{\bar{H},\mathcal{O}_1^{\bar{H}}}) \\
&\leq 2\Pr[\neg\Omega] + SD(\bar{\mathcal{A}}^{H,\mathcal{O}_0^H}, \bar{\mathcal{A}}^{H,\mathcal{O}_1^{\bar{H}}}) \leq 2q_w\epsilon + \frac{q_r}{2}\epsilon + \sqrt{q_r\epsilon}\,.
\end{aligned}$$

where the bound on $SD(\bar{\mathcal{A}}^{H,\mathcal{O}_0^H}, \bar{\mathcal{A}}^{H,\mathcal{O}_1^{\bar{H}}})$ follows from the standard adaptive reprogramming lemma (see Proposition 2 in [GHHM21]). □

3.3 A Gap in Prior Analyses of FSwA

A gap in prior analyses of FSwA occurs in the UF-CMA-to-UF-NMA reduction. In this step, signature queries made by the considered UF-CMA-attacker $\mathcal{A}$, which has query access to a signing oracle $\mathsf{Sign}(\mathsf{sk}, \cdot)$ and a random oracle H, must be answered without knowledge of the secret key, replacing real signatures with simulated ones produced by an Honest-Verifier Zero Knowledge (HVZK) simulator associated with the sigma protocol. To ensure that the attacker cannot detect that it is being given simulated signatures, it is also necessary to reprogram the random oracle to be consistent with the transcripts produced by the simulator. The crucial step boils down to replacing the oracle Sign by the oracle Trans (see Fig. 3.3).

$\mathsf{Sign}(\mathsf{sk}, m)$	$\mathsf{Trans}(\mathsf{sk}, m)$
1: **repeat**	1: **repeat**
2: $\quad r \leftarrow \mathcal{D}$	2: $\quad r \leftarrow \mathcal{D}$
3: $\quad y := H(r, m)$	3: $\quad y \leftarrow \mathcal{Y}$
4: $\quad z \leftarrow f(r, y)$	4: $\quad z \leftarrow f(r, y)$
5: **until** $z \neq \perp$	5: **until** $z \neq \perp$
6:	6: $H(r, m) := y$
7: **return** (r, z)	7: **return** (r, z)

Figure 3.3: Oracles Sign and Trans.

Clearly, the adversary $\mathcal{A}$ can attempt to guess r and query H on r before calling Sign/Trans, and then detect the inconsistency introduced by the reprogramming in case of Trans. However, even if the adversary makes no prior H-queries, the distribution of the random oracle changes, and this is where the gap lies. The reprogramming in Trans only reprograms the random oracle with accepting transcripts and thereby shifts the random oracle slightly towards pairs $((r, m), y)$ such that $f(r, y) \neq \perp$ with higher probability. Even though one expects this change in the distribution of the random oracle to be small, there is still a gap that needs to be properly bounded.

Both Lyubashevsky [Lyu12] and KLS [KLS18] miss the loss incurred by the bias in H in their analysis. In [Lyu12] this is missed in the hop from the real signing oracle to Hybrid 1 in the proof of Lemma 5.3 — note that the bound in [Lyu12] remains correct due to a loose analysis. In [KLS18] the gap is missed in the game hop from G_0 to G_1 in the proof of Theorem 3.2. Moreover, this oversight is not a problem limited to [Lyu12] and [KLS18], and it potentially affects all FS-based schemes involving rejection sampling. This includes a long list of works [LNP22, DKL$^+$18, DFG19b, BKP20, BDK$^+$22] on lattice-based and isogeny-based signature schemes (and non-interactive proof systems) that need to be re-examined carefully.

3.4 A UF-CMA-to-UF-NMA Reduction

Throughout this section, and the rest of the chapter, let Σ be an aborting sigma protocol, and $\mathcal{S} := \mathsf{FSwA}[\Sigma, H]$ be the corresponding FSwA signature scheme, as introduced in Sect. 3.1, with parameters ϵ and p defined as in (3.1) and (3.2). We assume $\mathcal{S}$ to be (ζ, T)-acHVZK for given ζ and T.

Following standard notation, we write $\mathbf{Adv}^{\mathsf{UF\text{-}CMA}}_{\mathcal{S}}(\mathcal{A})$ for the advantage of an attacker $\mathcal{A}$ of winning the standard UF-CMA security game (in the QROM) for the scheme $\mathcal{S}$, and similarly $\mathbf{Adv}^{\mathsf{UF\text{-}NMA}}_{\mathcal{S}}(\mathcal{B})$ for the advantage of an attacker $\mathcal{B}$ of winning the standard UF-NMA security game.

3.4.1 The Statements

Our main result is a UF-CMA-to-UF-NMA reduction. The reductions loss is in terms of (bounds on) the parameters p and ϵ. Since these parameters are in general key-dependent, we first state the improved reduction loss for a fixed choice of key pair $(\mathsf{sk}, \mathsf{pk})$, and we write p_{sk} and ϵ_{sk} when we want to make the dependency on the choice of the key explicit (where we assume without loss of generality that pk is determined by sk).

Similarly, we write ζ_{sk} for the statistical distance of the simulated transcript to the actual accepted transcript *for the specific choice* sk *of the key pair*; it then obviously holds that $\mathbb{E}[\zeta_{\mathsf{sk}}] = \zeta$, with the expectation taken over $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}$. Finally, we write $\mathbf{Adv}^{\mathsf{UF\text{-}CMA}}_{\mathcal{S}}(\mathcal{A}, \mathsf{sk})$ and $\mathbf{Adv}^{\mathsf{UF\text{-}NMA}}_{\mathcal{S}}(\mathcal{B}, \mathsf{sk})$ for the respective advantages when the key is chosen to be sk.

The proof of the following main theorem is presented in the subsequent subsections.

Theorem 3.5. *Let $\mathcal{S}$ be a generalized FSwA signature scheme. Then for every* UF-CMA *attacker $\mathcal{A}^{H,\bullet}$ making at most Q_H quantum queries to H and q_S classical queries to the signing oracle, there exists an* UF-NMA *attacker $\mathcal{B}^H$ making at most Q_H quantum queries to H such that for every fixed choice of key sk with $p_{\mathsf{sk}} < 1$, and for $q_H = Q_H + 1$ we have*

$$\mathbf{Adv}^{\mathsf{UF\text{-}CMA}}_{\mathcal{S}}(\mathcal{A}, \mathsf{sk}) \leq \mathbf{Adv}^{\mathsf{UF\text{-}NMA}}_{\mathcal{S}}(\mathcal{B}, \mathsf{sk}) + \frac{3q_S\sqrt{q_H\epsilon_{\mathsf{sk}}}}{1-p_{\mathsf{sk}}} + 2q_H\sqrt{\frac{q_S\epsilon_{\mathsf{sk}}}{1-p_{\mathsf{sk}}}} + q_S\zeta_{\mathsf{sk}} \,.$$

Moreover, if we count runtime in terms of the number of gates, except that each arithmetic operation on $\mathcal{X}$ and $\mathcal{Y}$ and every comparison among them (with respect to a strict total ordering) are counted as unit runtime, then $\mathsf{TIME}(\mathcal{B}) \leq \mathsf{TIME}(\mathcal{A}) + O\big(\, q_S T + q_H q_S + q_S^2 \,\big)$.

Remark 3.6. *Suppose $\mathcal{B}$ is allowed to use a QRAM, where each cell may contain an element of $\mathcal{X} \times \mathcal{Y}$, up to $O(1)$ many memory pointers and up to $O(1)$ many auxiliary bits. If we count each arithmetic operation and each comparison of the memory pointers as being unit runtime, then $\mathcal{B}$ can further achieve the runtime* $\mathsf{TIME}(\mathcal{B}) \leq \mathsf{TIME}(\mathcal{A}) + O\big(\, q_S T + (q_S + q_H)\log q_S \,\big)$ *using only $O(q_S)$ many cells.*

When taking the expectation over sk on both sides, in order to get the average reduction loss for a random key-pair, one can apply Jensen's inequality to $\mathbb{E}[\sqrt{\epsilon_{\mathsf{sk}}}]$ get a bound in terms of the expectation of ϵ_{sk} (over the choice of sk). Unfortunately, this does not work for the parameter p_{sk}, where Jensen's inequality goes the wrong way round. Hence, we need to have a bound $\bar{p}$ on p_{sk} that holds for all sk, or holds except with small probability (over the choice of sk).

Towards optimizing the bound, it may also make sense to avoid some bad, yet unlikely, choices of $(\mathsf{sk}, \mathsf{pk})$ that make ϵ_{sk} large, i.e. to consider a bound $\bar{\epsilon}$

on the sub-normalized conditional expectation $\Pr[\mathsf{sk} \in \Gamma_\epsilon]\mathbb{E}[\epsilon_{\mathsf{sk}}|\mathsf{sk} \in \Gamma_\epsilon]$, where Γ_ϵ is a subset of the keys for which $\Pr[\mathsf{sk} \notin \Gamma_\epsilon]$ is small.

Altogether, this then gives the following statement.

Corollary 3.7. *Let $\mathcal{S}$ be a generalized FSwA signature scheme that is (ζ, T)-acHVZK. Furthermore, let Γ_ϵ and Γ_p be subsets of keys sk such that $p_{\mathsf{sk}} \leq \bar{p}$ for all $\mathsf{sk} \in \Gamma_p$ and $\Pr[\mathsf{sk} \in \Gamma_\epsilon]\mathbb{E}[\epsilon_{\mathsf{sk}}|\mathsf{sk} \in \Gamma_\epsilon] \leq \bar{\epsilon}$ for parameters $0 < \bar{\epsilon}, \bar{p} < 1$. Then for every $\mathsf{UF\text{-}CMA}$ attacker $\mathcal{A}$ making at most Q_H quantum queries to H and q_S classical queries to the signing oracle, the $\mathsf{UF\text{-}NMA}$ attacker $\mathcal{B}$ (dependent on $\mathcal{A}$) as defined in Theorem 3.5 is such that the following holds for $q_H := Q_H + 1$:*

$$\mathbf{Adv}^{\mathsf{UF\text{-}CMA}}_{\mathcal{S}}(\mathcal{A}) \leq \mathbf{Adv}^{\mathsf{UF\text{-}NMA}}_{\mathcal{S}}(\mathcal{B}) + \frac{3q_S\sqrt{q_H\bar{\epsilon}}}{1-\bar{p}} + 2q_H\sqrt{\frac{q_S\bar{\epsilon}}{1-\bar{p}}} + q_S\zeta + \Pr[\mathsf{sk} \notin \Gamma_p] + \Pr[\mathsf{sk} \notin \Gamma_\epsilon] \,.$$

3.4.2 Proof Strategy

Towards proving the above claim, we consider an arbitrary $\mathsf{UF\text{-}CMA}$ attacker $\mathcal{A}^{H,\bullet}$, where by default the queries to the oracle $\bullet$ are answered by the signing algorithm/oracle Sign in the obvious way.[6] It will be convenient to assume that $\mathcal{A}$ makes one more query to H (which are $q_H = Q_H + 1$ queries in total) in order to check himself if the forged signature is valid under a new message that has not been queried, and then aborts (i.e. outputs $\perp$) if the check fails.

Our goal is to show that

$$\mathcal{A}^{H,\mathsf{Sign}^H(\mathsf{sk},\cdot)}(\mathsf{pk}) \approx \mathcal{A}^{H,\mathsf{Sim}^{\bar{H}}(\mathsf{pk},\cdot)}(\mathsf{pk}) \tag{3.3}$$

with an upper bound on the distance that is in line with the security loss in the theorem statement. Here, Sim simulates the signing oracle by exploiting the non-abort ZK property and the ability to reprogram H, as specified in Fig. 3.4 below.

$\mathsf{Sim}^{\bar{H}}(\mathsf{pk}, m)$:
1: $(\hat{r}, \hat{y}, \hat{z}) \leftarrow \mathsf{acSim}(\mathsf{pk})$
2: $H(\hat{r}, m) := \hat{y}$
3: **return** $(\hat{r}, \hat{z})$

Figure 3.4: Simulating the signing oracle by means of the acHVZK simulator and reprogramming H.

This then implies that the $\mathsf{UF\text{-}NMA}$ attacker $\bar{\mathcal{B}}^H(\mathsf{pk}) = \mathcal{B}^{\mathcal{S}^H}(\mathsf{pk})$ obtained by running $\mathcal{B}^{\bar{H}}(\mathsf{pk}) := \mathcal{A}^{H,\mathsf{Sim}^{\bar{H}}(\mathsf{pk},\cdot)}(\mathsf{pk})$ but simulating the write queries to H

[6] I.e., using the secret key that corresponds to the public key that is given to $\widetilde{\mathcal{A}}$ as input.

internally, is similarly successful in forging a signature as the original UF-CMA attacker $\mathcal{A}$. The crucial property of Sim is of course that it does not need the secret key, and so can indeed be simulated by $\mathcal{B}$ itself.

By assumption on $\mathcal{A}$ (to verify the forged signature before outputting it), we know that $\mathcal{B}^{\bar{H}}(\mathsf{pk})$ outputs a forgery σ^* for a message m^* that correctly verifies under the reprogrammed oracle H (or else outputs $\perp$). However, since H gets reprogrammed only at places $(\hat{r}, m)$ for $m^* \neq m$, σ^* also verifies under the original (unreprogrammed) choice of H. Consequently, whenever $\bar{\mathcal{B}}$ outputs non-$\perp$, it outputs a valid forgery. Thus, we have

$$\begin{aligned}\mathbf{Adv}^{\mathsf{UF\text{-}NMA}}(\bar{\mathcal{B}}) &= \Pr\left[\bar{\mathcal{B}}^{H}(\mathsf{pk}) \neq \perp\right] = \Pr\left[\mathcal{B}^{\bar{H}}(\mathsf{pk}) \neq \perp\right] = \Pr\left[\mathcal{A}^{H,\mathsf{Sim}^{\bar{H}}(\mathsf{pk},\cdot)}(\mathsf{pk}) \neq \perp\right] \\ &\geq \mathbf{Adv}^{\mathsf{UF\text{-}CMA}}(\mathcal{A}) - SD\Big(\mathcal{A}^{H,\mathsf{Sign}^{H}(\mathsf{sk},\cdot)}(\mathsf{pk}), \mathcal{A}^{H,\mathsf{Sim}^{\bar{H}}(\mathsf{sk},\cdot)}(\mathsf{pk})\Big)\,, \qquad (3.4)\end{aligned}$$

where the second and third equalities follow $\bar{\mathcal{B}}^{H}(\mathsf{pk}) \simeq \mathcal{B}^{\bar{H}}(\mathsf{pk}) = \mathcal{A}^{H,\mathsf{Sign}^{\bar{H}}(\mathsf{sk},\cdot)}(\mathsf{pk})$.

Remark 3.8. *If we aim for* strong *unforgeability, similar argument applies, but we additionally require what is known as the* computational unique-response property*, which prevents an efficient attacker to come up with two valid triples $(r, y, z_1), (r, y, z_2)$ with the same first message r and the same challenge y but distinct responses $z_1 \neq z_2$*

Towards showing the closeness (3.3), we introduce intermediate oracles Prog and Trans between Sign and Sim as in Fig. 3.5. The closeness (3.3) is to be shown by means of the following sequence of closeness results:

$$\mathcal{A}^{H,\mathsf{Sign}^{H}(\mathsf{sk},\cdot)}(\mathsf{pk}) \overset{(a)}{\approx} \mathcal{A}^{H,\mathsf{Prog}^{H}(\mathsf{sk},\cdot)}(\mathsf{pk}) \overset{(b)}{\approx} \mathcal{A}^{H,\mathsf{Trans}^{H}(\mathsf{sk},\cdot)}(\mathsf{pk}) \overset{(c)}{\approx} \mathcal{A}^{H,\mathsf{Sim}(\mathsf{pk})^{\bar{H}}}(\mathsf{pk})\,.$$

The closeness claims (a) and (b) are to be shown in Sections 3.4.3 and 3.4.4 respectively, and the closeness claim (c) follows directly from the defining properties of the non-abort ZK simulator; with distance ζ_{sk} for a fixed choice sk of the key, and with distance ζ on average.

$\mathsf{Sign}^{H}(\mathsf{sk}, m)$:	$\mathsf{Prog}^{\bar{H}}(\mathsf{sk}, m)$:	$\mathsf{Trans}^{\bar{H}}(\mathsf{sk}, m)$:
repeat	**repeat**	**repeat**
$r \leftarrow \mathcal{D}$	$r \leftarrow \mathcal{D}$	$r \leftarrow \mathcal{D}$
$y := H(r, m)$	$H(r, m) := y \leftarrow \mathcal{Y}$	$y \leftarrow \mathcal{Y}$
$z := f(r, y)$	$z := f(r, y)$	$z := f(r, y)$
until $z \neq \perp$	**until** $z \neq \perp$	**until** $z \neq \perp$
		$H(r, m) := y$
return (r, z)	**return** (r, z)	**return** (r, z)

Figure 3.5: The oracles Prog, Trans compared with Sign.

Looking ahead, both (a) and (b) hold even for fixed choices of sk and pk; we now consider them arbitrary but fixed in the remainder of this work, and we do not write them explicitly anymore as input to $\mathcal{A}, \mathsf{Sign}$ etc., and we leave the dependency of p and ϵ on the key implicit again.

3.4.3 From Sign to Prog

For $\mathcal{A}$ making q_S queries to Sign, let $\mathcal{G}_i^{\bar{H},\bullet} := \mathcal{A}^{H,[(\mathsf{Prog})^{i-1},\bullet,(\mathsf{Sign}^{\bar{H}})^{q_S-i}]}$ be a run of $\mathcal{A}$ such that the first $i-1$ signing queries are answered by Prog, the ith signing query is answered by an unspecified oracle (which will later be instantiated either by Sign or by Prog), and all remaining signing queries are answered by Sign. Our goal here, is to prove the closeness of $\mathcal{G}_i^{\bar{H},\mathsf{Sign}} \approx \mathcal{G}_i^{\bar{H},\mathsf{Prog}}$. For that purpose, we consider the following sequence of intermediate oracles $O_j^{\blacklozenge} := \mathsf{Loop}^{[(\mathsf{B_P})^{j-1},\blacklozenge,(\mathsf{B_S})^{\infty}]}$ for every $j \in \mathbb{Z}_{>0}$, where $O_1^{\mathsf{B_S}} = \mathsf{Loop}^{\mathsf{B_S}} = \mathsf{Sign}$ and $O_\infty^{\blacklozenge} = \mathsf{Loop}^{\mathsf{B_P}} = \mathsf{Prog}$ regardless of the instantiation of $\blacklozenge$, and so we might just denote the latter as O_∞. Our proof proceeds as outlined below:

$$
\begin{aligned}
\mathcal{A}^{H,\mathsf{Sign}} &= \mathcal{G}_1^{\bar{H},\mathsf{Sign}} = \mathcal{G}_1^{\bar{H},O_1^{\mathsf{B_S}}} \approx \cdots \approx \mathcal{G}_1^{\bar{H},O_k^{\mathsf{B_S}}} \approx \mathcal{G}_1^{\bar{H},O_\infty} \\
&= \mathcal{G}_2^{\bar{H},\mathsf{Sign}} = \mathcal{G}_2^{\bar{H},O_2^{\mathsf{B_S}}} \approx \cdots \approx \mathcal{G}_2^{\bar{H},O_k^{\mathsf{B_S}}} \approx \mathcal{G}_2^{\bar{H},O_\infty} \\
&\vdots \\
&= \mathcal{G}_{q_S}^{\bar{H},\mathsf{Sign}} = \mathcal{G}_{q_S}^{\bar{H},O_{q_S}^{\mathsf{B_S}}} \approx \cdots \approx \mathcal{G}_{q_S}^{\bar{H},O_k^{\mathsf{B_S}}} \approx \mathcal{G}_{q_S}^{\bar{H},O_\infty} = \mathcal{A}^{H,\mathsf{Prog}} ,
\end{aligned}
$$

where $k \in \mathbb{Z}_{>0}$ and the closenesses are to be shown in Lemmas 3.9 and 3.10.

$\mathsf{Loop}^{\blacklozenge}(m)$:	$\mathsf{B_S}^{H}(m)$:	$\mathsf{B_P}^{\bar{H}}(m)$:
1: **repeat**	1: $r \leftarrow \mathcal{D}$	1: $r \leftarrow \mathcal{D}$
2: $\quad out \leftarrow \blacklozenge(m)$	2: $y := H(r,m)$	2: $H(r,m) := y \leftarrow \mathcal{Y}$
3: **until** $out \neq \bot$	3: $z \leftarrow f(r,y)$	3: $z \leftarrow f(r,y)$
4: **return** out	4: **if** $z = \bot$ **return** $\bot$	4: **if** $z = \bot$ **return** $\bot$
	5: **return** (r,z)	5: **else return** (r,z)

Figure 3.6: The repetition loop $\mathsf{Loop}^{\blacklozenge}$ (left), and different instantiations of the body of the loop $(\mathsf{B_S}, \mathsf{B_P})$.

Lemma 3.9. *Let $\mathcal{A}^{H,\bullet}$ be given q_H quantum queries to H and q_S classical queries to $\bullet$. Then for every $i \in [q_S]$ and $j \in \mathbb{Z}_{\geq 0}$ we have*

$$
SD\left(\mathcal{G}_i^{\bar{H},O_j^{\mathsf{B_S}}}, \mathcal{G}_i^{\bar{H},O_\infty}\right) \leq p^{j-1} .
$$

Proof. Let E_j be the event that the jth iteration of O_j is reached. Conditioned on $\neg E_j$, the oracles behave identically, i.e. $O_j[\neg E_j] = O_{j+1}[\neg E_j]$. Hence we have

$$SD\left(\mathcal{G}_i^{\bar{H},O_j^{\mathsf{B_S}}}, \mathcal{G}_i^{\bar{H},O_\infty}\right) \leq \Pr[\neg E_j] \leq p^{j-1} .$$

□

Lemma 3.10. *Let $\mathcal{A}^{H,\bullet}$ be given q_H quantum queries to H and q_S classical queries to $\bullet$. Then for every $i \in [q_S]$ and $j \in \mathbb{Z}_{\geq 0}$ we have*

$$SD\left(\mathcal{G}_i^{\bar{H},O_j^{\mathsf{B_S}}}, \mathcal{G}_i^{\bar{H},O_{j+1}^{\mathsf{B_S}}}\right) \leq p^{j-1} \cdot \left(\left(\frac{2(i-1)}{1-p} + 2(j-1) + \frac{q_H}{2}\right)\epsilon + \sqrt{q_H\epsilon}\right) .$$

Proof. Let E_j be the event that the jth iteration of O_j is reached. Conditioned on $\neg E_j$, the oracles behave identically, i.e. $O_j[\neg E_j] = O_{j+1}[\neg E_j]$. We thus have

$$\begin{aligned} SD\left(\mathcal{G}_i^{\bar{H},O_j^{\mathsf{B_S}}}, \mathcal{G}_i^{\bar{H},O_j^{\mathsf{B_P}}}\right) &= \Pr[E_j] \cdot SD\left(\mathcal{G}_i^{\bar{H},O_j^{\mathsf{B_S}}[E_j]}, \mathcal{G}_i^{\bar{H},O_j^{\mathsf{B_P}}[E_j]}\right) \\ &\leq p^{j-1} \cdot \left(\left(\frac{2(i-1)}{1-p} + 2(j-1) + \frac{q_H}{2}\right)\epsilon + \sqrt{q_H\epsilon}\right) , \end{aligned}$$

where the last inequality is via a direct application of our adaptive reprogramming lemma (Lemma 3.4). □

Corollary 3.11. *Let $\mathcal{A}^{H,\bullet}$ make at most q_H quantum queries to H and q_S classical queries to $\bullet$. Then*

$$SD\left(\mathcal{A}^{H,\mathsf{Sign}^H}, \mathcal{A}^{H,\mathsf{Prog}^{\bar{H}}}\right) \leq \frac{2q_S^2\epsilon}{(1-p)^2} + \frac{3q_S\sqrt{q_H\epsilon}}{2(1-p)} \leq \frac{3q_S\sqrt{q_H\epsilon}}{1-p} ,$$

where the last inequality holds as long as $q_H > 0$ and the right-hand-side is at most 1.

Proof. Combining Lemmas 3.9 and 3.10, we obtain

$$\begin{aligned} SD\left(\mathcal{G}_i^{\bar{H},\mathsf{Sign}}, \mathcal{G}_{i+1}^{\bar{H},\mathsf{Prog}}\right) &\leq SD\left(\mathcal{G}_i^{\bar{H},\mathcal{O}_{k+1}^{\mathsf{B_S}}}, \mathcal{G}_i^{\bar{H},\mathcal{O}_\infty}\right) + \sum_{j\in[k]} SD\left(\mathcal{G}_i^{\bar{H},\mathcal{O}_j^{\mathsf{B_S}}}, \mathcal{G}_i^{\bar{H},\mathcal{O}_{j+1}^{\mathsf{B_S}}}\right) \\ &\leq p^k + \sum_{j\in[k]} p^{j-1} \cdot \left(\left(\frac{2(i-1)}{1-p} + 2(j-1) + \frac{q_H}{2}\right)\epsilon + \sqrt{q_H\epsilon}\right) \\ &\leq p^k + \frac{2q_S\epsilon}{(1-p)^2} + \frac{q_H\epsilon}{2(1-p)} + \frac{\sqrt{q_H\epsilon}}{1-p} , \end{aligned}$$

where the last inequality is via $p \leq 1$ and $i \leq q_S$ and the p^k term vanishes as $k \to \infty$. Summing the above over $i \in [q_H]$, the proof is concluded. □

3.4.4 From Prog to Trans

For the purpose of showing closeness of $\mathcal{A}^{\mathsf{Prog},H}$ and $\mathcal{A}^{\mathsf{Trans},H}$, we introduce a second instantiation H' of the random oracle, which is set to be equal to H at the beginning, and we modify Prog to Prog' so as to also reprogram H', but only on the accepted transcript (see Fig. 3.7 middle). Looking ahead, we notice that this detour via Prog' and H' is not done in the ROM proof; there, we have a (more) direct argument to go from $\mathcal{A}^{\mathsf{Prog},H}$ to $\mathcal{A}^{\mathsf{Trans},H}$, very similar to the one going from $\mathcal{A}^{\mathsf{Sign},H}$ to $\mathcal{A}^{\mathsf{Prog},H}$. The reason we do it this way here is that we obtain a better bound than when trying to mimic the reasoning that is used in the ROM proof.

$\mathsf{Prog}(m)$:	$\mathsf{Prog}'(m)$:	$\mathsf{Trans}(m)$:
1: **repeat**	1: **repeat**	1: **repeat**
2: $\quad r \leftarrow \mathcal{D}(sk)$	2: $\quad r \leftarrow \mathcal{D}(sk)$	2: $\quad r \leftarrow \mathcal{D}(sk)$
3: $\quad H(r,m) := y \leftarrow \mathcal{Y}$	3: $\quad H(r,m) := y \leftarrow \mathcal{Y}$	3: $\quad y \leftarrow \mathcal{Y}$
4: $\quad z := f(r,z)$	4: $\quad z := f(r,z)$	4: $\quad z := f(r,z)$
5: **until** $z \neq \perp$	5: **until** $z \neq \perp$	5: **until** $z \neq \perp$
6:	6: $H'(r,m) := y$	6: $H(r,m) := y$
7: **return** (r,z)	7: **return** (r,z)	7: **return** (r,z)

Figure 3.7: The oracles Prog, Prog' and Trans.

Since the adversary $\mathcal{A}$ in an execution of $\mathcal{A}^{H,\mathsf{Prog}}$ has its random-oracle queries answered by H, and $\mathcal{A}$ has no access to H', we obviously have that $\mathcal{A}^{H,\mathsf{Prog}} = \mathcal{A}^{H,\mathsf{Prog}'}$. Similarly, $\mathcal{A}^{H',\mathsf{Prog}'} = \mathcal{A}^{H,\mathsf{Trans}}$. Thus, it remains to show closeness of $\mathcal{A}^{H,\mathsf{Prog}'}$ and $\mathcal{A}^{H',\mathsf{Prog}'}$. Towards this goal, we first settle the following properties of an execution of Prog'.

Proposition 3.12. *For an arbitrary but fixed message m_0, let (r,y,z) be the first non-$\perp$ transcript produced in an invocation of $\mathsf{Prog}'(m_0)$, and let S' be the set of r's sampled in the loop for which $z = \perp$. Then the following holds.*

- *The distribution of (S',r,y,z) is invariant to the choice of m_0.* (3.5)
- *S' is statistically independent of (r,y,z).* (3.6)
- *For every $r^\circ \in A$, $\Pr[r^\circ \in S'] \leq \frac{\epsilon}{1-p}$.* (3.7)

Proof. Let $t_i = (r_i, y_i, z_i)$ be the transcript sampled in the ith iteration of the loop. For the purpose of the analysis, we assume that t_i is sampled for *every* $i \in \mathbb{Z}_{>0}$, even if the loop stops before. Then, the t_i's are i.i.d. distributed, and S' equals $\{r_1, \ldots, r_{K-1}\}$, with K being minimal such that $z_K \neq \perp$ and $(r,y,z) = t_K$. As the sampling of (S',r,y,z) does not involve m_0 at all, (3.5) follows immediately.

For the analysis of (3.6), we consider the list $L := [t_1, \ldots, t_{K-1}]$; clearly showing independence of L and (r, y, z) implies independence of S' and (r, y, z). Further consider an arbitrary but fixed list $L^\circ = [t_1^\circ, \ldots, t_{k-1}^\circ]$ of transcripts $t_i^\circ = (r_i^\circ, y_i^\circ, z_i^\circ)$, and an arbitrary but fixed transcript $t^\circ = (r^\circ, y^\circ, z^\circ)$. With the goal to show that

$$\Pr\left[L = L^\circ \text{ and } (r, y, z) = t^\circ\right] = \Pr\left[L = L^\circ\right] \cdot \Pr\left[(r, y, z) = t^\circ\right] , \tag{3.8}$$

we may assume $z_1^\circ = \cdots = z_{k-1}^\circ = \bot$ and $z^\circ \neq \bot$, because otherwise both sides of Equation (3.8) vanish trivially. But then, by definition of L and (r, y, z),

$$\begin{aligned}
&\Pr\left[L = L^\circ \text{ and } (r, y, z) = t^\circ\right] = \Pr\left[\forall i < k : t_i = t_i^\circ \text{ and } t_k = t^\circ\right] \\
&= \Pr\left[\forall i < k : t_i = t_i^\circ \text{ and } t_k = t^\circ \text{ and } z_k \neq \bot\right] \\
&= \Pr\left[\begin{matrix} z_k \neq \bot \\ \forall i < k : t_i = t_i^\circ \end{matrix}\right] \cdot \Pr\left[t_k = t^\circ \,\middle|\, \begin{matrix} z_k \neq \bot \\ \forall i < k : t_i = t_i^\circ \end{matrix}\right] \\
&= \Pr\left[\begin{matrix} z_k \neq \bot \\ \forall i < k : t_i = t_i^\circ \end{matrix}\right] \cdot \Pr\left[t_k = t^\circ \middle| z_k \neq \bot\right] \\
&= \Pr\left[L = L^\circ\right] \cdot \Pr\left[t_k = t^\circ \middle| z_k \neq \bot\right] ,
\end{aligned}$$

where the fourth equality is due to independence between $(t_1, \ldots, t_{k-1})$ and t_k. Furthermore, summing up both sides of the above equality over all choices of L°, noting that $\Pr\left[t_k = t^\circ \middle| z_k \neq \bot\right]$ does not depend on k (since the t_i's are i.i.d.), we immediately get that $\Pr\left[t_k = t^\circ \middle| z_k \neq \bot\right] = \Pr\left[(r, y, z) = t^\circ\right]$, which shows (3.8) and thus (3.6).

Next, notice that $|L| \geq \ell$ implies $z_1 = \cdots = z_\ell = \bot$. Thus,

$$\begin{aligned}
&\Pr\left[a^\circ \in S'\right] \leq \sum_{\ell \geq 1} \Pr\left[r_\ell = r^\circ \text{ and } |L| \geq \ell\right] \\
&\leq \sum_{\ell \geq 1} \Pr\left[r_\ell = r^\circ \text{ and } z_1 = \cdots = z_{\ell-1} = \bot\right] \\
&= \sum_{\ell \geq 1} \Pr\left[r_\ell = r^\circ\right] \cdot \Pr\left[z_1 = \cdots = z_{\ell-1} = \bot\right] \leq \sum_{\ell \geq 1} p^{\ell-1}\epsilon \leq \frac{\epsilon}{1-p} ,
\end{aligned}$$

where the equality holds due to the independence between a_ℓ and $(z_1, \ldots, z_{\ell-1})$. This concludes (3.7). □

For this purpose, for every $0 \leq i \leq q_H$ we let $\mathcal{G}_i$ be the hybrid between $\mathcal{A}^{H,\mathsf{Prog}'}$ and $\mathcal{A}^{H',\mathsf{Prog}'}$ that has the first i queries to the random oracle answered by H', and the remaining ones by H. Obviously, $\mathcal{G}_0 = \mathcal{A}^{H,\mathsf{Prog}'}$, while $\mathcal{G}_{q_H} = \mathcal{A}^{H',\mathsf{Prog}'}$. Thus, considering an arbitrary but fixed $1 \leq i \leq q_H$ and setting

$\mathcal{G} := \mathcal{G}_{i-1}$ and $\mathcal{G}' := \mathcal{G}_i$, it is sufficient to show that $\mathcal{G}$ and $\mathcal{G}'$ are close. This is indeed the case:

Lemma 3.13. $SD(\mathcal{G}, \mathcal{G}') \leq 2\sqrt{\frac{q_S \epsilon}{1-p}}$.

Proof. Below, we refer to the ith query of $\mathcal{A}$ to the random oracle, i.e., the query on which $\mathcal{G}$ and $\mathcal{G}'$ differ, as the *crucial query*.

In the respective executions of $\mathcal{G}$ and $\mathcal{G}'$, we define S as the set of all the r's that Prog' sampled but for which the corresponding $z = \perp$, in all the invocations of Prog' before the crucial query. Thus, by construction, at the time of the crucial query, H and H' differ at most at the points in S. (They might agree on a point in S, if the freshly sampled value for H at this point equals the old value.)

For the sake of analysis, consider a binary projective measurement on the input query register for the crucial query, which measures whether or not the input (r, m) is such that $r \in S$. Let Γ be satisfied if $r \notin S$, and let $\tilde{\mathcal{G}}$ and $\tilde{\mathcal{G}}'$ be the two respective games obtained by performing this measurement. Since H and H' only differ at the places (r, m) where $r \in S$, conditioned on Γ, the two oracles behave identically, and thus do $\tilde{\mathcal{G}}$ and $\tilde{\mathcal{G}}'$. Furthermore, the probability $\Pr[\Gamma]$ is the same in both games.

Thus by a double application of the gentle measurement lemma [Wil11], we have

$$SD(\mathcal{G}, \mathcal{G}') \leq SD(\mathcal{G}, \tilde{\mathcal{G}}'[\Gamma]) + SD(\tilde{\mathcal{G}}'[\Gamma], \tilde{\mathcal{G}}[\Gamma]) + SD(\tilde{\mathcal{G}}[\Gamma], \mathcal{G}]) \leq 2\sqrt{\Pr[\neg\Gamma]}\,.$$

Hence, it remains to bound the probability $\Pr[\neg\Gamma]$. The intuition is that S collects those r's that Prog dismisses; thus, $\mathcal{A}$ does not get to see them, so it is hard for him to find an element in S, hence Γ is satisfied most likely. However, turning this intuition into a rigorous argument is not fully straightforward, since the set S, as a random variable, has a somewhat odd distribution.

Let Q be the random variable indicating the number of queries made to Prog prior to the crucial query; we have with certainty that $Q \leq q_S$.

A crucial observation that holds for both $\mathcal{G}, \mathcal{G}'$ is that, conditioned on $Q = q$ for an arbitrary but fixed q, the set S equals $S'_1 \cup \cdots \cup S'_q$ where every S'_j is the set S' that was produced in the jth query of Prog' as specified in Proposition 3.12. We note that, at the time the adversary $\mathcal{A}$ makes the crucial query, H has not been queried, and (r, y, z) in Proposition 3.12 is the only information that is dissipated to the adversary for every prior query to Prog'. It follows from (3.5) and (3.6) that every S'_j is independent from the view of adversary, and hence so is S.

Due to the independence, it suffices to bound $\Pr[r^\circ \in S | Q = q]$ for every $r^\circ \in \mathcal{R}$. Then it follows from the union bound and (3.7) that

$$\Pr[r^\circ \in S | Q = q] \leq \sum_{j \in [q]} \Pr[r^\circ \in S'_j] \leq \frac{q_S \epsilon}{1-p}\,.$$

Putting things together, the proof is concluded. □

Corollary 3.14. *Let $\mathcal{A}^{H,\bullet}$ make at most q_H quantum queries to H and at most q_S queries to $\bullet$. Then,*

$$SD\left(\mathcal{A}^{H,\mathsf{Prog}}, \mathcal{A}^{H,\mathsf{Trans}}\right) \leq 2q_H\sqrt{\frac{q_S\epsilon}{1-p}}\,.$$

3.4.5 Wrapping up the Proof of Theorem 3.5

Collecting the bounds in Corollaries 3.11 and 3.14, for a fixed choice of sk, we obtain

$$\begin{aligned}
&SD\left(\mathcal{A}^{H,\mathsf{Sign}}, \mathcal{A}^{H,\mathsf{Sim}}\right)\\
&\leq SD\left(\mathcal{A}^{H,\mathsf{Sign}}, \mathcal{A}^{H,\mathsf{Prog}}\right) + SD\left(\mathcal{A}^{H,\mathsf{Prog}}, \mathcal{A}^{H,\mathsf{Trans}}\right) + SD\left(\mathcal{A}^{H,\mathsf{Trans}}, \mathcal{A}^{H,\mathsf{Sim}}\right)\\
&\leq \frac{3q_S\sqrt{q_H\epsilon_{\mathsf{sk}}}}{1-p_{\mathsf{sk}}} + 2q_H\sqrt{\frac{q_S\epsilon_{\mathsf{sk}}}{1-p_{\mathsf{sk}}}} + q_S\zeta_{\mathsf{sk}}\,.
\end{aligned}$$

Plugging the above back to Eq. (3.4), we conclude the proof. □

3.4.6 Classical Bounds

Assuming now $\mathcal{A}$ makes only classical queries to H, we give a classical bound below. First, Lemma 3.10 can be replaced with

$$SD\left(\mathcal{G}_j^{\bar{H},O_j^{\mathsf{BS}}}, \mathcal{G}_i^{\bar{H},O_{j+1}^{\mathsf{BS}}}\right) \leq p^{j-1}\left(\frac{i-1}{1-p} + j - 1 + q_H\right)\epsilon\,,$$

leading to a tighter classical variant of Corollary 3.11 as follows

$$\begin{aligned}
SD\left(\mathcal{A}^{H,\mathsf{Sign}^H}, \mathcal{A}^{H,\mathsf{Prog}^{\bar{H}}}\right) &\leq SD\left(\mathcal{G}_i^{\bar{H},O_{k+1}^{\mathsf{BS}}}, \mathcal{G}_i^{\bar{H},O_\infty}\right) + \sum_{j\in[k]} SD\left(\mathcal{G}_i^{\bar{H},O_j^{\mathsf{BS}}}, \mathcal{G}_i^{\bar{H},O_{j+1}^{\mathsf{BS}}}\right)\\
&\leq p^k + \sum_{j\in[k]} p^{j-1}\left(\frac{i-1}{1-p} + j - 1 + q_H\right)\\
&\leq p^k + \left(\frac{q_S}{(1-p)^2} + \frac{q_H}{1-p}\right)\epsilon\,,
\end{aligned}$$

where the p^k term vanishes as $k \to \infty$ and the last inequality is again via $i \leq q_S$. Then, we replace Lemma 3.13 with

$$SD(\mathcal{G}, \mathcal{G}') = SD(\tilde{\mathcal{G}}', \tilde{\mathcal{G}}) \leq \Pr[\neg\Gamma]\cdot SD(\tilde{\mathcal{G}}'[\Gamma], \tilde{\mathcal{G}}[\Gamma]) \leq \Pr[\neg\Gamma] \leq \frac{q_S\epsilon}{1-p}\,,$$

where $\tilde{\mathcal{G}}$ and $\tilde{\mathcal{G}}'$ are as defined in the proof of Lemma 3.13, which gives us

$$SD\big(\mathcal{A}^{H,\mathsf{Prog}}, \mathcal{A}^{H,\mathsf{Trans}}\big) \leq \frac{q_H q_S \epsilon}{1-p} \ .$$

Collecting the bounds, for a fixed choice of sk, we obtain

$$\begin{aligned} &SD\big(\mathcal{A}^{H,\mathsf{Sign}}, \mathcal{A}^{H,\mathsf{Sim}}\big) \\ &\leq SD\left(\mathcal{A}^{H,\mathsf{Sign}}, \mathcal{A}^{H,\mathsf{Prog}}\right) + SD\left(\mathcal{A}^{H,\mathsf{Prog}}, \mathcal{A}^{H,\mathsf{Trans}}\right) + SD\left(\mathcal{A}^{H,\mathsf{Trans}}, \mathcal{A}^{H,\mathsf{Sim}}\right) \\ &\leq \left(\frac{q_S^2}{(1-p)^2} + \frac{2q_H q_S}{1-p}\right)\epsilon + q_S \zeta_{\mathsf{sk}} \ , \end{aligned}$$

Plugging into Eq. (3.4), we immediately obtain the following.

Theorem 3.15. *Let $\mathcal{S}$ be a generalized FSwA signature scheme. Then, for every* UF-CMA *attacker $\mathcal{A}^{H,\bullet}$ making at most Q_H and q_S classical queries to H and the signing oracle respectively, the* UF-NMA *attacker $\mathcal{B}$ (dependent on $\mathcal{A}$) as defined in Theorem 3.5 is such that for every fixed choice of key sk with $p_{\mathsf{sk}} < 1$, and for $q_H = Q_H + 1$ we have*

$$\mathbf{Adv}^{\mathsf{UF\text{-}CMA}}_{\mathcal{S}}(\mathcal{A}, \mathsf{sk}) \leq \mathbf{Adv}^{\mathsf{UF\text{-}NMA}}_{\mathcal{S}}(\mathcal{B}, \mathsf{sk}) + \left(\frac{q_S^2}{(1-p_{\mathsf{sk}})^2} + \frac{2q_H q_S}{1-p_{\mathsf{sk}}}\right)\epsilon_{\mathsf{sk}} + q_S \zeta_{\mathsf{sk}} \ .$$

Corollary 3.16. *Let $\mathcal{S}$ be a generalized FSwA signature scheme that is (ζ, T)-acHVZK. Furthermore, let Γ_ϵ and Γ_p be subsets of keys sk such that $p_{\mathsf{sk}} \leq \bar{p}$ for all $\mathsf{sk} \in \Gamma_p$ and $\Pr[\mathsf{sk} \in \Gamma_\epsilon]\mathbb{E}[\epsilon_{\mathsf{sk}} | \mathsf{sk} \in \Gamma_\epsilon] \leq \bar{\epsilon}$ for parameters $0 < \bar{\epsilon}, \bar{p} < 1$. Then for every* UF-CMA *attacker $\mathcal{A}$ making at most Q_H and q_S classical queries to H and the signing oracle respectively, the* UF-NMA *attacker $\mathcal{B}$ (dependent on $\mathcal{A}$) as defined in Theorem 3.5 is such that the following holds for $q_H := Q_H + 1$:*

$$\begin{aligned} \mathbf{Adv}^{\mathsf{UF\text{-}CMA}}_{\mathcal{S}}(\mathcal{A}) \leq \mathbf{Adv}^{\mathsf{UF\text{-}NMA}}_{\mathcal{S}}(\mathcal{B}) + \left(\frac{q_S^2}{(1-\bar{p})^2} + \frac{2q_H q_S}{1-\bar{p}}\right)\bar{\epsilon} + q_S \zeta \\ + \Pr[\mathsf{sk} \notin \Gamma_p] + \Pr[\mathsf{sk} \notin \Gamma_\epsilon] \ . \end{aligned}$$

3.5 Tighter Bounds in QROM

In this section we provide tighter bounds in QROM. We recycle the same reduction and the same proving strategy as in Section 3.4. The only difference is that we provide a more efficient hybrid sequence to show the closeness $\mathcal{A}^{H,\mathsf{Prog}} \approx \mathcal{A}^{H,\mathsf{Trans}}$, compared to Section 3.4.4.

3.5.1 The Statements

Theorem 3.17. *Let $\mathcal{S}$ be a generalized FSwA signature scheme. Then for every* UF-CMA *attacker $\mathcal{A}^{H,\bullet}$ making at most Q_H quantum queries to H and q_S classical queries to the signing oracle, the* UF-NMA *attacker $\mathcal{B}$ (dependent on $\mathcal{A}$) as defined in Theorem 3.5 is such that is such that for every fixed choice of key* sk *with $p_{\mathsf{sk}} < 1$, and for $q_H = Q_H + 1$ we have*

$$\mathbf{Adv}^{\mathsf{UF\text{-}CMA}}_{\mathcal{S}}(\mathcal{A}, \mathsf{sk}) \leq \mathbf{Adv}^{\mathsf{UF\text{-}NMA}}_{\mathcal{S}}(\mathcal{B}, \mathsf{sk}) + \frac{8q_S\sqrt{q_H\epsilon_{\mathsf{sk}}}}{1-p_{\mathsf{sk}}} + q_S\zeta_{\mathsf{sk}} \ .$$

Corollary 3.18. *Let $\mathcal{S}$ be a generalized FSwA signature scheme that is (ζ, T)-acHVZK. Furthermore, let Γ_ϵ and Γ_p be subsets of keys* sk *such that $p_{\mathsf{sk}} \leq \bar{p}$ for all $\mathsf{sk} \in \Gamma_p$ and $\Pr[\mathsf{sk} \in \Gamma_\epsilon]\mathbb{E}[\epsilon_{\mathsf{sk}} | \mathsf{sk} \in \Gamma_\epsilon] \leq \bar{\epsilon}$ for parameters $0 < \bar{\epsilon}, \bar{p} < 1$. Then for every* UF-CMA *attacker $\mathcal{A}$ making at most Q_H quantum queries to H and q_S classical queries to the signing oracle, the* UF-NMA *attacker $\mathcal{B}$ (dependent on $\mathcal{A}$) as defined in Theorem 3.5 is such that the following holds for $q_H := Q_H + 1$:*

$$\mathbf{Adv}^{\mathsf{UF\text{-}CMA}}_{\mathcal{S}}(\mathcal{A}) \leq \mathbf{Adv}^{\mathsf{UF\text{-}NMA}}_{\mathcal{S}}(\mathcal{B}) + \frac{8q_S\sqrt{q_H\bar{\epsilon}}}{1-\bar{p}} + q_S\zeta + \Pr[\mathsf{sk} \notin \Gamma_p] + \Pr[\mathsf{sk} \notin \Gamma_\epsilon] \ .$$

3.5.2 From Prog to Trans

We recap the oracles Loop, $\mathsf{B_P}$ and introduce a new oracle $\mathsf{B_T}$ in Fig. 3.8. Our strategy for proving closeness of $\mathcal{A}^{H,\mathsf{Prog}}$ and $\mathcal{A}^{H,\mathsf{Trans}}$ is to replace, query by query and iteration by iteration, the body $\mathsf{B}^{\bar{H}}_{\mathsf{P}}$ of the repeat loop of $\mathsf{Prog}^{\bar{H}} = \mathsf{Loop}^{\mathsf{B}^{\bar{H}}_{\mathsf{P}}}$ by the body $\mathsf{B}^{\bar{H}}_{\mathsf{T}}$ of the repeat loop of $\mathsf{Trans}^{\bar{H}} = \mathsf{Loop}^{\mathsf{B}^{\bar{H}}_{\mathsf{T}}}$.

$\mathsf{Loop}^{\bullet}(\mathsf{sk}, m)$:	$\mathsf{B}^{\bar{H}}_{\mathsf{P}}(\mathsf{sk}, m)$:	$\mathsf{B}^{\bar{H}}_{\mathsf{T}}(\mathsf{sk}, m)$:
1: **repeat**	1: $r \leftarrow \mathcal{D}$	1: $r \leftarrow \mathcal{D}$
2: $\quad out \leftarrow \bullet(\mathsf{sk}, m)$	2: $H(r, m) := y \leftarrow \mathcal{Y}$	2: $y \leftarrow \mathcal{Y}$
3: **until** $out \neq \perp$	3: $z \leftarrow f(r, y)$	3: $z \leftarrow f(r, y)$
4: **return** out	4: **if** $z \neq \perp$, $H(r, m) := y$	4: **if** $z \neq \perp$, $H(r, m) := y$
	5: **if** $z = \perp$ **return** $\perp$	5: **if** $z = \perp$ **return** $\perp$
	6: **else return** (r, z)	6: **else return** (r, z)

Figure 3.8: The repetition loop (left), and different instantiations of the body of the loop $(\mathsf{B_P}, \mathsf{B_T})$. The greyed out line 4 in $\mathsf{B_P}$ is irrelevant and can be ignored.

In order to capture the corresponding hybrid game and hybrid step, we introduce the following game, played by an oracle algorithm $\mathcal{C}^{\bar{H},\bullet}$ with the following features (see Fig. 3.9). During its run, $\mathcal{C}$ is allowed to make multiple

quantum read and *classical write* queries to H, and moreover one single query to an unspecified oracle that is to be instantiated by $\mathsf{B_P}$ or $\mathsf{B_T}$.

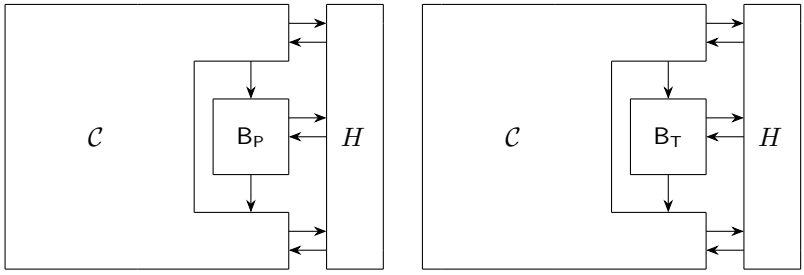


Figure 3.9: The oracle algorithm $\mathcal{C}$, which makes an fixed number of (at most) q_r quantum read queries to H, an expected number of (at most) q_w classical write queries to H, and one query to either $\mathsf{B_P}$ or $\mathsf{B_T}$.

For parameters q_r, q_w, we then define $\mathbf{Adv}(q_r, q_w)$ to be the maximal advantage of distinguishing the two games from Fig. 3.9, i.e.,

$$\mathbf{Adv}(q_r, q_w) := \sup_{\mathcal{C}} SD\left(\mathcal{C}^{\bar{H},\mathsf{B_P}}, \mathcal{C}^{\bar{H},\mathsf{B_T}}\right) , \tag{3.9}$$

takes the supremum over all $\mathcal{C}^{\bar{H},\blacklozenge}$ as above that make at most q_r quantum read queries to H *in the worst case* and at most q_w classical write queries *on average*, regardless of how the unspecified oracle ($\blacklozenge$) is instantiated.

The following allows us to control the closeness of $\mathcal{A}^{H,\mathsf{Prog}}$ and $\mathcal{A}^{H,\mathsf{Trans}}$ in terms of $\mathbf{Adv}(q_r, q_w)$.

Lemma 3.19. *Let $\mathcal{A}^{H,\bullet}$ be given q_H (read) queries to H and q_S signing queries. Then for p as in (3.1),*

$$SD\left(\mathcal{A}^{H,\mathsf{Prog}^{\bar{H}}}, \mathcal{A}^{H,\mathsf{Trans}^{\bar{H}}}\right) \leq \frac{q_S}{1-p} \cdot \mathbf{Adv}\left(q_H, \frac{q_S}{1-p}\right) .$$

At first glance, this is a straightforward hybrid argument, where we switch, one by one, the body of the jth iteration of the repeat loop in the ith signing query from $\mathsf{B_P}$ to $\mathsf{B_T}$; however, one needs to be careful since there is no fixed upper bound on the number of times the loop in Prog and Trans is repeated. However, via similar reasoning as in [BBD+23], we can exploit that it becomes less and less likely that the loop where we switch from $\mathsf{B_P}$ to $\mathsf{B_T}$ is reached, and so we can bound the distinguishing advantage by an infinite geometric series, which can be controlled. For this to work it is crucial that $\mathcal{A}$ cannot influence the number of loop repetitions (by the way of choosing m); whether the loop is repeated or not depends solely on the random choice of r.

For completeness, we work out the details in the formal proof below; it is somewhat tedious but altogether rather straightforward.

Proof. Let

$$\mathcal{G}_i^{\bar{H},\bullet} := \mathcal{A}^{H,[(\mathsf{Trans}^{\bar{H}})^{i-1},\bullet,(\mathsf{Prog}^{\bar{H}})^{q_S-i}]}$$

be a run of $\mathcal{A}$ such that the first $i-1$ signing queries are answered by Trans, the ith signing query is answered by an unspecified oracle (which will later be instantiate either by Trans or by Prog), and all remaining signing queries are answered by Prog. By construction, $\mathcal{G}_i^{\bar{H},\bullet}$ makes q_H quantum read queries to H, an expected number of at most $q_S/(1-p)$ classical write queries to H (the ones made by the runs of Trans and Prog), and one query to the unspecified oracle. Furthermore,

$$\mathcal{A}^{H,\mathsf{Prog}^{\bar{H}}} = \mathcal{G}_1^{\bar{H},\mathsf{Prog}^{\bar{H}}}\,, \quad \mathcal{G}_i^{\bar{H},\mathsf{Trans}} = \mathcal{G}_{i+1}^{\bar{H},\mathsf{Prog}} \quad \text{and} \quad \mathcal{G}_{q_S}^{\bar{H},\mathsf{Trans}^{\bar{H}}} = \mathcal{A}^{H,\mathsf{Trans}^{\bar{H}}}\,.$$

Our goal is to show the closeness of $\mathcal{G}_i^{\bar{H},\mathsf{Prog}^{\bar{H}}}$ and $\mathcal{G}_i^{\bar{H},\mathsf{Trans}^{\bar{H}}}$ for every $i \in \{1,\dots,q_S\}$, with error at most $\frac{1}{1-p}\mathbf{Adv}\big(q_H, \frac{q_S}{1-p}\big)$, which then implies the claim via

$$\mathcal{A}^{H,\mathsf{Prog}^{\bar{H}}} = \mathcal{G}_1^{\bar{H},\mathsf{Prog}^{\bar{H}}} \approx \mathcal{G}_1^{\bar{H},\mathsf{Trans}^{\bar{H}}} = \mathcal{G}_2^{\bar{H},\mathsf{Prog}^{\bar{H}}} \approx \cdots \approx \mathcal{G}_{q_S}^{\bar{H},\mathsf{Trans}^{\bar{H}}} = \mathcal{A}^{H,\mathsf{Trans}^{\bar{H}}}\,.$$

To show the claimed closeness, we do a similar hybrid argument as above, but now over the different iterations of the repeat loop in Trans and Prog. Concretely, we consider

$$\mathsf{Loop}_j^{\bar{H},\blacklozenge} := \mathsf{Loop}^{[(\mathsf{B}_\mathsf{T}^{\bar{H}})^{j-1},\blacklozenge,(\mathsf{B}_\mathsf{P}^{\bar{H}})^{\infty}]}$$

and observe that

$$\mathsf{Loop}_1^{\bar{H},\mathsf{B}_\mathsf{P}} = \mathsf{Prog}^{\bar{H}}\,,\ \ \mathsf{Loop}_j^{\bar{H},\mathsf{B}_\mathsf{T}} = \mathsf{Loop}_{j+1}^{\bar{H},\mathsf{B}_\mathsf{P}} \ \text{ and } \ \mathsf{Loop}_\infty^{\bar{H},\mathsf{B}_\mathsf{P}} = \mathsf{Loop}_\infty^{\bar{H},\mathsf{B}_\mathsf{T}} = \mathsf{Trans}^{\bar{H}}\,,$$

and so it remains to show the following closeness claims:

$$\begin{aligned}\mathcal{G}_i^{\bar{H},\mathsf{Prog}^{\bar{H}}} &= \mathcal{G}_i^{\bar{H},\mathsf{Loop}_1^{\bar{H},\mathsf{B}_\mathsf{P}}} \approx \mathcal{G}_i^{\bar{H},\mathsf{Loop}_1^{\bar{H},\mathsf{B}_\mathsf{T}}} = \mathcal{G}_i^{\bar{H},\mathsf{Loop}_2^{\bar{H},\mathsf{B}_\mathsf{P}}} \\ &\approx \cdots \approx \mathcal{G}_i^{\bar{H},\mathsf{Loop}_k^{\bar{H},\mathsf{B}_\mathsf{P}}} \approx \mathcal{G}_i^{\bar{H},\mathsf{Loop}_\infty^{\bar{H},\mathsf{B}_\mathsf{P}}} = \mathcal{G}_i^{\bar{H},\mathsf{Trans}^{\bar{H}}}\end{aligned} \tag{3.10}$$

for sufficiently large $k > 1$.

The last closeness claim is rather straightforward. Indeed, $\mathsf{Loop}_\infty^{\bar{H},\mathsf{B}_\mathsf{P}}$ and $\mathsf{Loop}_k^{\bar{H},\mathsf{B}_\mathsf{P}}$ behave (potentially) differently only if the repeat loop, which is at the core of the two algorithms, is repeated at least k times, which happens

only if all the k prior calls to B_T produce $\perp$.[7] Formally, we write E_k for this event that the loop is repeated at least k times, and we observe that it happens with probability $\Pr[E_k] = p^{k-1}$ only. Then $\mathsf{Loop}_k[\neg E_k] = \mathsf{Loop}_\infty[\neg E_k]$, and therefore, exploiting that $\mathcal{G}_i$ makes only one call to (whatever version of) Loop,

$$\begin{aligned} SD\left(\mathcal{G}_i^{\bar{H},\mathsf{Loop}_k^{\bar{H},\mathsf{B_P}}}, \mathcal{G}_i^{\bar{H},\mathsf{Loop}_\infty^{\bar{H},\mathsf{B_P}}}\right) &\le SD\left(\mathcal{G}_i^{\bar{H},\mathsf{Loop}_k^{\bar{H},\mathsf{B_P}}[E_k]}, \mathcal{G}_i^{\bar{H},\mathsf{Loop}_\infty^{\bar{H},\mathsf{B_P}}[E_k]}\right)\Pr[E_k] \\ &\le \Pr[E_k] \ \le p^{k-1}\,. \end{aligned}$$

It remains to show that $\mathcal{G}_i^{\bar{H},\mathsf{Loop}_j^{\bar{H},\mathsf{B_P}}} \approx \mathcal{G}_i^{\bar{H},\mathsf{Loop}_j^{\bar{H},\mathsf{B_T}}}$ for every j. Similar to above, we note and exploit that

$$\mathsf{Loop}_j^{\bar{H},\mathsf{B_P}}[\neg E_j] = \mathsf{Loop}_j^{\bar{H},\mathsf{B_T}}[\neg E_j]\,,$$

i.e., the two behave identically if the jth iteration is not reached. Thus,

$$\begin{aligned} SD\left(\mathcal{G}_i^{\bar{H},\mathsf{Loop}_j^{\bar{H},\mathsf{B_P}}}, \mathcal{G}_i^{\bar{H},\mathsf{Loop}_j^{\bar{H},\mathsf{B_T}}}\right) &\le SD\left(\mathcal{G}_i^{\bar{H},\mathsf{Loop}_j^{\bar{H},\mathsf{B_P}}[E_j]}, \mathcal{G}_i^{\bar{H},\mathsf{Loop}_j^{\bar{H},\mathsf{B_T}}[E_j]}\right)\Pr[E_j] \\ &\le SD\left(\mathcal{C}_{i,j}^{\bar{H},\mathsf{B_P}}, \mathcal{C}_{i,j}^{\bar{H},\mathsf{B_T}}\right)p^{j-1} \le \mathbf{Adv}\Big(q_H, \tfrac{q_S}{1-p}\Big)p^{j-1}\,, \end{aligned}$$

where the second inequality is obtained by letting $\mathcal{C}_{i,j}^{\bar{H},\blacklozenge}$ to be the oracle algorithm $\mathcal{G}_i^{\bar{H},\mathsf{Loop}_j^{\bar{H},\blacklozenge}[E_j]}$, which performs the run of $\mathsf{Loop}_j[E_j]$ (which is promised to reach the jth iteration) internally, but forwards the oracle query. Noting that $\mathcal{C}_{i,j}^{\bar{H},\blacklozenge}$ is as required, with at most q_H quantum read queries to H and an average of at most $\frac{q_S}{1-p}$ classical write queries, the final upper bound applies.

Adding up all the error terms in (3.10), we get that

$$\begin{aligned} SD\left(\mathcal{G}_i^{\bar{H},\mathsf{Prog}^{\bar{H}}}, \mathcal{G}_i^{\bar{H},\mathsf{Trans}^{\bar{H}}}\right) &\le \mathbf{Adv}\Big(q_H, \frac{q_S}{1-p}\Big)\sum_{j=1}^{k-1} p^{j-1} + p^{k-1} \\ &\longrightarrow \ \frac{1}{1-p}\mathbf{Adv}\Big(q_H, \frac{q_S}{1-p}\Big) \end{aligned}$$

for $k \to \infty$. This concludes the proof. □

The main technical challenge, and so the main innovation of this work, lies in establishing the following claim.

Proposition 3.20. *For any positive $q_r, q_w \in \mathbb{Z}$ and for* $\mathbf{Adv}$ *as specified in* (3.9)

$$\mathbf{Adv}(q_r, q_w) \le (5q_w + q_r)\epsilon + 2\sqrt{q_r\epsilon}\,.$$

[7] It is actually necessary to loop for $k+1$ iterations for the two to behave differently, but we do not need to be tight here.

We note that the only difference between B_P and B_T is whether H gets reprogrammed or not in case $z = \perp$ (in which case r remains unknown). This difference can only be detected when $\mathcal{C}$ makes a future query to H on input r; but due to the assumed high entropy in r, this is unlikely to happen. Turning this intuition into a proof when $\mathcal{C}$ can make *quantum* queries to H results in a hybrid argument over the q_r queries to H, which in turn results in a bound on the distinguishing advantage of the order $q_r\sqrt{\epsilon}$. Thus, the actual challenge lies in finding a hybrid argument that shows that the advantage actually scales as $\sqrt{q_r\epsilon}$ (plus neglectable terms).

Proof. For the sake of the analysis, we introduce the following aborting variants of B_P and B_T, defined in Fig. 3.10. The only different to the non-aborting variants is line 6., where aB_P and aB_T instruct to abort instead of returning (r, z) in case $z \neq \perp$. We stress that the abort command is a *global* abort, causing the ambient game ($\mathcal{C}^{\bar{H},\mathsf{aB}_\mathsf{P}}$ or $\mathcal{C}^{\bar{H},\mathsf{aB}_\mathsf{T}}$) to abort if $z \neq \perp$, instead of returning (r, z) to the ambient game then.

$\mathsf{aB}_\mathsf{P}^{\bar{H}}(m)$:	$\mathsf{aB}_\mathsf{T}(m)$:
1: $r \leftarrow \mathcal{D}$	1: $r \leftarrow \mathcal{D}$
2: $H(r, m) := y \leftarrow \mathcal{Y}$	2: $y \leftarrow \mathcal{Y}$
3: $z \leftarrow f(r, y)$	3: $z \leftarrow f(r, y)$
4: **if** $z \neq \perp$ **then** $H(r, m) := y$	4: **if** $z \neq \perp$ **then** $H(r, m) := y$
5: **if** $z = \perp$ **then return** $\perp$	5: **if** $z = \perp$ **then return** $\perp$
6: **else abort**	6: **else abort**

Figure 3.10: Aborting variants of B_P and B_T, which cause the ambient game to abort if $z \neq \perp$, instead of returning (r, z). Line 4. then becomes irrelevant also for aB_T.

Since $\mathsf{B}_\mathsf{P}^{\bar{H}}$ and $\mathsf{B}_\mathsf{T}^{\bar{H}}$ behave identically anyway if $z \neq \perp$ (both have reprogrammed $H(r, m)$ and return (r, m)), asking to abort in that a case does not affect the distinguishing advantage, i.e.,

$$\Pr\left[1 \leftarrow \mathcal{C}^{\bar{H},\mathsf{B}_\mathsf{P}^{\bar{H}}}\right] - \Pr\left[1 \leftarrow \mathcal{C}^{\bar{H},\mathsf{B}_\mathsf{T}^{\bar{H}}}\right] = \Pr\left[1 \leftarrow \mathcal{C}^{\bar{H},\mathsf{aB}_\mathsf{P}^{\bar{H}}}\right] - \Pr\left[1 \leftarrow \mathcal{C}^{\bar{H},\mathsf{aB}_\mathsf{T}}\right] .$$

In order to show that the right hand side is small, we proceed through the following sequence of hybrid games $\mathcal{G}_0$ to $\mathcal{G}_5$, given in Fig. 3.11. We refer to the $\mathcal{G}_i$'s as "games" but after all these are just algorithms $\mathcal{G}_i^{\bar{H}}$ with write access to H, and so the concepts from Sect. 2.2 readily apply.

We also note that $\mathcal{G}_0^{\bar{H}}$ is semantically equal to $\mathcal{C}^{\bar{H},\mathsf{aB}_\mathsf{P}^{\bar{H}}}$, i.e. $\mathcal{G}_0^{\bar{H}} = \mathcal{C}^{\bar{H},\mathsf{aB}_\mathsf{P}^{\bar{H}}}$; we merely have split $\mathcal{C}$ into two parts, which respectively captures $\mathcal{C}$'s behavior before and after the call to $\mathsf{aB}_\mathsf{P}^{\bar{H}}$, and we have spelled out $\mathsf{aB}_\mathsf{P}^{\bar{H}}$. Correspondingly for $\mathcal{G}_5^{\bar{H}}$ and $\mathcal{C}^{\bar{H},\mathsf{aB}_\mathsf{T}^{\bar{H}}}$.

$\mathcal{G}_0$:
1: $(m, \mathsf{st}) \leftarrow \mathcal{C}_0^{\bar{H}}$
2:
3: $r \leftarrow \mathcal{D}$
4: $H(r, m) := y \leftarrow \mathcal{Y}$
5: $z \leftarrow f(r, y)$
6: **if** $z \neq \perp$ **abort**
7: $b \leftarrow \mathcal{C}_1^{\bar{H}}(\mathsf{st}, \perp)$
8: **return** b

$\mathcal{G}_1$:
1: $(m, \mathsf{st}) \leftarrow \mathcal{C}_0^{\bar{H}}$
2:
3: $r \leftarrow \mathcal{D}$
4: $y := H(r, m)$
5: $z \leftarrow f(r, y)$
6: **if** $z \neq \perp$ **abort**
7: $b \leftarrow \mathcal{C}_1^{\bar{H}}(\mathsf{st}, \perp)$
8: **return** b

$\mathcal{G}_2$:
1: $(m, \mathsf{st}) \leftarrow \mathcal{C}_0^{\bar{H}}$
2: $b \leftarrow \mathcal{C}_1^{\bar{H}}(\mathsf{st}, \perp)$
3: $r \leftarrow \mathcal{D}$
4: $y := H(r, m)$
5: $z \leftarrow f(r, y)$
6: **if** $z \neq \perp$ **abort**
7:
8: **return** b

$\mathcal{G}_3$:
1: $(m, \mathsf{st}) \leftarrow \mathcal{C}_0^{\bar{H}}$
2: $b \leftarrow \mathcal{C}_1^{\bar{H}}(\mathsf{st}, \perp)$
3: $r \leftarrow \mathcal{D}$
4: $H(r, m) := y \leftarrow \mathcal{Y}$
5: $z \leftarrow f(r, y)$
6: **if** $z \neq \perp$ **abort**
7:
8: **return** b

$\mathcal{G}_4$:
1: $(m, \mathsf{st}) \leftarrow \mathcal{C}_0^{\bar{H}}$
2: $b \leftarrow \mathcal{C}_1^{\bar{H}}(\mathsf{st}, \perp)$
3: $r \leftarrow \mathcal{D}$
4: $y \leftarrow \mathcal{Y}$
5: $z \leftarrow f(r, y)$
6: **if** $z \neq \perp$ **abort**
7:
8: **return** b

$\mathcal{G}_5$:
1: $(m, \mathsf{st}) \leftarrow \mathcal{C}_0^{\bar{H}}$
2:
3: $r \leftarrow \mathcal{D}$
4: $y \leftarrow \mathcal{Y}$
5: $z \leftarrow f(r, y)$
6: **if** $z \neq \perp$ **abort**
7: $b \leftarrow \mathcal{C}_1^{\bar{H}}(\mathsf{st}, \perp)$
8: **return** b

Figure 3.11: The hybrid games.

Game hop $\mathcal{G}_0$ to $\mathcal{G}_1$. The game $\mathcal{G}_1$ is obtained from $\mathcal{G}_0$ by replacing the reprogramming step $H(r, m) := y \leftarrow \mathcal{Y}$ to the hash evaluation $y := H(r, m)$ in line 4. Therefore, recalling the bound q_r on the number of quantum read queries to H and the bound q_w on the expected number of (classical) write queries of $\mathcal{C}$, from directly applying our variant of the adaptive reprogramming lemma (Lemma 3.4) we obtain

$$\left| \Pr\left[1 \leftarrow \mathcal{G}_0\right] - \Pr\left[1 \leftarrow \mathcal{G}_1\right] \right| \leq \left(2q_w + \frac{q_r}{2}\right)\epsilon + \sqrt{q_r \epsilon}\,.$$

As a quick remark, considering the bigger context, we note that y is now computed as in the original signing oracle; thus, at first glance it seems that we are making a step back again, towards $\mathcal{A}^{H,\mathsf{Sign}}$ instead of $\mathcal{A}^{H,\mathsf{Trans}}$. However, it is a crucial step in this delicate sequence of hybrids.

Game hop $\mathcal{G}_1$ to $\mathcal{G}_2$. The game $\mathcal{G}_2$ is identical to $\mathcal{G}_1$ except that the run of $\mathsf{aB_P}$ (including the decision to abort) is delayed to the very end of the game. Conditioned on the event that the execution of $\mathcal{C}_1^{\bar{H}}(\mathsf{st}, \perp)$ does not reprogram H at the point (r, m), for the r sampled in step 3., the two games $\mathcal{G}_1$ and $\mathcal{G}_2$

behave identically. I.e., using our formalism,

$$\mathcal{G}_1\big[\mathcal{C}_1^{\bar{H}}(\mathsf{st},\perp) \text{ does not program } H(r,m)\big] = \mathcal{G}_2[\,\mathcal{C}_1^{\bar{H}}(\mathsf{st},\perp) \text{ does not program } H(r,m)\,]\,.$$

Due to the min-entropy requirement (3.2) on r, and due to the bound q_w on the expected number of write queries that $\mathcal{C}$ performs, the probability that $\mathcal{C}_1^{\bar{H}}(\mathsf{st},\perp)$ does reprogram $H(r,m)$ is at most $q_w\epsilon$ (and it is the same probability in both games). Therefore,

$$\big|\ \Pr\left[1 \leftarrow \mathcal{G}_1\right] - \Pr\left[1 \leftarrow \mathcal{G}_2\right]\ \big| \leq q_w\epsilon\,.$$

Game hop $\mathcal{G}_2$ to $\mathcal{G}_3$. The game $\mathcal{G}_3$ is defined from $\mathcal{G}_2$ by replacing the hash evaluation $y := H(r,m)$ in line 4 to reprogramming $H(r,m) := y \leftarrow \mathcal{Y}$. This is again a direct application of the adaptive reprogramming lemma, and so

$$\big|\ \Pr\left[1 \leftarrow \mathcal{G}_2\right] - \Pr\left[1 \leftarrow \mathcal{G}_3\right]\ \big| \leq \Big(2q_w + \frac{q_r}{2}\Big)\epsilon + \sqrt{q_r\epsilon}\,.$$

Game hop $\mathcal{G}_3$ to $\mathcal{G}_4$. The game $\mathcal{G}_4$ is obtained from $\mathcal{G}_3$ by dropping the reprogramming $H(r,m) := y$ in line 4. Since there are no further queries to H after that point, this change has no effect on the output b, and so

$$\Pr\left[1 \leftarrow \mathcal{G}_3\right] = \Pr\left[1 \leftarrow \mathcal{G}_4\right]\ .$$

Game hop $\mathcal{G}_4$ to $\mathcal{G}_5$. The game $\mathcal{G}_5$ is the same as $\mathcal{G}_4$, but the run of $\mathcal{C}_1^{\bar{H}}(\mathsf{st},\perp)$ is moved to the end again. This is just a syntactic change, which only affects *when* the abort decision is made, but does not affect the actual outcome of the game. Hence

$$\Pr\left[1 \leftarrow \mathcal{G}_4\right] = \Pr\left[1 \leftarrow \mathcal{G}_5\right]\ .$$

Collecting the upperbounds, the proof is concluded. □

3.5.3 Wrapping up the Proof of Theorem 3.17

For a fixed choice of sk, collecting the bounds in Lemma 3.19 and *Proposition* 3.20, we obtain

$$SD(\mathcal{A}^{H,\mathsf{Prog}}, \mathcal{A}^{H,\mathsf{Trans}}) \leq \frac{q_S}{1-p} \cdot \mathbf{Adv}\left(q_H, \frac{q_S}{1-p}\right)$$
$$\leq \left(\frac{5q_S^2}{(1-p)^2} + \frac{q_S q_H}{1-p}\right)\epsilon + \frac{2q_S}{1-p}\sqrt{q_H\epsilon} \leq \sqrt{\frac{6q_S^2 q_H}{(1-p)^2}} + \frac{2q_S}{1-p}\sqrt{q_H\epsilon} \leq \frac{5q_S}{1-p}\sqrt{q_H\epsilon}\,,$$

where the third inequality holds as long as $q_H > 0$ (which is satisfied because $q_H = Q_H + 1 > 0$) and the right-hand side is at most 1. Combined with

Corollary 3.11, we obtain

$$\begin{aligned}
&SD\left(\mathcal{A}^{H,\mathsf{Sign}}, \mathcal{A}^{H,\mathsf{Sim}}\right) \\
&\leq SD\left(\mathcal{A}^{H,\mathsf{Sign}}, \mathcal{A}^{H,\mathsf{Prog}}\right) + SD\left(\mathcal{A}^{H,\mathsf{Prog}}, \mathcal{A}^{H,\mathsf{Trans}}\right) + SD\left(\mathcal{A}^{H,\mathsf{Trans}}, \mathcal{A}^{H,\mathsf{Sim}}\right) \\
&\leq \frac{3q_S}{1-p_{\mathsf{sk}}}\sqrt{\frac{q_H\epsilon_{\mathsf{sk}}}{1-p_{\mathsf{sk}}}} + \frac{5q_S}{1-p_{\mathsf{sk}}}\sqrt{\frac{q_H\epsilon_{\mathsf{sk}}}{1-p_{\mathsf{sk}}}} + q_S\zeta_{\mathsf{sk}} \leq \frac{8q_S}{1-p_{\mathsf{sk}}}\sqrt{\frac{q_H\epsilon_{\mathsf{sk}}}{1-p_{\mathsf{sk}}}} + q_S\zeta_{\mathsf{sk}} \,.
\end{aligned}$$

Plugging the above back to Eq. (3.4), we conclude Theorem 3.17.

3.6 Concrete Analysis of Dilithium

In this section, we show, partly computer aided, a lower bound on the min-entropy, i.e., an upper bound on the guessing probability, of the first message $\mathsf{highBits}(\mathbf{Ay}, 2\gamma_2)$ in the sigma protocol underlying Dilithium for some relevant choices of the parameters.

We briefly recall relevant details of Dilithium here. Let $R_q \cong \mathbb{F}_q[X]/(X^n+1)$ be a cyclotomic ring over the finite field $\mathbb{F}_q$ of order q, where X^n+1 splits completely in $\mathbb{F}_q$, i.e. there exist pair-wise distinct $\alpha_1, \ldots, \alpha_n \in \mathbb{F}_q$ such that $X^n+1 = (X-\alpha_1)\cdots(X-\alpha_n)$. Let $k, \ell, \gamma_1, \gamma_2 \in \mathbb{Z}_{>0}$ be so that $k \geq \ell$, $\gamma_1 > \gamma_2$, and let $S_{\gamma_1-1} \subseteq R_q$ be of size $2\gamma_1 - 1$. In addition, let $\mathsf{highBits} : R_q \times \mathbb{Z}_{>0} \to R_q$ be a function so that each preimage of $\mathsf{highBits}(\cdot, 2\gamma_2)$ (restricting the second input to be $2\gamma_2$) is of size at most $2\gamma_2 + 1$.[8] What we are interested in, is to show that, with overwhelming probability over the choice of $\mathbf{A} \leftarrow R_q^{k\times\ell}$, the min-entropy of $\mathsf{highBits}(\mathbf{Ay}, 2\gamma_2)$ is large over the randomness of $\mathbf{y} \leftarrow S_{\gamma_1-1}^{\ell}$. The concrete parameters are specified in Fig. 3.12 below.

	n	ℓ	q	γ_1	γ_2
NIST2	256	4	8380417	2^{17}	$(q-1)/88$
NIST3	256	5	8380417	2^{19}	$(q-1)/32$
NIST5	256	7	8380417	2^{19}	$(q-1)/32$

Figure 3.12: Concrete parameters of Dilithium.

3.6.1 Controlling the Min-Entropy via the Rank of A

First, note that, for the top-most square $\mathbf{A}^{\square} \in R_q^{\ell\times\ell}$ of $\mathbf{A} \in R_q^{k\times\ell}$,

$$\mathsf{H}_\infty(\mathsf{highBits}(\mathbf{Ay}, 2\gamma_2)) \geq \mathsf{H}_\infty(\mathsf{highBits}(\mathbf{A}^{\square}\mathbf{y}, 2\gamma_2)) \geq \mathsf{H}_\infty(\mathbf{A}^{\square}\mathbf{y}) - n\ell\log(2\gamma_2+1)\,.$$

[8] We are going to slightly abuse the notation: when $\mathsf{highBits}(\cdot, 2\gamma_2)$ is applied to a tuple of elements in R_q, we take it as understood that it is applied componentwisely to the tuple.

Furthermore,

$$\mathsf{H}_\infty(\mathbf{A}^\square \mathbf{y}) \geq \mathsf{H}_\infty(\mathbf{y}) - (n\ell - \mathsf{rank}(\mathbf{A}^\square)) \log(q)\,,$$

where $\mathsf{rank}(\mathbf{A}^\square)$ is the rank of $\mathbf{A}^\square$ acting on R_q^ℓ as a $\mathbb{F}_q$-linear space. Finally, by the choice of $\mathbf{y}$, $\mathsf{H}_\infty(\mathbf{y}) \geq n\ell \log(2\gamma_1 - 1)$. Thus, altogether,

$$\mathsf{H}_\infty(\mathsf{highBits}(\mathbf{A}\mathbf{y}, 2\gamma_2)) \geq n\ell \log\left(\frac{2\gamma_1 - 1}{2\gamma_2 + 1}\right) - (n\ell - \mathsf{rank}(\mathbf{A}^\square)) \log(q)\,. \quad (3.11)$$

Therefore, it suffices to have good enough control over $\mathsf{rank}(\mathbf{A}^\square)$.

3.6.2 Numerically Controlling the Rank of A

The following is a direct consequence of the requirement that $X^n + 1$ splits completely in $\mathbb{F}_q$.

Lemma 3.21. *There is a $\mathbb{F}_q$-algebra isomorphism between*

$$\phi : R_q^{\ell\times\ell} \xrightarrow{\sim} \bigoplus_{1\leq i\leq n} \mathbb{F}_q^{\ell\times\ell}\,.$$

Furthermore, ϕ is $\mathbb{F}_q$-rank-preserving, i.e. for every $\mathbf{A}^\square$, with $\phi(\mathbf{A}^\square) = \bigoplus_i \mathbf{D}_i$, we have $\mathsf{rank}(\mathbf{A}^\square) = \mathsf{rank}(\phi(\mathbf{A}^\square)) = \sum_i \mathsf{rank}(\mathbf{D}_i)$.

From Lemma 3.21, we now know that the distribution of $\mathsf{rank}(\mathbf{A}^\square)$ equals the distribution of $\sum_i \mathsf{rank}(\mathbf{D}_i)$ for random and independent $\mathbf{D}_1, \ldots, \mathbf{D}_n \leftarrow \mathbb{F}_q^{\ell\times\ell}$.

The rank of a random matrix. Below, we thus consider a uniformly random $\mathbf{D} \leftarrow \mathbb{F}_q^{\ell\times\ell}$, and we work out the distribution of $\mathsf{rank}(\mathbf{D})$. For this purpose, let

$$\mathbf{D} = \begin{pmatrix}\mathbf{D}^1 & \ldots & \mathbf{D}^\ell\end{pmatrix}\,,$$

where each $\mathbf{D}^j \in \mathbb{F}_q^\ell$ is the jth column of $\mathbf{D}$. Define the *rank sequence* $r_1, \ldots, r_\ell$ given by

$$r_j := \mathsf{rank}\begin{pmatrix}\mathbf{D}^1 & \ldots & \mathbf{D}^j\end{pmatrix}\,.$$

With the convention that $r_0 := 0$, define their difference sequence $d_1, \ldots, d_\ell$ as $d_j := r_j - r_{j-1} \in \{0, 1\}$. In other words, d_j indicates whether the jth column increases the rank or not.

Lemma 3.22 below gives the distribution of the difference sequence $(d_1, \ldots, d_n)$ for a random $\mathbf{D}$.

Lemma 3.22. *The probability that a random matrix $\mathbf{D} \in \mathbb{F}_q^{\ell\times\ell}$ has a given*

difference sequence $(d_1, \ldots, d_\ell) \in \{0,1\}^\ell$ *is*

$$\pi_\ell(q, d_1, \ldots, d_\ell) := \prod_{1 \le j \le \ell} \left(d_j + (-1)^{d_j} q^{-(\ell - r_{j-1})} \right) ,$$

where $r_1, \ldots, r_\ell$ *is naturally defined as* $r_j = d_1 + \cdots + d_j$*, with* $r_0 = 0$*.*

Proof. For any $j \in \{1, \ldots, \ell\}$, conditioned on the columns $\mathbf{D}^1, \ldots, \mathbf{D}^{j-1}$, the matrix $\mathbf{D}$ has $d_j = 0$ if and only if the jth column $\mathbf{D}^j$ lies in $\mathsf{span}_{\mathbb{F}_q}\{\mathbf{D}^1, \ldots, \mathbf{D}^{j-1}\}$, which happens with probability

$$\Pr\left[\mathbf{D}^j \in \mathsf{span}_{\mathbb{F}_q}\{\mathbf{D}^1, \ldots, \mathbf{D}^{j-1}\}\right] = \frac{\left|\mathsf{span}_{\mathbb{F}_q}\{\mathbf{D}^1, \ldots, \mathbf{D}^{j-1}\}\right|}{q^\ell} = q^{-(\ell - r_{j-1})} ,$$

and it has $d_j = 1$ with complementary probability $1 - q^{-(\ell - r_{j-1})}$. The probability of a particular difference sequence $d_1, \ldots, d_\ell$ is then the product of the respective probabilities above, which matches the claim. □

Since $\mathsf{rank}(\mathbf{D}) = d_1 + \cdots + d_\ell$, we have

$$\Pr\left[\mathsf{rank}(\mathbf{D}) = r\right] = \sum_{d \in S_r^\ell} \pi_\ell(q, d) , \tag{3.12}$$

where $S_r^\ell := \{(d_1, \ldots, d_\ell) \in \{0,1\}^\ell \mid d_1 + \cdots + d_\ell = r\}$. Note that, by using Lemma 3.22, one can show that $\Pr\left[\mathsf{rank}(\mathbf{D}) = r\right] = p_r(1/q)$ for an (ℓ-dependent) integer polynomial p_r of degree at most ℓ^2. The equality (3.12) gives rise to an algorithm that computes the distribution of $\Pr\left[\mathsf{rank}(\mathbf{D}) = r\right]$ (as polynomials in $1/q$) in time $2^{O(\ell)}$. For $\ell = 5$ (which is in line with the NIST3 parameters of Dilithium) one obtains the polynomials given in Fig. 3.13.[9]

[9] For such a small choice of ℓ the exponential run time is no issue. As a matter of fact, this could still be worked out by hand.

$$
\begin{aligned}
p_0(1/q) &= q^{-25} \\
p_1(1/q) &= -q^{-25} - q^{-24} - q^{-23} - q^{-22} - q^{-21} + q^{-20} + q^{-19} + q^{-18} + q^{-17} + q^{-16} \\
p_2(1/q) &= q^{-24} + q^{-23} + 2q^{-22} + 2q^{-21} + q^{-20} - q^{-19} - 2q^{-18} - 4q^{-17} - 4q^{-16} \\
&\quad - 2q^{-15} - q^{-14} + q^{-13} + 2q^{-12} + 2q^{-11} + q^{-10} + q^{-9} \\
p_3(1/q) &= -q^{-22} - q^{-21} - 2q^{-20} - q^{-19} + 3q^{-17} + 4q^{-16} + 5q^{-15} + 3q^{-14} \\
&\quad - 3q^{-12} - 5q^{-11} - 4q^{-10} - 3q^{-9} + q^{-7} + 2q^{-6} + q^{-5} + q^{-4} \\
p_4(1/q) &= q^{-19} + q^{-18} - q^{-16} - 2q^{-15} - 3q^{-14} - 2q^{-13} + q^{-12} + 3q^{-11} + 4q^{-10} \\
&\quad + 3q^{-9} + q^{-8} - 2q^{-7} - 3q^{-6} - 2q^{-5} - q^{-4} + q^{-2} + q^{-1} \\
p_5(1/q) &= -q^{-15} + q^{-14} + q^{-13} - q^{-10} - q^{-9} - q^{-8} + q^{-7} + q^{-6} + q^{-5} \\
&\quad - q^{-2} - q^{-1} + 1 \,.
\end{aligned}
$$

Figure 3.13: The polynomials $p_r(1/q) = \Pr[\mathsf{rank}(\mathbf{D}) = r]$ for $\ell = 5$.

The rank of a random block-diagonal matrix. Towards controlling the distribution of $\mathsf{rank}(\mathbf{A}^{\square})$, we consider the generating function for the distribution of $\mathsf{rank}(\mathbf{D})$, given by

$$f(z) := \sum_{0 \le r \le \ell} \Pr[\mathsf{rank}(\mathbf{D}) = r] \cdot z^r = \sum_{0 \le r \le \ell} p_r(1/q) \cdot z^r \,. \tag{3.13}$$

Then, the nth power $f^n(z)$ of the above generating function generates the distribution of $\mathsf{rank}(\mathbf{A}^{\square}) = \mathsf{rank}(\mathbf{D}_1) + \cdots + \mathsf{rank}(\mathbf{D}_n)$, i.e.

$$f^n(z) = \sum_{0 \le r \le \ell n} \Pr\left[\mathsf{rank}(\mathbf{A}^{\square}) = r\right] \cdot z^r \,.$$

On input $f(z)$, as a polynomial in z with degree ℓ, with coefficient being polynomials in $1/q$ with degree (at most) ℓ^2, the nth power $f^n(z)$ can be computed in time polynomial in n and ℓ. $\Pr\left[\mathsf{rank}(\mathbf{A}^{\square}) = r\right]$ can then be obtained by reading out the coefficient of the degree-r term of $f^n(z)$, which is again an integer polynomial in $1/q$, and evaluating it (as a rational number) for the considered choice of $q \in \mathbb{Z}$.

For any $i \in \mathbb{Z}_{>0}$, we can then compute

$$\Pr\left[\mathsf{rank}(\mathbf{A}^{\square}) < n\ell - i\right] = \sum_{r < n\ell - i} \Pr\left[\mathsf{rank}(\mathbf{A}^{\square}) = r\right]$$

as a rational number $\frac{a}{b}$ with $a \le b \in \mathbb{Z}$, which we can then upper bound by writing $b = ad + e$ for $e \in \{0, \ldots, a-1\}$, and noting that

$$\Pr\left[\mathsf{rank}(\mathbf{A}^{\square}) < n\ell - i\right] = \frac{a}{b} = \frac{a}{ad+e} \le \frac{1}{d} \,.$$

Finally, counting the number of bits in the bit representation of d excluding the most significant bit gives us the largest $\delta \in \mathbb{Z}$ so that $2^\delta \leq d$, and thus $\Pr\left[\mathsf{rank}(\mathbf{A}^\square) < n\ell - i\right] \leq 2^{-\delta}$.

3.6.3 Plugging in the Numbers

For parameters as described in Fig. 3.12, we present the relevant quantities, obtained by following the above computation steps using Sage. In Fig. 3.14 below are the obtained upper bounds δ_i such that $\Pr[\mathsf{rank}(\mathbf{A}^\square) < n\ell - i] \leq \delta_i$ for selective choices of i's, together with the resulting bounds $\mathsf{H}_\infty(\mathsf{highBits}(\mathbf{Ay}, 2\gamma_2)) \geq \eta_i$ obtained via (3.11). For the NIST3 parameter, in particular, we see that except with probability at most 2^{-440} the matrix $\mathbf{A}^\square$ has corank at most 23, and then

$$\mathsf{H}_\infty(\mathsf{highBits}(\mathbf{Ay}, 2\gamma_2)) \geq 752\,,$$

which means that the guessing probability is at most

$$\mathsf{guess}\left(\mathsf{highBits}(\mathbf{Ay}, 2\gamma_2)\right) \leq 2^{-752}\,,$$

where the randomness is over the choice of $\mathbf{y}$.

i	0	1	5	6	8	12	23	33	44	63
$-\log \delta_i$	14	31	99	117	153	227	440	641	867	1268
$\eta_i^{(2)}$	471	448	356	333	287	195	0	0	0	0
$\eta_i^{(3)}$	1281	1258	1166	1143	1097	1005	752	522	269	0
$\eta_i^{(5)}$	1794	1771	1679	1656	1610	1518	1265	1035	782	345

Figure 3.14: $\Pr\left[\mathsf{H}_\infty(\mathsf{highBits}(\mathbf{Ay}, 2\gamma_2)) \geq \eta_i^{(\iota)}\right] \leq \delta_i$ for NISTι parameters, $\iota \in \{2, 3, 5\}$

One can obtain a slightly better bound by considering the *average* guessing probability over the choice of $\mathbf{A}$, averaged over the non-normalized distribution of $\mathbf{A}$ conditioned on $\mathbf{A}^\square$ having corank at most 12. Concretely, letting Γ_{23} be the event that $\mathbf{A}^\square$ has corank at most 23, where we know that $\Pr[\neg\Gamma_{23}] \leq 2^{-752}$, we obtain

$$\begin{aligned}
&\mathop{\mathbb{E}}_{A}\left[\,\mathsf{guess}\left(\mathsf{highBits}(\mathbf{Ay}, 2\gamma_2)\right) \;\middle|\; \Gamma_{23}\right] \cdot \Pr\left[\,\Gamma_{23}\,\right] \\
&\leq \sum_{0 \leq i \leq 23} \Pr[\mathsf{rank}(\mathbf{A}^\square) = n\ell - i \;\wedge\; \Gamma_{23}] \cdot 2^{-\eta_i^{(3)}} \leq \sum_{0 \leq i \leq 23} \delta_{i-1} \cdot 2^{-\eta_i^{(3)}} \leq 2^{-1172}\,,
\end{aligned}$$

with the convention that $\delta_{-1} = 1$. Similarly, we work out the (average-case) entropy bounds for all parameters. In Fig. 3.15, for a suitable choice of $a \in \mathbb{Z}_{\geq 0}$, and the corresponding event $\Gamma_\epsilon : \mathsf{rank}(\mathbf{A}^\square) \geq n\ell - a$, we provide an upperbound

for $\Pr[\neg\Gamma_\epsilon]$, and an upperbound on

$$\bar{\epsilon} := \mathop{\mathbb{E}}_{A}\left[\,\mathsf{guess}\left(\mathsf{highBits}(\mathbf{Ay}, 2\gamma_2)\right) \mid \Gamma_\epsilon\,\right] \cdot \Pr\left[\,\Gamma_\epsilon\,\right] \ .$$

Then, we compare the obtained classical security loss in Corollary 3.16

$$L = \left(\frac{q_S^2}{(1-\bar{p})^2} + \frac{2q_H q_S}{1-\bar{p}}\right)\bar{\epsilon} + q_S\zeta + \Pr\left[\,\neg\Gamma_p\,\right] + \Pr\left[\,\neg\Gamma_\epsilon\,\right] \ ,$$

with the corresponding quantum security loss in Corollary 3.7

$$L^* = \frac{8q_S\sqrt{q_H\bar{\epsilon}}}{1-\bar{p}} + q_S\zeta + \Pr\left[\,\neg\Gamma_p\,\right] + \Pr\left[\,\neg\Gamma_\epsilon\,\right] \ ,$$

taking $\zeta = 0$, Γ_p as always satisfied, and a heuristic choice for $\bar{p}$, as in [KLS18].

	$\bar{p}$	q_S	q_H	a	$\Pr[\neg\Gamma_\epsilon]$	$\bar{\epsilon}$	security loss
NIST2	$\leq \frac{49}{64}$	2^{64}	2^{128}	5	$\leq 2^{-99}$	$\leq 2^{-437}$	$L^* \leq 2^{-85}$
				12	$\leq 2^{-227}$	$\leq 2^{-404}$	$L \leq 2^{-209}$
			2^{64}	6	$\leq 2^{-117}$	$\leq 2^{-432}$	$L^* \leq 2^{-114}$
			1	8	$\leq 2^{-153}$	$\leq 2^{-422}$	$L^* \leq 2^{-141}$
NIST3	$\leq \frac{103}{128}$	2^{64}	2^{192}	23	$\leq 2^{-440}$	$\leq 2^{-1172}$	$L^* \leq 2^{-421}$
				44	$\leq 2^{-867}$	$\leq 2^{-1115}$	$L \leq 2^{-856}$
NIST5	$\leq \frac{759}{1024}$	2^{64}	2^{256}	33	$\leq 2^{-641}$	$\leq 2^{-1655}$	$L^* \leq 2^{-630}$
				63	$\leq 2^{-1268}$	$\leq 2^{-1591}$	$L \leq 2^{-1267}$

Figure 3.15: Concrete security loss of Dilithium, worked out via numeric calculation as described above.

3.6.4 Analytically Controlling ϵ_{sk}

Next, we give an analytic bound controlling $\mathsf{rank}(\mathbf{A}^\square)$ and hence ϵ_{sk} via (3.11). Crucially, by Lemma 8 in the full version of [BBD+23], over the random choice of the key pair $(\mathsf{sk}, \mathsf{pk})$, the distribution of $\mathsf{rank}(\mathbf{A}^\square)$ is identical to that of $\sum_{i\in[n]} \mathsf{rank}(\mathbf{D}_i)$ where $\mathbf{D}_1, \ldots, \mathbf{D}_n \leftarrow \mathbb{F}_q^{\ell\times\ell}$ are sampled uniformly and independently, which we bound below.

Theorem 3.23. *Let $\ell, n \in \mathbb{Z}_{>0}$ and q be a prime. Then for $\mathbf{D}_1, \ldots, \mathbf{D}_n \leftarrow \mathbb{F}_q^{\ell\times\ell}$*

and for every $a \in \mathbb{Z}_{\geq 0}$,

$$\Pr\left[\sum_{i\in[n]} \mathsf{rank}(\mathbf{D}_i) \leq n\ell - a\right] \leq e^{4/3}\,(n/q)^a \cdot (1-1/q)^{-n\ell}\,.$$

Proof. Let $\mathsf{corank}(\mathbf{D}_i) := \ell - \mathsf{rank}(\mathbf{D}_i)$ for each $i \in [n]$, we first work out the probability that $\mathsf{corank}(\mathbf{D}_i) = r$ for every $r \in \{0,\ldots,\ell\}$. Via one of the isomorphism theorems of linear spaces, once $\ker(\mathbf{D}_i) := \{v \in \mathbb{F}_q^\ell \mid \mathbf{D}_i v = 0\}$ and $\mathsf{Im}(\mathbf{D}_i) := \{\mathbf{D}_i v \mid v \in \mathbb{F}_q^\ell\}$ are fixed, the linear mapping $\mathbf{D}_i/\ker(\mathbf{D}_i) : \mathbb{F}_q^\ell/\ker(\mathbf{D}_i) \to \mathsf{Im}(\mathbf{D}_i)$ with $v + \ker(\mathbf{D}_i) \mapsto \mathbf{D}_i v$ uniquely determines $\mathbf{D}_i$. Note that for $\mathsf{corank}(\mathbf{D}_i) = r$, the spaces $\ker(\mathbf{D}_i)$ and $\mathsf{Im}(\mathbf{D}_i)$ can be (and only be) any sub-spaces of dimensions r and $n-r$ respectively, and $\mathbf{D}_i/\ker(\mathbf{D}_i)$ is bijective, and hence uniquely determined by an $(\ell-r)\times(\ell-r)$ invertible matrix (once the two spaces are fixed). Hence, we have the following chain of bijective correspondences:

$$\begin{gathered}
\left\{D \in \mathbb{F}_q^{\ell\times\ell} \;\middle|\; \mathsf{corank}(D) = r\right\} \\
\updownarrow \\
\left\{K \leq \mathbb{F}_q^\ell \;\middle|\; \dim(K) = r\right\} \times \left\{V \leq \mathbb{F}_q^\ell \;\middle|\; \dim(V) = \ell - r\right\} \times \mathsf{GL}(\ell-r, \mathbb{F}_q) \\
\updownarrow \\
\left\{K \leq \mathbb{F}_q^\ell \;\middle|\; \dim(K) = r\right\}^2 \times \mathsf{GL}(\ell-r, \mathbb{F}_q)\,,
\end{gathered}$$

where we denote the $\mathbb{F}_q$-subspace relation by $\leq$, and the second correspondence is via identifying the dual space $V^\perp := \{u \in \mathbb{F}_q^\ell \mid u^T v = 0\ \forall v \in V\}$ so that $\dim V^\perp + \dim V = \ell$ for every subspace $V \leq \mathbb{F}_q^\ell$. Working out the above numbers via counting, we obtain

$$\begin{aligned}
\Pr\left[\mathsf{corank}(\mathbf{D}_i) = r\right] &= \left|\left\{D \in \mathbb{F}_q^{\ell\times\ell} \;\middle|\; \mathsf{corank}(D) = r\right\}\right| \cdot q^{-\ell^2} \\
&= \left|\left\{K \leq \mathbb{F}_q^\ell \;\middle|\; \dim(V) = r\right\}\right|^2 \cdot \left|\,\mathsf{GL}(\ell-r, \mathbb{F}_q)\,\right| \cdot q^{-\ell^2} \\
&= \left(\frac{\prod_{i\in[r]}\left(q^\ell - q^{i-1}\right)}{\prod_{i\in[r]}\left(q^r - q^{i-1}\right)}\right)^2 \cdot \left(\prod_{i\in[\ell-r]}\left(q^{\ell-r} - q^{i-1}\right)\right) \cdot q^{-\ell^2} \\
&\leq q^{-r^2} \cdot \left(1 - \frac{1}{q}\right)^{-\ell}\,, \qquad (3.14)
\end{aligned}$$

where the last inequality is via pulling out the leading factors from the products, and simplifying the remaining terms.

What we are interested in is the event where $\sum_i \mathsf{rank}(\mathbf{D}_i) \leq n\ell - a$, which is equivalent to the event where $\sum_i \mathsf{corank}(\mathbf{D}_i) \geq a$, and can be bounded as

below, for every $t > 0$:

$$\begin{aligned}\Pr\left[\sum_{i\in[n]} \mathsf{corank}(\mathbf{D}_i) \geq a\right] &\leq e^{-at} \cdot \mathbb{E}\left[e^{\sum_i \mathsf{corank}(\mathbf{D}_i)\cdot t}\right] = e^{-at} \prod_{i\in[n]} \mathbb{E}\left[e^{\mathsf{corank}(\mathbf{D}_i)\cdot t}\right] \\ &\leq e^{-at} \cdot \left(\sum_{r\geq 0} e^{rt} \cdot q^{-r^2}\right)^n \cdot \left(1-\frac{1}{q}\right)^{-n\ell} \\ &\leq e^{-at} \cdot \exp\left(\sum_{r>0} n \cdot e^{rt} \cdot q^{-r^2}\right) \cdot \left(1-\frac{1}{q}\right)^{-n\ell}, \end{aligned} \tag{3.15}$$

where the first inequality is via Markov's bound, the first equality is via noticing that $\mathbf{D}_1, \ldots, \mathbf{D}_n$ are mutually independent, the second inequality is via (3.14), and the last inequality is via the fact that $1 + x \leq \exp(x)$ for every $x \in \mathbb{R}$. Plugging in $t = \ln(q) - \ln(n)$, we immediately obtain

$$\begin{aligned}(3.15) &\leq \left(\frac{n}{q}\right)^a \cdot \exp\left(\sum_{r>0} n^{-(r-1)} \cdot q^{-r(r-1)}\right) \cdot \left(1-\frac{1}{q}\right)^{-n\ell} \\ &\leq \left(\frac{n}{q}\right)^a \cdot \exp\left(\sum_{r\geq 0} n^{-r} \cdot q^{-2r}\right) \cdot \left(1-\frac{1}{q}\right)^{-n\ell} \\ &\leq \left(\frac{n}{q}\right)^a \cdot \exp\left(\frac{1}{1-1/(nq^2)}\right) \cdot \left(1-\frac{1}{q}\right)^{-n\ell} \\ &\leq \left(\frac{n}{q}\right)^a \cdot e^{4/3} \cdot \left(1-\frac{1}{q}\right)^{-n\ell}, \end{aligned}$$

where the last inequality is via the fact that $q \geq 2$. This concludes the proof. □

Combining the above and (3.11) with $\Gamma_\epsilon : \mathsf{corank}(A^{\square}) \leq a$, we get

$$\begin{aligned}\Pr[\Gamma_\epsilon] \cdot \mathbb{E}[\epsilon_{\mathsf{sk}} \mid \Gamma_\epsilon] &\leq \sum_{0\leq r\leq a} \Pr\left[\mathsf{corank}(\mathbf{A}^{\square}) = r\right] \cdot \mathbb{E}\left[\epsilon_{\mathsf{sk}} \;\middle|\; \mathsf{corank}(\mathbf{A}^{\square}) = r\right] \\ &\leq \sum_{0\leq r\leq a} e^{4/3}(n/q)^r \cdot (1-1/q)^{-n\ell} \cdot \left(\frac{2\gamma_2+1}{2\gamma_1-1}\right)^{n\ell} \cdot q^r \\ &\leq a \cdot e^{4/3} \cdot n^a \cdot \left(\frac{2\gamma_2+1}{2\gamma_1-1}\right)^{n\ell} \cdot (1-1/q)^{-n\ell}. \end{aligned}$$

Corollary 3.24. *Let Dilithium with relevant parameters $n, q, \ell, \gamma_1, \gamma_2$ be as*

described in [BBD+23], and in addition $q \geq n\ell$. Then for every $a \in \mathbb{Z}_{>0}$ there is an event Γ_ϵ of the key sk such that $\Pr[\neg\Gamma_\epsilon] \leq e^{7/3}(n/q)^{a+1}$, and

$$\Pr[\Gamma_\epsilon] \cdot \mathbb{E}[\epsilon_{\mathsf{sk}} \mid \Gamma_\epsilon] \leq a \cdot e^{7/3} \cdot n^a \cdot \left(\frac{2\gamma_2+1}{2\gamma_1-1}\right)^{n\ell} .$$

Combining the above bound with Corollary 3.18, and simplifying the bound under a suitable choice of a, yields Corollary 3.25, and the corresponding concrete bounds in Table 3.16 (taking Γ_p as always satisfied, and a heuristic choice of $\bar{p}$, as in [KLS18]).

	$\bar{p}$	q_S	q_H	a	security loss
NIST2	$\leq \frac{49}{64}$	2^{64}	2^{128}	5	$\leq 2^{-79}$
			2^{64}	7	$\leq 2^{-103}$
			1	9	$\leq 2^{-127}$
NIST3	$\leq \frac{103}{128}$	2^{64}	2^{192}	25	$\leq 2^{-371}$
NIST5	$\leq \frac{759}{1024}$	2^{64}	2^{256}	37	$\leq 2^{-547}$

Figure 3.16: Concrete security loss of Dilithium from Corollary 3.25.

Corollary 3.25. *Let Dilithium with relevant parameters $n, q, \ell, \gamma_1, \gamma_2$ be as described in this section above, and in addition $q \geq n\ell$. Let $0 < \bar{p} < 1$ and Γ_p be an event on $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}$ that implies $p_{\mathsf{sk}} \leq \bar{p}$. Then for every $\mathsf{UF\text{-}CMA}$ attacker $\mathcal{A}$ making at most Q_H quantum queries to H and q_S classical queries to the signing oracle, the $\mathsf{UF\text{-}NMA}$ attacker $\mathcal{B}$ (dependent on $\mathcal{A}$) as defined in Theorem 3.5 is such that for $q_H := Q_H + 1$ we have*

$$\mathbf{Adv}^{\mathsf{UF\text{-}CMA}}_{\mathsf{Dilithium}}(\mathcal{A}) \leq \mathbf{Adv}^{\mathsf{UF\text{-}NMA}}_{\mathsf{Dilithium}}(\mathcal{B}) + \frac{37 \cdot q_S}{1-\bar{p}} \sqrt{a \cdot n^a \cdot q_H \left(\frac{2\gamma_2+1}{2\gamma_1-1}\right)^{n\ell}} + \Pr[\neg\Gamma_p] ,$$

whenever

$$a := \left\lceil \frac{n\ell \cdot \log\left(\frac{2\gamma_1-1}{2\gamma_2+1}\right) - 2\log(q/n) - \log\left(q_S^2 q_H\right)}{2\log q - \log n} \right\rceil > 0 .$$

Chapter 4

BUFF Transformations

4.1 Introduction

Beyond the standard unforgeability security for digital signature schemes, additional security properties are sometimes desirable, like *exclusive ownership* [PS05], *message-bound signatures*, and *non-resignability* [JCCS19]. The NIST explicitly mentioned them as "*additional desirable security properties*" in their call for additional post-quantum signatures [NIST22]. As discussed in [CDF+21], there are real-life attacks in certain applications that exploit the lack of these additional security properties.

Non-resignability for example requires, informally, that it is hard for an attacker to maul a signature σ for a message m into a signature σ' under his own public-key, when he only gets to see the signature σ of m (under someone else's public key) and some auxiliary information on m, but not the message m itself. The relevance of non-resignability has been shown in [JCCS19], where the authors identified an attack against the "Dynamically Recreatable Key" protocol [KBJ+14] that indeed applies in case the deployed signature scheme does not satisfy non-resignability, uncovering a flaw in the protocol's original security analysis [ZBPB17].

On top of discussing these additional security properties and their relevance in applications, Cremers, Düzlü, Fiedler, Fischlin, and Janson [CDF+21] offer a generic transformation, the *BUFF transform* (which stands for **B**eyond **U**n**F**orgeability **F**eatures), that turns any signature scheme into a new signature scheme that is argued to then satisfy these additional security properties either in the random oracle model (ROM) or in the plain model under some non-standard assumptions on the hash function $\mathcal{F}$.[1]

[1]The original publication [CDF+21] has been revised in reaction to [DFHS24], one of the works on which this chapter is based; our discussion here is with respect to the original version of [CDF+21]; we discuss the revision [CDF+23] explicitly in Section 4.1.2.

The BUFF transform is very simple and causes little computational overhead. Indeed, instead of signing the original message, the BUFF-transformed signature scheme simply signs the hash $y = \mathcal{F}(m\|\mathsf{pk})$ of the message and the public-key, to get a signature σ, and then outputs the pair (σ, y) as signature; verification works in the obvious way. Motivated by the reference in the NIST call and the little overhead caused by the BUFF transform, several of the submissions to the NIST call for additional post-quantum signatures have the BUFF transform built in, or mention the possibility of applying the BUFF transform to the proposed scheme.[2]

There have also been some claims about (some of) these additional security properties being achieved by the three signature schemes that were selected by the NIST in 2022 to be standardized. Indeed, Cremers *et al.* [CDF+21] argue that Dilithium [LDK+20] uses the BUFF transform *implicitly*, allowing them to apply their main result regarding the BUFF transform. Falcon [PFH+20] does not achieve the beyond unforgeability features; however, the Falcon team announced that they will deploy the BUFF transform to achieve them [FHK+22]. Finally, Cremers *et al.* expect that SPHINCS [HBD+20] also achieves non-resignability, though only using some informal arguments.

4.1.1 Our Contribution

In this chapter, we show that the security of BUFF transformation, particularly the non-resignability (NR), is more subtle than originally believed. We then launch a quest to re-establishing the "right" definition of NR, via studying strength and achievability of various candidate definitions, respectively.

Impossibility of NR. First, we show in Section 4.3 that the NR security, as defined in [CDF+21], is essentially unachievable. When considering the plain model, we observe that for any signature scheme with the property that there is sufficient (computational) entropy in the message when given its signature (and the public key), there is a simple attack that entirely breaks NR of the signature scheme.[3]

Given that, by design, the BUFF transform satisfies this entropy requirement, it follows directly that the BUFF transform does *not* satisfy NR in the plain model, regardless of the hash function used (and regardless of the hash function being fixed or chosen from a family of possible hash functions). We stress that not only is there no proof for the NR of the BUFF transform in the plain model, but our aforementioned attack easily breaks it.

Moving to the random oracle model (ROM), somewhat surprising in the light of the positive results claimed in [CDF+21] on the BUFF transform in

[2]The following submissions explicitly refer to the BUFF transform: Squirrels [ENST23], Racoon [dEK+23], HAWK [BBD+24], PROV [GCF+23], Vox [PCF+23], and eMLE [LZ23].

[3]On the other hand, if the message can be efficiently computed from its signature, NR is also not satisfied, as already pointed out in [CDF+21].

the ROM, we show that, as a matter of fact, also in the ROM the BUFF transform does not satisfy NR. The matter is slightly more subtle here since prior works did not rigorously define NR in the ROM. What we show is that for the *natural extension* of NR to the ROM, our negative results from the plain model carry over, and thus, in particular, that the BUFF transform does not achieve (this natural notion of) NR in the ROM.

Given the positive claims in prior work, we discuss what is wrong with the reasoning in [CDF+21], where the BUFF transform is claimed to satisfy NR. Namely, the issue lies in the Φ-non-malleability property of the random oracle, incorrectly claimed in [BFS11] and used in [CDF+21]. More precisely, we show that Φ-non-malleability as stated in these works is unachievable.

We note that our generic attack on NR is embarrassingly simple in retrospect. It exploits that there is no restriction on the attacker's auxiliary information on the signed message m, subject to that it does not reveal m; this pretty much allows to embed the mauled signature σ' into the auxiliary information, making the attacker's job of finding σ' trivial. This attack has no (direct) real-world impact, since the auxiliary information is typically not adversarially chosen, but determined by the application. Instead, the point of our attack is to show that the formal definition put forward in [CDF+21] is too strong, and that prior positive results on achieving NR are incorrect. Thus, we need to go back to the drawing board: both the formal definition as well as achievability results need to be revised. This is what we do, to a certain degree, in the main part of this chapter, as discussed below.

A weaker variant: $\mathsf{NR}^{H,\perp}$. Facing the above strong negative result, we introduce in Section 4.4 a weaker variant of the original definition of the non-resignability property, which is still meaningful from an application perspective yet avoids the above generic attack, by requesting the auxiliary information to be computed without access to the random oracle.[4] This definition is thus still meaningful whenever in the considered application the computation of the auxiliary information does not depend on the random oracle that is used in the signing process for the considered signature scheme (which can typically be enforced via domain separation). We call this weaker variant $\mathsf{NR}^{H,\perp}$ (pronounced "NR-bot").

A natural question then is whether the BUFF transform satisfies $\mathsf{NR}^{H,\perp}$. Interestingly, this remains a non-trivial problem; as a matter of fact, depending on the precise formulation of the entropy requirement, which captures that the signed message m should remain hidden to the attacker, we show yet another negative result (see below).

On the positive side, we show that $\mathsf{NR}^{H,\perp}$ is satisfied in the ROM by a *salted version* of the BUFF transform, *if* the entropy requirement on the message m

[4] A more radical solution is to disallow any auxiliary information altogether, which in essence is done in [CDF+23]; see later for a more elaborate discussion of [CDF+23].

is *statistical* (rather than computational). The salted version of the BUFF transform includes a random salt in the hash and appends the salt to the signature. This is proven in the classical as well as in the quantum ROM (with different respective reduction losses), covering thus both classical and quantum attacks. Yet again on the negative side, by means of a counterexample in the form of a contrived signature scheme, we show that the above result on the salted version of the BUFF transform does not carry over in case the entropy requirement on the message m is *computational* (by means of the HILL entropy), as originally considered in [CDF+21]. This in particular also applies to the original (unsalted) BUFF transform.

Secret-key non-resignability. In Section 4.5, we introduce yet another variant of non-resignability, which we call *secret-key non-resignability* (sNR), and we show that the (original) BUFF transform satisfies sNR in the statistical setting, where the entropy condition holds statistically and adversaries may be computationally unbounded. Looking ahead at Section 4.8, this positive result also carries over to the computational setting, where the entropy condition holds computationally and adversaries have bounded computing power only.

In the statistical setting, sNR is strictly stronger than $\mathsf{NR}^{H,\perp}$; in the computational setting, the two notions are (probably) incomparable, yet sNR is strictly stronger than the notion considered in [CDF+23]. Given that, as shown in Section 4.4, the BUFF transform does not satisfy $\mathsf{NR}^{H,\perp}$ in the computational setting, our results appear to be the best we can hope for towards proving positive results on the non-resignability of the BUFF transform.

The approach in Section 4.5 recycle and adjust some of the arguments from the analyses of salted BUFF in Section 4.4. The crucial part of course is when Section 4.4 exploits the salt that originates from the salted BUFF transform, which we cannot do in Section 4.5, given that the original, unsalted variant is considered. Instead, we capture the crucial, missing piece in terms of a simple security game of the underlying hash function $\mathcal{F}$ called *Hide-and-Seek*, in the random oracle model, where, slightly more general than the usual case, we consider $\mathcal{F}$ that is given query access to a random oracle H. In essence, the game asks to find x when given $\mathcal{F}^H(x)$ and query-bounded access to H, where x may be chosen arbitrarily *dependent* on H subject to the condition that it is hard to guess when given access to H only, i.e., without being given $\mathcal{F}^H(x)$. We then reduce the sNR property of the BUFF transform to the hardness of winning Hide-and-Seek, when $\mathcal{F}$ is modelled as a random oracle H.

Despite its simplicity and harmless appearance, this game turns out to be surprisingly tricky to analyze. Thus, the technical core of Section 4.5 is in analyzing Hide-and-Seek of the random oracle, and showing that it is hard to win, both in the classical and in the quantum setting.

Fine-grained attack on BUFF. A natural question that is still open from Section 4.5 though, is whether the BUFF transform satisfies sNR in a more fine-grained use of the random oracle model, when considering a real-life iterative-hash-function design (such as Merkle-Damgård [Mer79, Mer90, Dam90] or Sponge [BDPA07]), where merely the round function is modelled via an idealized function. One might expect that the results where $\mathcal{F}$ is modelled as a random oracle would carry over, and that it is mainly a question of extending the proof — however, as we will see in the following, this is not the case.

We describe in Section 4.6 a simple attack on sNR of the BUFF transform when instantiated with *any* iterative hash function, where the round function may have access to an idealized function. This covers both Merkle-Damgård (in the random oracle model) and Sponge-based hash functions (in the random permutation model), and thus the design principles behind SHA-2, SHA-3, and SHAKE.

Again, the devil lies in the auxiliary information, which here can be abused to communicate an intermediate hash value to the adversary, making its task of resigning the unknown message a very easy one. Indeed, the adversary can then finalize the computation of the hash $\mathcal{F}(m\|\mathsf{pk}')$ even when it has uncertainty in the first blocks of the message m, and then simply sign the hash using its secret key sk', resulting in a resigning of the (partially) unknown message m. We note that this attack also applies to other notions of non-resignability where the adversary receives auxiliary information about the message.

We note that our negative result does not contradict the fact that Merkle-Damgård and Sponge are known to be *indifferentiable* from a random oracle, and thus can securely replace a random oracle (in certain cases), since the latter only holds for single-stage games, while non-resignability is a two-stage game. In that light, it is also not surprising that our attack shows some resemblance to the counter example provided in [RSS11].

Our attack is theoretical in nature, as similar to the attack in Section 4.3, it also exploits an artificial choice of the message and auxiliary information, which would be very unlikely to actually occur in a real-world protocol. However, more realistic attacks exploiting the iterative structure of the hash function might exist, highlighting the importance of provable security of the non-resignability property in this model.

Regaining sNR via the Sandwich BUFF. Towards preventing the above attack, and with the hope to re-establish sNR for the BUFF transform in the considered setting, we propose in Section 4.7 a simple modification to the transform, where instead of hashing $m\|\mathsf{pk}$ for signing, the hash of $m\|\mathsf{pk}\|m$ is computed and signed. Due to this sandwich structure of the hash input, we call this variant of the BUFF transform Sandwich BUFF, or sBUFF for short.

The main technical challenge in Section 4.7 is to show that this modification not only mitigates the attack, but also allows us to prove sNR for the Merkle-

Damgård hash function, when the round function is modelled using a random oracle.[5] To this end, we follow the similar strategy as in Section 4.5, show how an adversary against the sNR can be used to break the Hide-and-Seek game, which we then show is hard to win for the Merkle-Damgård hash function.

Achieving sNR in the computational setting Finally, we show in Section 4.8 that all positive results regarding sNR carries over to the computationally setting, where attackers are computationally bounded, and where the underlying entropy requirement is computational. This includes the sNR property of BUFF when $\mathcal{F}$ is modelled as a random oracle, and the sNR property of Sandwich BUFF in the fine-grained idealization setting.

4.1.2 Related Work

We already mentioned [CDF+21] and [JCCS19], upon which our work builds up. In reaction to our negative results in [DFHS24], on which Section 4.3 is based, the authors of [CDF+21] have updated their work; we briefly discuss this update [CDF+23] here.

In order to avoid our negative results (cf. Theorem 4.9), which exploit that the auxiliary information can be misused to embed a mauled signature, the authors modified the definition of non-resignability to require the auxiliary information to be *computationally independent* of the message (see [CDF+23, Fig. 4] for the non-resignability game and [CDF+23, Definition 4.3] for the actual definition). This is equivalent to not allowing any auxiliary information at all, and thus weaker than the variant of non-resignability we consider. Indeed, in the security reduction it is argued that, due to the computational independence, one can drop the auxiliary information altogether, and so reduce the non-resignability of the original BUFF transform to a variant of Φ-non-malleability with no auxiliary information (see [CDF+23, Fig. 1, right]).

Interestingly though, in the updated version [CDF+23], the authors have not adjusted their reasoning for their claim on the random oracle satisfying (the now weaker notion of) Φ-non-malleability, which is the place where the original flaw was hiding. They still argue via the very same informal reasoning as in the original version (see the quote in our Section 4.3.3). Although it is tempting to believe that this argumentation is sufficient, it is actually not.

Indeed, by means of a simple counter example we show in Appendix A.1 that *no* hash function (or hash function family), including the random oracle, satisfies the considered (weaker) notion of Φ-non-malleability. Thus, their claim on the BUFF transform satisfying the version of non-resignability considered in [CDF+23], under the assumption that the hash function satisfies the considered Φ-non-malleability notion, is vacuous.

[5] In [FHK25], one of the works on which this chapter is based, we also analyzed the security of Sponge, but for simplicity it is omitted here.

4.2 Preliminaries

4.2.1 (HILL) Entropy

The HILL entropy is a computational variant of min-entropy. First, we recall that for two random variables x and y, the computational distance

$$\delta_s(x, y) := \max_C \mid \Pr[C(x) = 1] - \Pr[C(y) = 1] \mid$$

is the maximum distinguishing advantage over all circuits C of size s. Then, the HILL entropy is defined as below.

Definition 4.1. *For a pair of random variables (x, z), the* conditional HILL entropy *(with parameters δ and s) is defined as*

$$^{\delta,s}\mathsf{HILL}_\infty(x \mid z) := \max_y \mathsf{H}_\infty(y \mid z)\,,$$

where the max *is over all random variables y such that $\delta_s\big((x, z), (y, z)\big) \leq \delta$.*

Remark 4.2. *When switching to asymptotic notation, for a family of pairs of random variables $\{(x_\lambda, z_\lambda)\}_{\lambda\in\mathbb{N}}$, a bound $\mathsf{HILL}_\infty(x_\lambda \mid z_\lambda) \geq k(\lambda)$ then naturally means that for every λ there exists Y_λ such that $\mathsf{H}_\infty(y_\lambda \mid z_\lambda) \geq k(\lambda)$, and for every polynomial $s(\lambda)$ there is a negligible $\delta(\lambda)$ such that $\delta_{s(\lambda)}\big((x_\lambda, z_\lambda), (y_\lambda, z_\lambda)\big) \leq \delta(\lambda)$. In this case, we tend to omit the security parameter λ and simply write (x, y) and $\mathsf{HILL}_\infty(x \mid z) \geq k$.*

In the plain model, the notion of min-entropy and the above HILL entropy captures unpredictability of random variables. Throughout this chapter, though, it is also relevant to consider the similar notion of unpredictability in the ROM. However, the random oracle itself becomes a source of randomness. Thus, by choosing the message to be the hash of a fixed string, it has high min-entropy but is still easy to guess, by just making one query to the random oracle. In the statistical setting, this can be dealt with by considering the min-entropy and additionally conditioning on the (function table of the) random oracle. On the other hand, conditioning on the exponentially large function table of the random oracle is problematic when considering the HILL entropy, which is defined computationally.

Below, we consider a natural way to overcome this issue by introducing a notion of HILL entropy in the ROM. In spirit, a random variable x, which may be correlated with the random oracle, has high HILL entropy, if it is indistinguishable from a random variable that has high statistical entropy, and where indistinguishability must hold for efficient oracle algorithms that may query the random oracle. Formally, for two random variables x, y that may depend on the random oracle H, we define the computational distance relative

to H as

$$\delta^H_{s,q}(x,y) := \max_C \left| \Pr\left[1 \leftarrow C^H(x)\right] - \Pr\left[1 \leftarrow C^H(y)\right] \right| ,$$

where the maximum is taken over all circuits C of size s and additionally given at most q queries to H. The definition of the (conditional) HILL entropy HILL^H_∞ relative to H is then in line with Definition 4.1.

Definition 4.3. *For a pair of random variables (x, z), possibly dependent on the random oracle H, the conditional HILL entropy (with parameters δ, s and q) relative to the random oracle is defined as*

$$^{\delta,s,q}\mathsf{HILL}^H_\infty(x \mid z) := \max_y \mathsf{H}_\infty(y \mid z, H) ,$$

where the max *is over all random variables y with $\delta^H_{s,q}((x,z),(y,z)) \leq \delta$.*

Note that we can also consider a variant where the indistinguishability is captured via *quantum* circuits, but for the sake of simplicity, here we consider only the classical variant of HILL entropy. Furthermore, similarly to Remark 4.2, we may also use an asymptotic notation and omit the parameters.

4.2.2 Non-Resignability and Φ-Non-Malleability

For a signature scheme $\mathcal{S} = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vrfy})$, *non-resignability* is defined via game NR, given in Fig. 4.1, which involves an *adversary* $\mathcal{A}_{\mathsf{NR}} = (\mathcal{D}, \mathcal{A})$ and a (possibly randomized) auxiliary function aux. We say that $\mathcal{A}_{\mathsf{NR}}$ is PPT if $\mathcal{D}$ and $\mathcal{A}$ are. In spirit, the goal of the adversary is to turn a signature σ for an unknown message m into a signature for the same message, but under a different, adversarially chosen, public key. We write

$$\mathbf{Adv}^{\mathsf{NR}}_{\mathcal{S}}(\lambda, \mathcal{A}_{\mathsf{NR}}, \mathsf{aux}) = \Pr[1 \leftarrow \mathsf{NR}_{\mathcal{S}}]$$

for the probability that the $\mathsf{NR}_{\mathcal{S}}$ game outputs 1 when instantiated with signature scheme $\mathcal{S}$ and with adversary $\mathcal{A}$ and auxiliary function aux. Similarly, for variants of NR and for other games that we will consider. For simplicity, we will often leave the security parameter λ implicit. We stress that beyond the input given to $\mathcal{A}$, no additional information is communicated from $\mathcal{D}$ to $\mathcal{A}$.

$\mathsf{NR}_{\mathcal{S}}$:
1: $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}$
2: $m \leftarrow \mathcal{D}(\mathsf{pk})$
3: $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m)$
4: $(\sigma', \mathsf{pk}') \leftarrow \mathcal{A}(\mathsf{pk}, \sigma, \mathsf{aux}(m, \mathsf{pk}))$
5: $v := \mathsf{Vrfy}(\mathsf{pk}', m, \sigma')$
6: **return** $(v = 1 \wedge \mathsf{pk}' \neq \mathsf{pk})$

Figure 4.1: Security game $\mathsf{NR}_{\mathcal{S}}$ (in the plain model) with an explicit hint function aux and a signature scheme $\mathcal{S} = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vrfy})$. The original definition in [CDF+21] had both m and $\mathsf{aux}(m, \mathsf{pk})$ produced by $\mathcal{D}(\mathsf{pk})$. This change in the definition only makes our negative result stronger (since it is a restriction on how h is produced), and will be convenient later on when trying to restore positive results.

As pointed out in [CDF+21], an adversary $\mathcal{A}_{\mathsf{NR}} = (\mathcal{D}, \mathcal{A})$ can easily win NR if $\mathcal{A}$ can compute m; indeed, it can then just sign m under a public-key pk' for which it knows the secret key. Thus, for this to be a potentially hard game, we need to enforce an entropy requirement on m. In this work, we consider two variants, by requiring the *statistical* entropy $\mathsf{H}_{\infty}(m \mid \mathsf{pk}, a)$ or the *computational* entropy $\mathsf{HILL}_{\infty}(m \mid \mathsf{pk}, a)$ to be lower bounded, where m, pk and a are chosen as in NR.

Following [CDF+21] (subject to this minor change in the game NR mentioned in Fig. 4.1), non-resignability is defined as follows.

Definition 4.4. *In the plain model, a signature scheme $\mathcal{S} = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vrfy})$ is called non-resignable if for every PPT adversary $\mathcal{A}_{\mathsf{NR}} = (\mathcal{D}, \mathcal{A})$ and every PPT function aux that satisfy the computational entropy condition*

$$\mathsf{HILL}_{\infty}\limits_{\substack{(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KGen} \\ m \leftarrow \mathcal{D}(\mathsf{pk})}} \big(m \,\big|\, \mathsf{pk}, \mathsf{aux}(m, \mathsf{pk})\big) \geq \omega(\log \lambda) \tag{4.1}$$

it holds that $\mathbf{Adv}^{\mathsf{NR}}_{\mathcal{S}}(\mathcal{A}_{\mathsf{NR}}, \mathsf{aux}) \leq \mathsf{negl}(\lambda)$.

A related notion, which is used in [CDF+21] towards proving non-resignability of the BUFF transform (see Section 4.2.3), is Φ-non-malleability, first introduced in [BCFW09], for a hash function $\mathcal{F}$. The definition is via game $\Phi\text{-}\mathsf{NM}_{\mathcal{F}}$, given in Fig. 4.2, and, as for non-resignability, we need to require a certain amount of statistical entropy $\mathsf{H}_{\infty}(x \mid \mathsf{aux}(x))$ or computational entropy $\mathsf{HILL}_{\infty}(x \mid \mathsf{hk}, \mathsf{aux}(x))$, for the game to be non-trivial.

Remark 4.5. *Strictly speaking, [BCFW09, CDF+21] consider key-ed hash functions, while we restrict our attention to key-less hash functions for simplicity.*

$\Phi\text{-}\mathsf{NM}_{\mathcal{F}}$:
1: $x \leftarrow \mathcal{D}$
2: $y := \mathcal{F}(x)$
3: $(y', \phi) \leftarrow \mathcal{A}(y, \mathsf{aux}(x))$
4: **return** $[\, \phi \in \Phi \wedge \mathcal{F}(\phi(x)) = y' \wedge \phi(x) \neq x \,]$

Figure 4.2: Security game $\Phi\text{-}\mathsf{NM}_{\mathcal{F}}$ with explicit hint function aux and a hash function $\mathcal{F}$.

Following [BFS11, CDF+21], for a family Φ of functions, Φ-non-malleability is defined as follows. Looking ahead, of particular interest is the case where x consists of two parts, conveniently written as $x = (\mathsf{pk}, m)$, and Φ consists of shifts of pk but leaves m untouched.

Definition 4.6. *A hash function $\mathcal{F}$ is called Φ-non-malleable if for any PPT adversary $\mathcal{A}_{\mathsf{NR}} = (\mathcal{D}, \mathcal{A})$ that satisfies the computational entropy condition*

$$\underset{x \leftarrow \mathcal{D}}{\mathsf{HILL}^{\infty}_{\infty}}\big(x \,\big|\, \mathsf{aux}(x)\big) \geq \omega(\log \lambda) \tag{4.2}$$

it holds that $\mathbf{Adv}^{\Phi\text{-}\mathsf{NM}}_{\mathcal{F}}(\mathcal{A}) \leq \mathsf{negl}(\lambda)$.

4.2.3 BUFF Transform

The BUFF transform [CDF+21] is a generic transform for signature schemes to achieve additional security properties beyond standard unforgeability. The transformation comes in two variants — BUFF and BUFF-lite. Throughout this work, we only consider the former, stronger variant[6] which is displayed in Fig. 4.3.

KGen:	$\mathsf{Sign}(\mathsf{sk}, m)$:	$\mathsf{Vrfy}(\mathsf{pk}, m, \sigma)$:
1: $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}_{\mathcal{S}}$	1: $y := \mathcal{F}(m, \mathsf{pk})$	1: $(\sigma_{\mathcal{S}}, y) := \sigma$
2: **return** $(\mathsf{sk}, \mathsf{pk})$	2: $\sigma_{\mathcal{S}} \leftarrow \mathsf{Sign}_{\mathcal{S}}(\mathsf{sk}, y)$	2: **return** $\mathsf{Vrfy}(\mathsf{pk}, y, \sigma_{\mathcal{S}}) \wedge$
	3: **return** $\sigma := (\sigma_{\mathcal{S}}, y)$	3: $y = \mathcal{F}(m, \mathsf{pk})$

Figure 4.3: The signature scheme $\mathsf{BUFF}[\mathcal{S}, \mathcal{F}] = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vrfy})$, obtained from applying the BUFF transform to $\mathcal{S} = (\mathsf{KGen}_{\mathcal{S}}, \mathsf{Sign}_{\mathcal{S}}, \mathsf{Vrfy}_{\mathcal{S}})$ with a hash function $\mathcal{F}$.

[6] The weaker one, BUFF-lite, does not achieve non-resignability, which is the focus of our work.

Remark 4.7. *A typical hash function takes as input a bit string, in which case we take it as understood that the hash* $\mathcal{F}(m, \mathsf{pk})$ *in BUFF is defined in terms of a unique representation of* (m, pk)*. Throughout the chapter, whenever this becomes relevant (like, in Section 4.6), we assume for simplicity that the public-key length* $|\,\mathsf{pk}\,|$ *is fixed and the hash is evaluated after concatenating its input, i.e.* $\mathcal{F}(m, \mathsf{pk}) := \mathcal{F}(m \| \mathsf{pk})$*.*

The idea of the BUFF transform is to sign the hash of the message and the public key, and to append this hash to the signature. This "binds" the public key to the signature, which ensures that no other keys can be generated that Ver such a signature, thus ensuring what is known as *exclusive ownership*. The idea behind *non-resignability* is as follows. In order to turn a signature into a new signature for the same message but under a different public key pk', the adversary needs to produce the hash value $y' := \mathcal{F}(m, \mathsf{pk}')$ for the modified public key pk' and the unknown message m, which should be hard by the Φ-non-malleability of the hash function. Indeed, [CDF+21] states the following result (where we omit the claims regarding further security properties that are not relevant to our work).

Claim 4.8 ([CDF+21, Theorem 5.5])**.** *Let* $\mathcal{S}$ *be an EUFCMA-secure signature scheme. Then the application of the BUFF transformation produces an EUFCMA-secure signature scheme* $\mathsf{BUFF}[\mathcal{S}, \mathcal{F}]$ *that additionally provides [...]* NR *if* $\mathcal{F}$ *is* Φ*-non-malleable where* $\Phi = \{\phi_{\mathsf{pk}'} \mid \mathsf{pk} \in \mathcal{K}\}$ *and* $\phi_{\mathsf{pk}'}(\mathsf{pk}, m) = (\mathsf{pk}', m)$*.*

In combination with the claim on the random oracle being Φ-non-malleable for this choice Φ from [BFS11], the authors of [CDF+21] then conclude non-resignability of the BUFF transform in the ROM.

4.3 On the Impossibility of Non-Resignability

In this section, we show strong negative results on the non-resignability property in general, and on the BUFF transform in particular.

First, we consider the plain model, where we show, by means of a simple attack, that non-resignability is not satisfied when applied to a signature scheme with the property that the message has high (computational) entropy when given its signature (and the public key).[7]

Since the BUFF transform, when applied to any signature scheme, satisfies the considered entropy requirement (assuming the hash function to be compressing), it follows directly that the BUFF transform does *not* satisfy non-resignability in the plain model, regardless of the hash function used. We

[7] In the other extreme, if the message can be efficiently computed from its signature, non-resignability is also not satisfied, as already pointed out in [CDF+21].

stress that not only is there no proof for the non-resignability of the BUFF transform in the plain model, but there is also an attack that breaks it.

These negative results from the plain model carry over to the random oracle model (ROM) when considering the *natural extension* of the non-resignability property to the ROM (prior works did not rigorously specify the property in the ROM). Thus, also in the ROM the BUFF transform does not (necessarily) satisfy non-resignability, invalidating the positive results claimed in [CDF+21] in that respect. In essence, the claim on the random oracle being Φ-non-malleable, made in [CDF+21, BFS11], is false.

4.3.1 Non-Resignability and BUFF Transform in the Plain Model

It is clear that the $\mathsf{NR}_{\mathcal{S}}$ game (Fig. 4.1) is easy to win if the message m can be efficiently computed from its signature σ. In the following theorem, we show that if, on the other hand, the signature scheme is such that the message m has high computational entropy given its signature (and the public key), then another attack applies.[8]

Theorem 4.9. *Let $\mathcal{S}$ be a signature scheme such that for a random message $m' \leftarrow \mathcal{M}$ and keys $(\mathsf{sk}', \mathsf{pk}') \leftarrow \mathsf{KGen}(1^\lambda)$ we have $\mathsf{HILL}_\infty(m' \mid \mathsf{pk}', \mathsf{Sign}(\mathsf{sk}', m)) \geq \omega(\log \lambda)$. Then there exists a PPT adversary $\mathcal{A}_{\mathsf{NR}} = (\mathcal{D}, \mathcal{A})$ and a PPT function aux such that the computational entropy condition (4.1) is satisfied, yet*

$$\mathbf{Adv}^{\mathsf{NR}}_{\mathcal{S}}(\mathcal{A}_{\mathsf{NR}}, \mathsf{aux}) \geq 1 - \mathsf{negl}(\lambda)\,.$$

The attack is surprisingly simple. Instead of burdening $\mathcal{A}$ with finding σ, which, intuitively, is hard since $\mathcal{A}$ does not know m, we simply let aux compute σ and hand it over to $\mathcal{A}$ as auxiliary information h. The entropy condition on the signature scheme then ensures that this is an eligible attack. The proof below spells out the details.

Proof. We construct the adversary $\mathcal{A}_{\mathsf{NR}} = (\mathcal{D}, \mathcal{A})$ and function aux as shown in Fig. 4.4. In the first stage, $\mathcal{D}$ samples a message uniformly at random and outputs it. The function aux, which receives m as an input, generates a new key pair $(\mathsf{sk}', \mathsf{pk}') \leftarrow \mathsf{KGen}(1^\lambda)$, computes $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}', m)$, and outputs the hint $h := (\sigma, \mathsf{pk}')$. In the second stage, $\mathcal{A}$ receives as input the public key pk, the signature $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m)$, and the hint $h = (\sigma, \mathsf{pk}')$, and it outputs (σ, pk').

By the completeness property, we have $\Pr[\mathsf{Vrfy}(\mathsf{pk}', m, \sigma) = 1] \geq 1 - \mathsf{negl}(\lambda)$. We further have $\Pr[\mathsf{pk}' \neq \mathsf{pk}] \geq 1 - \mathsf{negl}(\lambda)$ due to the high min-entropy of key generation.

[8] This leaves open only a very small, artificial gap for signature schemes that may potentially satisfy non-resignability: the message must be hard to compute from its signature while having low conditional HILL entropy.

It remains to argue that $\mathcal{A}_{\mathsf{NR}}$ satisfies the entropy condition (4.1). It holds that

$$\begin{aligned}\mathsf{HILL}_{\infty}\big(m \,\big|\, \mathsf{pk}, \mathsf{aux}(m, \mathsf{pk})\big) &= \mathsf{HILL}_{\infty}\big(m \,\big|\, \mathsf{aux}(m, \mathsf{pk})\big) \\ &= \mathsf{HILL}_{\infty}\big(m \,\big|\, \mathsf{pk}', \mathsf{Sign}(\mathsf{sk}', m)\big) \geq \omega(\log \lambda)\,,\end{aligned}$$

where the first equality holds by the independence of pk and $(m, \mathsf{aux}(m, \mathsf{pk}))$, the second equality holds by the construction of aux, and the last inequality holds from the entropy requirement. Taking all of this together, we get that $\mathcal{A}_{\mathsf{NR}}$ is a valid adversary playing $\mathsf{NR}_{\mathcal{S}}$ such that

$$\mathbf{Adv}^{\mathsf{NR}}_{\mathcal{S}}(\mathcal{A}_{\mathsf{NR}}, \mathsf{aux}) \geq 1 - \mathsf{negl}(\lambda)\,.$$

This concludes the proof. □

$\mathcal{D}(\mathsf{pk})$:	$\mathcal{A}(\mathsf{pk}, \sigma, h)$:	$\mathsf{aux}(m, \mathsf{pk})$:
1: $m \leftarrow \mathcal{M}$	1: $(\sigma, \mathsf{pk}') := h$	1: $(\mathsf{sk}', \mathsf{pk}') \leftarrow \mathsf{KGen}$
2: **return** m	2: **return** (σ, pk')	2: $\sigma \leftarrow \mathcal{S}.\mathsf{Sign}(\mathsf{sk}', m)$
		3: **return** (σ, pk')

Figure 4.4: Adversary $\mathcal{A}_{\mathsf{NR}} = (\mathcal{D}, \mathcal{A})$ and function aux used in the proof of Theorem 4.9.

Having established Theorem 4.9, it then follows as an immediate corollary that the BUFF-transform does not achieve non-resignability, no matter what hash function is used, as long as it is compressing, so that there is entropy in the message m when given $\mathcal{F}_{\mathsf{hk}}(\mathsf{pk}, m)$ (here, the entropy is even statistical).

Corollary 4.10. *Let $\mathcal{S}$ be a signature scheme, and let $\mathsf{BUFF}[\mathcal{S}, \mathcal{F}]$ be the signature scheme obtained via the BUFF transform obtained by using a (keyed) hash function $\mathcal{F}$ that compresses by at least the size of the public key plus $\omega(\log \lambda)$ bits. Then there exists a PPT adversary $\mathcal{A}_{\mathsf{NR}} = (\mathcal{D}, \mathcal{A})$ and a PPT function aux such that the entropy condition (4.1) is satisfied, yet*

$$\mathbf{Adv}^{\mathsf{NR}}_{\mathsf{BUFF}[\mathcal{S},\mathcal{F}]}(\mathcal{A}_{\mathsf{NR}}, \mathsf{aux}) \geq 1 - \mathsf{negl}(\lambda)\,.$$

Clearly, a non-compressing hash function avoids this particular attack; however, it is unclear if security would be restored (in particular in the light of Footnote 8). Also, from a practical point of view, hash functions used in the BUFFtransform will be compressing.

4.3.2 Non-Resignability and the BUFF Transform in the ROM

When considering the random oracle model, things become somewhat subtle. First, we note that no definition of non-resignability *in the ROM* has been explicitly provided in the previous literature; the definitions given in [CDF+21] are in the plain model. When switching to the ROM, one needs to specify who is given access to the random oracle. Clearly, considering a signature scheme "in the ROM", we give KGen, Sign, and Vrfy oracle access to the random oracle H.[9] Also, by default, the attacker is given oracle access to the random oracle. Thus, looking at Fig. 4.1, this means we certainly want to give $\mathcal{D}$ and $\mathcal{A}$ oracle access to the random oracle. But, say, what about the function aux? Given that in the original definition in [CDF+21], the auxiliary information h is actually computed by $\mathcal{D}$ (and our re-writing in terms of a function aux is for later convenience), it is natural to then also allow the function aux to have oracle access the random oracle. We make this explicit in Fig. 4.5, where we give the resulting security game for non-resignability *in the ROM*.

However, there is another subtle matter in the definition of non-resignability when switching to the ROM. Namely, the entropy condition (4.1), as well as its unconditional counterpart $\mathsf{H}_\infty(m \mid \mathsf{pk}, \mathsf{aux}^H(m, \mathsf{pk})) \geq \omega(\log \lambda)$, are not sufficient anymore for the definition to be meaningful. Indeed, $\mathcal{D}^H$ could simply choose m to be the hash of 0. In the ROM, this will be of high entropy and independent of pk; yet, $\mathcal{A}^H$ can easily recover it (as the hash of 0), and then honestly Sign it using his secret key. For this reason, in the definition of non-resignability in the ROM, we change the entropy requirement on the message to hold when additionally conditioning on the random oracle, i.e., we require

$$\mathsf{H}_\infty\limits_{\substack{(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KGen}^H(1^\lambda) \\ m \leftarrow \mathcal{D}^H(\mathsf{pk})}} \left(m \,\middle|\, \mathsf{pk}, \mathsf{aux}^H(m, \mathsf{pk}), H\right) \geq \omega(\log \lambda)\,. \tag{4.3}$$

We stress that we consider the *statistical* variant of the entropy condition here; with H an exponentially large function table, switching to the computational HILL variant will bring up further issues, which we want to avoid — for now (though we will look into this issue in Section 4.4.4). Furthermore, having this more stringent requirement on the attacker only makes our negative result stronger.

The following theorem shows that, considering the above natural definition of non-resignability in the ROM, the impossibility result from the plain model (Theorem 4.9) carries over pretty much in the obvious way.

Theorem 4.11. *Let H be a random oracle and $\mathcal{S}^H = (\mathsf{KGen}^H, \mathsf{Sign}^H, \mathsf{Vrfy}^H)$ be a signature scheme given query access to H such that for message $m' \leftarrow \mathcal{M}$ and key-pair $(\mathsf{sk}', \mathsf{pk}') \leftarrow \mathsf{KGen}^H$ we have $\mathsf{H}_\infty(m' \mid \mathsf{pk}', \mathsf{Sign}^H(\mathsf{sk}', m), H) \geq$*

[9] We make this explicit by writing KGen^H etc.

$\mathsf{NR}_{\mathcal{S}}^{H}$:
1: $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}^{H}(1^{\lambda})$
2: $m \leftarrow \mathcal{D}^{H}(\mathsf{pk})$
3: $\sigma \leftarrow \mathsf{Sign}^{H}(\mathsf{sk}, m)$
4: $(\sigma, \mathsf{pk}') \leftarrow \mathcal{A}^{H}\left(\mathsf{pk}, \sigma, \mathsf{aux}^{H}(m, \mathsf{pk})\right)$
5: $v := \mathsf{Vrfy}^{H}(\mathsf{pk}', m, \sigma)$
6: **return** $(v = 1 \wedge \mathsf{pk}' \neq \mathsf{pk})$

Figure 4.5: Security game $\mathsf{NR}_{\mathcal{S}}^{H}$ for the signature $\mathcal{S}^{H} = (\mathsf{KGen}^{H}, \mathsf{Sign}^{H}, \mathsf{Vrfy}^{H})$ in the random oracle model. In this variant, both the adversary and the function aux are granted access to the random oracle H.

$\omega(\log \lambda)$. *Then there exists a PPT adversary* $\mathcal{A}_{\mathsf{NR}}^{H} = (\mathcal{D}^{H}, \mathcal{A}^{H})$ *and a PPT function* aux^{H} *with access to* H *such that the entropy condition* (4.3) *is satisfied, yet*

$$\mathbf{Adv}_{\mathcal{S}}^{\mathsf{NR}^{H}}(\mathcal{A}_{\mathsf{NR}}, \mathsf{aux}) \geq 1 - \mathsf{negl}(\lambda)\,.$$

The proof follows essentially from the proof of Theorem 4.9 with some obvious adjustments regarding the random oracle.

Proof. For the signature scheme $\mathcal{S}^{H} = (\mathsf{KGen}^{H}, \mathsf{Sign}^{H}, \mathsf{Vrfy}^{H})$, we give adversaries $\mathcal{D}^{H}$ and $\mathcal{A}^{H}$ and the auxiliary function aux^{H} that wins the game $\mathsf{NR}_{\mathcal{S}}^{H}$ with overwhelming probability, in Fig. 4.6. In the first stage, $\mathcal{D}^{H}$ chooses a random message m that it will output. The auxiliary function, which receives the message m as input, first generates a new key pair $(\mathsf{sk}', \mathsf{pk}') \leftarrow \mathsf{KGen}^{H}(1^{\lambda})$, computes $\sigma' \leftarrow \mathsf{Sign}^{H}(\mathsf{sk}', y')$, and outputs $h := (\mathsf{pk}', \sigma')$. In the second state, $\mathcal{A}^{H}$ gets the public key pk, the signature σ (of message m using secret key sk), and the hint h (consisting of pk' and σ') as input and simply outputs $h = (\mathsf{pk}', \sigma')$. It is easy to see that $\mathsf{NR}_{\mathcal{S}}^{H}$ outputs 1 with overwhelming probability. This is because, by the correctness of $\mathcal{S}$, we have that σ' is a valid signature of m under pk' with overwhelming probability, and by the high min-entropy of a public key conditioned on H, we have $\mathsf{pk}' \neq \mathsf{pk}$ with overwhelming probability.

The remaining is to argue that $\mathcal{A}_{\mathsf{NR}}$ satisfies the entropy condition (4.3). It holds that

$$\begin{aligned}\mathsf{H}_{\infty}(m \mid \mathsf{pk}, \mathsf{aux}^{H}(m, \mathsf{pk}), H) &= \mathsf{H}_{\infty}(m \mid \mathsf{aux}^{H}(m, \mathsf{pk}), H) \\ &= \mathsf{H}_{\infty}(m \mid \mathsf{pk}', \mathsf{Sign}^{H}(\mathsf{sk}', m), H) \geq \omega(\log \lambda)\,.\end{aligned}$$

The first equality holds by noticing that $m \to (\mathsf{aux}^{H}(m, \mathsf{pk}), H) \to \mathsf{pk}$ forms a Markov chain, the second equality holds by the construction of aux^{H}, and the

last inequality holds from the entropy requirement. Collecting the above yields

$$\mathbf{Adv}^{\mathsf{NR}^H}_{\mathcal{S}}(\mathcal{A}_{\mathsf{NR}}, \mathsf{aux}) \geq 1 - \mathsf{negl}(\lambda)\,,$$

which concludes the proof. □

$\mathcal{D}^H(\mathsf{pk})$:	$\mathcal{A}^H(\mathsf{pk}, \sigma, h)$:	$\mathsf{aux}^H(m, \mathsf{pk})$:
1: $m \leftarrow \mathcal{M}$	1: $(\sigma, \mathsf{pk}') := h$	1: $(\mathsf{sk}', \mathsf{pk}') \leftarrow \mathsf{KGen}^H(1^\lambda)$
2: **return** m	2: **return** (σ, pk')	2: $\sigma \leftarrow \mathcal{S}.\mathsf{Sign}^H(\mathsf{sk}', m)$
		3: **return** (σ, pk')

Figure 4.6: Adversary $\mathcal{A}^H_{\mathsf{NR}} = (\mathcal{D}^H, \mathcal{A}^H)$ and function aux used in the proof of Theorem 4.11.

We can leverage Theorem 4.11 to show that a BUFF-transformed signature scheme does not achieve non-resignability in the ROM. This is formalized in the following corollary.

Corollary 4.12. *Let H be a random oracle, $\mathcal{F}^H$ be a PPT hash function given query access to H, compressing by at least the size of the public-key plus $\omega(\log \lambda)$ bits, $\mathcal{S}^H$ be a signature scheme given query access to H, and $\mathsf{BUFF}[\mathcal{S}, \mathcal{F}]$ be the signature scheme obtained via the BUFF transform with $\mathcal{F}^H$. Then there exists a PPT adversary $\mathcal{A}^H_{\mathsf{NR}} = (\mathcal{D}^H, \mathcal{A}^H)$ and a PPT function aux such that the entropy condition (4.3) is satisfied, yet*

$$\mathbf{Adv}^{\mathsf{NR}^H}_{\mathsf{BUFF}[\mathcal{S},\mathcal{F}]}(\mathcal{A}_{\mathsf{NR}}, \mathsf{aux}) \geq 1 - \mathsf{negl}(\lambda)\,.$$

The above negative claim on the BUFF transform contradicts [CDF+21], which claims that the BUFF transform does satisfy non-resignability in the ROM (though without being explicit about the definition in the ROM). We discuss below the source of the false positive claim.

4.3.3 Φ-Non-Malleability in the ROM

[CDF+21] argues the non-resignability of the BUFF transform (in the ROM) in two steps. First, they prove the security of the BUFF transform *in the plain model* under the assumption that the hash function (family) satisfies the notion of Φ-non malleability (for a certain class Φ of functions). The formal statement is given in [CDF+21, Theorem 5.5] (cf. Theorem 4.8). Then, the following is remarked in [CDF+21, page 9], from which it is then concluded that the BUFF transform satisfies non-resignability in the ROM.

> *We note that if we model H as a random oracle then the hash function satisfies the definition of Φ-non-malleability for any class Φ*

> *where the functions ϕ preserve sufficient entropy in x, as will be the case for our results. The reason is that the adversary can only output a related random oracle value y' if it has queried the random oracle about $\phi(x)$ before. But this is infeasible if $\phi(x)$ still contains enough entropy.*

Note that this claim originates from [BFS11], where a similar argument is made.

We show that this claim on the Φ-non-malleability of the random oracle is incorrect (under some mild assumption on Φ). As a matter of fact, the same kind of attack as for the non-resignability of signature schemes applies here as well: we can simply let aux compute the mauled hash value. This bypasses the argument that the adversary has to make this particular query to the random oracle.

First, we explicitly spell out in Fig. 4.7 the security game of Φ-non-malleability *in the ROM*, for any PPT hash function $\mathcal{F}^H$ with query access to the random oracle H. Then, we state the negative result in Theorem 4.13 below. Note that the latter in particular implies that the random oracle itself, i.e., when setting $\mathcal{F}^H = H$, is not Φ-non-malleable.

$\Phi\text{-}\mathsf{NM}^H_{\mathcal{F}}$:
1: $x \leftarrow \mathcal{D}^H$
2: $h := \mathsf{aux}^H(x)$
3: $y := \mathcal{F}^H(x)$
4: $(y', \phi) \leftarrow \mathcal{A}^H(y, h)$
5: **return** $(\mathcal{F}^H(\phi(x)) = y' \wedge \phi(x) \neq x)$

Figure 4.7: Security game $\Phi\text{-}\mathsf{NM}^H_{\mathcal{F}}$ for the hash function $\mathcal{F}^H$ in the ROM and an arbitrary function family Φ.

Theorem 4.13. *Let H be a random oracle, $\mathcal{F}^H : \mathcal{X} \to \mathcal{Y}$ be a PPT hash function given query access to H, compressing by least $\omega(\log \lambda)$ bits, and $\Phi \subseteq \mathcal{X}^{\mathcal{X}}$ be such that there is a PPT algorithm $\mathcal{D}_\Phi$ producing $\phi \in \Phi$ that does not fix most points with overwhelming probability, i.e.*

$$\Pr_{\substack{\phi \leftarrow \mathcal{D}_\Phi \\ x \leftarrow \mathcal{X}}} [\phi(x) = x] \leq \mathsf{negl}(\lambda) \; . \tag{4.4}$$

Then, there exist a PPT adversary $\mathcal{A}^H_{\Phi\text{-}\mathsf{NM}} = (\mathcal{D}^H, \mathcal{A}^H)$ and a polynomial-time computable function aux^H, both given query access to H, such that the entropy condition

$$\mathsf{H}_\infty\big(x \,\big|\, \mathsf{aux}^H(x), H\big) \geq \omega(\log \lambda)$$

is satisfied, yet

$$\mathbf{Adv}_{\mathcal{F}}^{\Phi\text{-}\mathsf{NM}^H}(\mathcal{A}_{\Phi\text{-}\mathsf{NM}}, \mathsf{aux}) \geq 1 - \mathsf{negl}(\lambda)\,.$$

Proof. We give a PPT adversary $\mathcal{A}^H_{\Phi\text{-}\mathsf{NM}} = (\mathcal{D}^H, \mathcal{A}^H)$ that wins the game $\Phi\text{-}\mathsf{NM}^H$ with overwhelming probability. Both $\mathcal{A}^H_{\Phi\text{-}\mathsf{NM}} = (\mathcal{D}^H, \mathcal{A}^H)$ and aux^H are given in Fig. 4.8. In the first stage, adversary $\mathcal{D}$ chooses x uniformly at random. The auxiliary function on input $x \in \mathcal{X}$, chooses a function $\phi \leftarrow \mathcal{D}_\Phi$, computes $y' := \mathcal{F}^H(\phi(x))$, and returns the pair (y', ϕ) as hint. In the second stage, adversary $\mathcal{A}$ gets $y = \mathcal{F}^H(x)$ along with the hint $h = (y', \phi)$ as input and outputs the (y', ϕ) from the hint. It is easy to see that $1 \leftarrow \Phi\text{-}\mathsf{NM}^H$, unless $\phi(x) = x$, which only happens with negligible probability due to (4.4). Furthermore, the entropy requirement follows from the fact that x is chosen uniformly from $\mathcal{X}$ independent of (ϕ, H), and that $\mathcal{F}^H$ is compressing by $\omega(\log \lambda)$ bits. □

$\mathcal{D}^H(\mathsf{pk})$	$\mathcal{A}^H(y, h)$	$\mathsf{aux}^H(x)$
1: $x \leftarrow \mathcal{X}$	1: $(y', \phi) := h$	1: $\phi \leftarrow \mathcal{D}_\Phi$
2: **return** x	2: **return** (y', ϕ)	2: $y' := \mathcal{F}^H(\phi(x))$
		3: **return** (y', ϕ)

Figure 4.8: The adversary $\mathcal{A}^H_{\Phi\text{-}\mathsf{NM}} = (\mathcal{D}^H, \mathcal{A}^H)$ and the function aux^H used in the proof of Theorem 4.13.

Remark 4.14. *The above theorem is stated for the random oracle model. It is easy to see, though, that the result carries over to the plain model for concrete hash functions.*

4.4 Salted BUFF and NR-bot

In this section, we partly recover from the negative results from the previous section by considering a salted version of the BUFF transform, and showing that it satisfies a weaker variant of non-resignability in the ROM. At the end of this section, we show another negative result, when the entropy requirement posed on the adversaries is only computational.

4.4.1 Positive Results

The formal specification of the salted BUFF transform, denoted $\$\text{-}\mathsf{BUFF}$, is given in Fig. 4.9. It matches with the original BUFF transform, except that some random salt s is added to the signature and used for the hash.

Our goal is to show that the salted BUFF transform satisfies the weaker variant of non-resignability in the ROM obtained by replacing the non-resignability

KGen:	$\mathsf{Sign}(\mathsf{sk}, m)$:	$\mathsf{Vrfy}(\mathsf{pk}, m, (\sigma, y, s))$:
1: $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}_{\mathcal{S}}$	1: $s \leftarrow \{0,1\}^{\ell}$	1: $y' := \mathcal{F}(m, \mathsf{pk}, s)$
2: **return** $(\mathsf{sk}, \mathsf{pk})$	2: $y := \mathcal{F}(m, \mathsf{pk}, s)$	2: $d := \mathsf{Vrfy}_{\mathcal{S}}(\mathsf{pk}, y', \sigma)$
	3: $\sigma \leftarrow \mathsf{Sign}_{\mathcal{S}}(\mathsf{sk}, y)$	3: **return** $(d = 1 \wedge y = y')$
	4: **return** (σ, y, s)	

Figure 4.9: The salted BUFF transform $\$\text{-}\mathsf{BUFF}[\mathcal{S}, \mathcal{F}]$ for a signature scheme $\mathcal{S} = (\mathsf{KGen}_{\mathcal{S}}, \mathsf{Sign}_{\mathcal{S}}, \mathsf{Vrfy}_{\mathcal{S}})$ and a hash function $\mathcal{F}$.

game $\mathsf{NR}_{\mathcal{S}}$ to $\mathsf{NR}_{\mathcal{S}}^{H,\perp}$, as given in Fig. 4.10. The only difference is that the function aux, which computes a piece of auxiliary information, is not given access to the random oracle anymore.

$\mathsf{NR}_{\mathcal{S}}^{H,\perp}$

1: $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}^{H}$
2: $m \leftarrow \mathcal{D}^{H}(\mathsf{pk})$
3: $(\sigma', \mathsf{pk}') \leftarrow \mathcal{A}^{H}\left(\mathsf{pk}, \mathsf{Sign}^{H}(\mathsf{sk}, m), \mathsf{aux}(m, \mathsf{pk})\right)$
4: **return** $\mathsf{Vrfy}^{H}(\mathsf{pk}', m, \sigma') \ \wedge \ \mathsf{pk}' \neq \mathsf{pk}$

Figure 4.10: The $\mathsf{NR}_{\mathcal{S}}^{H,\perp}$ game.

Indeed, below we will prove the following. To start with, we consider the entropy condition on the message to be statistical; as explained in Section 4.3.2, here in the ROM we additionally need to condition on H (in order to avoid letting $m = H(0)$). In Section 4.4.4, we then discuss the case of computational entropy.

We consider both the case of *classical* and of *quantum* queries by the adversary $\mathcal{A}$ when querying the random oracle, as well as a "semi-quantum" case where $\mathcal{D}$ is classical yet $\mathcal{A}$ may be quantum. The latter is motivated by the fact that $\mathcal{D}$ is typically not adversarially chosen, but determined by the considered application.[10]

Theorem 4.15. *Let H be a random oracle with co-domain denoted by $\mathcal{Y}$, let $\mathcal{S}^H = (\mathsf{KGen}, \mathsf{Sign}^H, \mathsf{Vrfy}^H)$ be a signature scheme with a key generation that has no access to H, and let $\$\text{-}\mathsf{BUFF}[\mathcal{S}, H]$ be the signature scheme obtained by applying the salted BUFF transform (cf. Fig. 4.9) to $\mathcal{S}$. Furthermore, let $\mathcal{A}_{\mathsf{NR}}^H = (\mathcal{D}^H, \mathcal{A}^H)$ be a computationally unbounded $\mathsf{NR}^{H,\perp}$ adversary, aux be any (possibly randomized) function, and let $\mathsf{Sign}^H, \mathcal{D}^H, \mathcal{A}^H$ make at most $q_S, q_{\mathcal{D}}, q_{\mathcal{A}}$*

[10] In the fully quantum case, we even allow the signing procedure to make quantum queries to H; this is not really relevant but obtained for free.

queries to H respectively. Assuming

$$\mathsf{H}_{\infty}^{\substack{(\mathsf{sk},\mathsf{pk})\leftarrow\mathsf{KGen}\\ m\leftarrow\mathcal{D}^{H}(\mathsf{pk})}}\big(m \,\big|\, H,\mathsf{pk},\mathsf{aux}(m,\mathsf{pk})\big) \geq \log(1/\epsilon)\,,$$

then for $\mathsf{Sign}^H, \mathcal{D}^H, \mathcal{A}^H$ making quantum queries in general, it holds that

$$\mathbf{Adv}^{\mathsf{NR}^{H,\perp}}_{\$\text{-}\mathsf{BUFF}[\mathcal{S},H]}(\mathcal{A}_{\mathsf{NR}},\mathsf{aux}) \leq \sqrt{q_{\mathcal{D}}\cdot 2^{-\ell}} + \frac{q_{\mathcal{D}}\cdot 2^{-\ell}}{2} + 4(q_{\mathcal{A}}+q_S)\sqrt{\epsilon} + \frac{(2q_{\mathcal{D}}+1)^2}{|\mathcal{Y}|}\,, \tag{4.5}$$

and if $\mathcal{D}^H$ is restricted to classical queries,

$$\mathbf{Adv}^{\mathsf{NR}^{H,\perp}}_{\$\text{-}\mathsf{BUFF}[\mathcal{S},H]}(\mathcal{A}_{\mathsf{NR}},\mathsf{aux}) \leq q_{\mathcal{D}}\cdot 2^{-\ell} + 4(q_{\mathcal{A}}+q_S)\sqrt{\epsilon} + \frac{(q_{\mathcal{D}}+1)}{|\mathcal{Y}|}\,. \tag{4.6}$$

In case where $\mathsf{Sign}^H, \mathcal{D}^H, \mathcal{A}^H$ are all restricted to classical queries, then we have

$$\mathbf{Adv}^{\mathsf{NR}^{H,\perp}}_{\$\text{-}\mathsf{BUFF}[\mathcal{S},H]}(\mathcal{A}_{\mathsf{NR}},\mathsf{aux}) \leq q_{\mathcal{D}}\cdot 2^{-\ell} + 2(q_{\mathcal{A}}+q_S)\cdot\epsilon + \frac{(q_{\mathcal{D}}+1)}{|\mathcal{Y}|}\,. \tag{4.7}$$

Remark 4.16. *In the setting where the key generation KGen^H is given access to the random oracle H, it is not too hard to extend the non-resignability of* \$-BUFF *into such a setting, by noticing that any sufficiently long portion of the random salt s in* \$-BUFF *is hard to guess by KGen^H, and hence separating the domain queried by KGen^H from ones queried by Sign^H and Vrfy^H up to some negligible advantage.*

4.4.2 Handling Classical Adversaries

Proof of (4.7). The proof proceeds via the games $\mathcal{G}_0,\ldots,\mathcal{G}_6^i$ displayed in Fig. 4.11. Steps from $\mathcal{G}_0$ to $\mathcal{G}_3$ are symmetric, in that they argue closeness between games, while each of the rest upperbounds the winning probability of one game via another.

The closeness between games $\mathcal{G}_0 \approx \mathcal{G}_1 \approx \mathcal{G}_2 \approx \mathcal{G}_3$ is via arguing that an adversary cannot detect some reprogramming in the random oracle except with small probability. For $\mathcal{G}_0 \approx \mathcal{G}_1$, one exploits that the reprogrammed point involves a freshly chosen salt s in uniform distribution. For $\mathcal{G}_1 \approx \mathcal{G}_2 \approx \mathcal{G}_3$, one exploits that m has high entropy, conditioned on the view of the attacker throughout the execution of the intermediate game $\mathcal{G}_2$.

$\mathcal{G}_0$ to $\mathcal{G}_1$ hop. The only difference between $\mathcal{G}_0$ and $\mathcal{G}_1$ is that the former computes $y \leftarrow H(m,\mathsf{pk},s)$, while the latter does a reprogramming via $H(m,\mathsf{pk},s) :=$

$\mathcal{G}_0$

1: $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}$
2: $m \leftarrow \mathcal{D}^H(\mathsf{pk})$
3: $h := \mathsf{aux}(m, \mathsf{pk})$
4: $s \leftarrow \{0,1\}^\ell$
5: $y := H(m, \mathsf{pk}, s)$
6: $(y', \mathsf{pk}', s') \leftarrow \mathcal{B}^H_{\mathsf{sk}}(y, h, s)$
7: **return** $H(m, \mathsf{pk}', s) = y' \wedge \mathsf{pk} \neq \mathsf{pk}'$

$\mathcal{B}^H_{\mathsf{sk}}(y, h, s)$

1: $\sigma \leftarrow \mathsf{Sign}^H(\mathsf{sk}, y)$
2: **return** $\mathcal{A}^H(\mathsf{pk}, \sigma, h)$

$\mathcal{C}^\bullet_{\mathsf{sk}}(m)$

1: $(s, y) \leftarrow \{0,1\}^\ell \times \mathcal{Y}$
2: **return** $\mathcal{B}^\bullet_{\mathsf{sk}}(y, \mathsf{aux}(m, \mathsf{pk}), s)$

$\mathcal{G}_1$

1: $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}$
2: $m \leftarrow \mathcal{D}^H(\mathsf{pk})$
3: $h := \mathsf{aux}(m, \mathsf{pk})$
4: $s \leftarrow \{0,1\}^\ell$
5: $H(m, \mathsf{pk}, s) := y \leftarrow \mathcal{Y}$
6: $(y', \mathsf{pk}', s') \leftarrow \mathcal{B}^H_{\mathsf{sk}}(y, h, s)$
7: **return** $H(m, \mathsf{pk}', s) = y' \wedge \mathsf{pk} \neq \mathsf{pk}'$

$\mathcal{G}_2$

1: $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}$
2: $m \leftarrow \mathcal{D}^H(\mathsf{pk})$
3: $h := \mathsf{aux}(m, \mathsf{pk})$
4: $s \leftarrow \{0,1\}^\ell$
5: $y \leftarrow \mathcal{Y}$
6: $(y', \mathsf{pk}', s') \leftarrow \mathcal{B}^H_{\mathsf{sk}}(y, h, s)$
7: **return** $H(m, \mathsf{pk}', s) = y' \wedge \mathsf{pk} \neq \mathsf{pk}'$

$\mathcal{G}_3$

1: $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}$
2: $m \leftarrow \mathcal{D}^H(\mathsf{pk})$
3: $h := \mathsf{aux}(m, \mathsf{pk})$
4: $s \leftarrow \{0,1\}^\ell$
5: $y \leftarrow \mathcal{Y}$
6: $(y', \mathsf{pk}', s') \leftarrow \mathcal{B}^{H[(m,\cdot,\cdot) \mapsto \perp]}_{\mathsf{sk}}(y, h, s)$
7: **return** $H(m, \mathsf{pk}', s) = y' \wedge \mathsf{pk} \neq \mathsf{pk}'$

$\mathcal{G}_4$

1: $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}$
2: $m \leftarrow \mathcal{D}^H(\mathsf{pk})$
3: $(y', \mathsf{pk}', s') \leftarrow \mathcal{C}^{H[(m,\cdot,\cdot) \mapsto \perp]}_{\mathsf{sk}}(m)$
4: **return** $H(m, \mathsf{pk}', s) = y'$

$\mathcal{G}^i_5$

1: $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}$
2: $m \leftarrow \mathcal{D}^H(\mathsf{pk})$
3: $(y', \mathsf{pk}', s') \leftarrow \mathcal{C}^{H[(m,\cdot,\cdot) \mapsto \perp]}_{\mathsf{sk}}(m)$
4: **return** $H(m, \mathsf{pk}', s) = y'$
5: $\qquad \wedge (m, \mathsf{pk}', s') = (m^i, \mathsf{pk}^i, s^i)$
6: {where $(m^i, \mathsf{pk}^i, s^i)$ is $\mathcal{D}$'s ith query}

$\mathcal{G}^i_6$

1: $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}$
2: $m \leftarrow \mathcal{D}^H(\mathsf{pk})$
3: $(y', \mathsf{pk}', s') \leftarrow \mathcal{C}^{H[(m^i,\cdot,\cdot) \mapsto \perp]}_{\mathsf{sk}}(m^i)$
4: **return** $H(m^i, \mathsf{pk}', s) = y'$
5: $\qquad \wedge (m, \mathsf{pk}', s') = (m^i, \mathsf{pk}^i, s^i)$
6: {where $(m^i, \mathsf{pk}^i, s^i)$ is $\mathcal{D}$'s ith query}

Figure 4.11: The sequence of games considered in the proof of (4.7). In all games, H is understood to be uniformly random. In the game $\mathcal{G}_4$ etc., $H[(m, \cdot, \cdot) \mapsto \perp]$ denotes the oracle that blocks queries of the form $(m, \cdot, \cdot)$, i.e., replies with some special value $\perp$ in that case, and replies with H applied to the query otherwise.

$y \leftarrow \mathcal{Y}$. Thus, both games behave identically unless $\mathcal{D}$ has queried the corresponding input (m, pk, s), which happens with probability at most $q_{\mathcal{D}} \cdot 2^{-\ell}$ in either game, due to the random choice of s.

$\mathcal{G}_1$ to $\mathcal{G}_2$ hop. Without loss of generality, we may assume $\mathsf{pk}' \neq \mathsf{pk}$, in which case the reprogramming of H, done in $\mathcal{G}_1$ but not in $\mathcal{G}_2$, does not affect the final hash $H(m, \mathsf{pk}', s')$. Thus, there is a difference in the two games only if $\mathcal{B}$ makes a query to (m, pk, s); however, in $\mathcal{G}_2$, using that y and s are independent of (m, H, h), $\mathsf{sk} \to (\mathsf{pk}, H, h) \to m$ is a Markov's chain, and by the entropy condition, we have that

$$\mathsf{guess}(m \mid \mathsf{sk}, H, y, h, s) = \mathsf{guess}(m \mid \mathsf{sk}, H, h) = \mathsf{guess}(m \mid \mathsf{pk}, H, h) \leq \epsilon \ ,$$

and so this happens with probability at most $(q_{\mathcal{A}} + q_{\mathcal{S}})\epsilon$. Thus,

$$\Pr\left[1 \leftarrow \mathcal{G}_1\right] \leq \Pr\left[1 \leftarrow \mathcal{G}_2\right] + (q_{\mathcal{A}} + q_{\mathcal{S}})\epsilon \ .$$

$\mathcal{G}_2$ to $\mathcal{G}_3$ hop. Similar as above, the difference between $\mathcal{G}_2$ and $\mathcal{G}_3$ can only be noticed when $\mathcal{B}$ makes a query of the form $(m, \cdot, \cdot)$, which again happens with probability at most $\mathsf{guess}(m \mid \mathsf{sk}, H, y, h, s) \leq \epsilon$. Therefore,

$$\Pr\left[1 \leftarrow \mathcal{G}_2\right] \leq \Pr\left[1 \leftarrow \mathcal{G}_3\right] + (q_{\mathcal{A}} + q_{\mathcal{S}})\epsilon \ .$$

$\mathcal{G}_3$ to $\mathcal{G}_4$ hop. In $\mathcal{G}_4$, we relax the winning condition by dropping the requirement $\mathsf{pk}' \neq \mathsf{pk}$; this only increases the winning probability. Furthermore, we replace $\mathcal{B}$ in $\mathcal{G}_3$ by $\mathcal{C}$ in $\mathcal{G}_4$, which computes $h := h(m)$ and samples $s \leftarrow \{0,1\}^{\ell}$ and $y \leftarrow \mathcal{Y}$ as a first step, and then runs $\mathcal{B}$ on input (y, h, s); this change is only syntactically and does not affect the winning probability. Thus,

$$\Pr\left[1 \leftarrow \mathcal{G}_3\right] \leq \Pr\left[1 \leftarrow \mathcal{G}_4\right] \ .$$

$\mathcal{G}_4$ to $\mathcal{G}_5^i$ hop. Since $\mathcal{D}$ is classical, assume without loss of generality that it never repeats a query. If (m, pk', s') has never been queried by $\mathcal{D}$, i.e., $(m, \mathsf{pk}', s') \neq (m^j, \mathsf{pk}^j, s^j)$ for all $j \in \{1, \ldots, q_{\mathcal{D}}\}$, then (using that $\mathcal{B}$ is blocked from queries of the form $(m, \cdot, \cdot)$) the hash $H(m, \mathsf{pk}', s')$ is random and independent of y, in which case they are equal with probability $1/|\mathcal{Y}|$. Therefore we obtain,

$$\begin{aligned}\Pr\left[1 \leftarrow \mathcal{G}_4\right] &= 1/|\mathcal{Y}| + \sum_{i \in [q_{\mathcal{D}}]} \Pr\begin{bmatrix} 1 \leftarrow \mathcal{G}_4 \\ (m, \mathsf{pk}', s') = (m^i, \mathsf{pk}^i, s^i) \end{bmatrix} \\ &= 1/|\mathcal{Y}| + \sum_{i \in [q_{\mathcal{D}}]} \Pr\left[1 \leftarrow \mathcal{G}_5^i\right] \ .\end{aligned}$$

$\mathcal{G}_5^i$ to $\mathcal{G}_6^i$ hop. In $\mathcal{G}_5^i$, due to the extra condition $m = m^i$ for winning the game, replacing m by m^i as in $\mathcal{G}_6^i$ has no effect on the winning probability, and dropping the requirement again then only increases the probability. Thus

$$\Pr\left[1 \leftarrow \mathcal{G}_5^i\right] \leq \Pr\left[1 \leftarrow \mathcal{G}_6^i\right] .$$

It remains to show that the latter probability is small. First, we may assume that $\mathcal{D}$'s queries $(m^j, \mathsf{pk}^j, s^j)$ are all distinct. Furthermore, we may assume that once $\mathcal{D}$ has decided on the ith query $(m^i, \mathsf{pk}^i, s^i)$, it stops without making this query; the game then simply proceeds as described with running $\mathcal{C}$ on input m^i. This shows that $\mathcal{C}$'s input is independent of $H(m^i, \mathsf{pk}^i, s^i)$, and so is his output y' then, given that he is blocked from queries of the form $(m, \cdot, \cdot)$. Hence

$$\Pr\left[1 \leftarrow \mathcal{G}_6^i\right] \leq 1/|\mathcal{Y}| .$$

Combining all the (in)equalities then concludes (4.7). □

4.4.3 Handling Quantum Adversaries

Proof of (4.5). The proof of (4.5) is identical to that of (4.7) up to some small changes in the argumentation for the first four game hops, and a more significant change of strategy in the last two. Indeed, we reuse the first four games and define modified versions of $\mathcal{G}_5^i$ and $\mathcal{G}_6^i$, as specified in Fig. 4.12. Namely, in case of superposition queries by $\mathcal{G}_0$, we cannot define $(m^i, \mathsf{pk}^i, s^i)$ as the ith query; instead, rather naturally, we introduce $(m^i, \mathsf{pk}^i, s^i)$ by *measuring* the ith query. Furthermore, for technical reasons, we then reprogram H on $(m^i, \mathsf{pk}^i, s^i)$ by a random value Θ, from this or the next query onward.

$\mathcal{G}_5^i$	$\mathcal{G}_6^i$
1: $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}$	1: $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}$
2: $m \leftarrow \mathcal{D}^{H_i^\Theta}$	2: $m \leftarrow \mathcal{D}^{H_i^\Theta}$
3: {measure ith query $(m^i, \mathsf{pk}^i, s^i)$}	3: {measure ith query $(m^i, \mathsf{pk}^i, s^i)$}
4: $(y', \mathsf{pk}', s') \leftarrow \mathcal{C}^{H[(m,\cdot,\cdot)\mapsto\perp]}(m)$	4: $(y', \mathsf{pk}', s') \leftarrow \mathcal{C}^{H[(m^i,\cdot,\cdot)\mapsto\perp]}(m^i)$
5: **return** $H^{\Theta_i}(m, \mathsf{pk}', s') = y'$	5: **return** $H^{\Theta_i}(m^i, \mathsf{pk}^i, s^i) = y'$
6: $\wedge (m, \mathsf{pk}', s') = (m^i, \mathsf{pk}^i, s^i)$	

Figure 4.12: The modified games $\mathcal{G}_5^i$ and $\mathcal{G}_6^i$ for the proof of (4.5). In both games, H is understood to be uniformly random. H_i^Θ is the oracle that implements H until just before the ith query, then measures that query in the computational basis to obtain $(m^i, \mathsf{pk}^i, s^i)$, and subsequently answers queries from $\mathcal{D}$ with $H[(m^i, \mathsf{pk}^i, s^i) \mapsto \Theta]$, i.e., with H but reprogrammed to a random value Θ at $(m^i, \mathsf{pk}^i, s^i)$, either from the ith or the $(i+1)$th query onward, with this choice being made uniformly at random as well.

$\mathcal{G}_0$ to $\mathcal{G}_1$ hop. The only difference between $\mathcal{G}_0$ and $\mathcal{G}_1$ is that the former computes $y := H(m, \mathsf{pk}, s)$, while the latter reprograms $H(m, \mathsf{pk}, s) := y \leftarrow \mathcal{Y}$. With s being uniformly random chosen, this is a direct application of the *adaptive reprogramming lemma* (Theorem 1 in [GHHM21]) to bound the distinguishing probability:

$$\Pr[1 \leftarrow \mathcal{G}_0] \leq \Pr[1 \leftarrow \mathcal{G}_1] + \sqrt{q_{\mathcal{D}} \cdot 2^{-\ell}} + \frac{q_{\mathcal{D}} \cdot 2^{-\ell}}{2} .$$

$\mathcal{G}_1$ to $\mathcal{G}_2$ hop. Without loss of generality, we may assume $\mathsf{pk}' \neq \mathsf{pk}$, in which case the reprogramming of H, done in $\mathcal{G}_1$ but not in $\mathcal{G}_2$, does not affect the final hash $H(m, \mathsf{pk}', s')$. Thus, the only difference between the two games is that $\mathcal{B}$ interacts with the original H in $\mathcal{G}_2$, and with H that is reprogrammed to $\perp$ at the point (pk, m, s) in $\mathcal{G}_1$.

This is a direct application for O2H ([AHU19, Theorem 3]). We note that in game $\mathcal{G}_2$, $\mathcal{B}$ has access to H, y, h and s. Using that y and s are independent of (sk, m, H, h) and that $m \to (\mathsf{pk}, H, h) \to \mathsf{sk}$ forms a Markov's chain, we obtain

$$\mathsf{guess}(m \mid \mathsf{sk}, H, y, h, s) = \mathsf{guess}(m \mid \mathsf{sk}, H, h) = \mathsf{guess}(m \mid \mathsf{pk}, H, h) \leq \epsilon ,$$

where the last inequality is given by the entropy condition. Thus, measuring a random query of $\mathcal{B}$ in $\mathcal{G}_2$ yields (pk, m, s) with probability at most ϵ. Therefore, by O2H,

$$\Pr[1 \leftarrow \mathcal{G}_1] \leq \Pr[1 \leftarrow \mathcal{G}_2] + 2(q_{\mathcal{A}} + q_{\mathcal{S}})\sqrt{\epsilon} .$$

$\mathcal{G}_2$ to $\mathcal{G}_3$ hop. Again by O2H - arguing as above that the measurement outcome when measuring a random query of $\mathcal{B}$ in $\mathcal{G}_2$ is of the form $(m, \cdot, \cdot)$ with probability at most ϵ - we obtain

$$\Pr[1 \leftarrow \mathcal{G}_2] \leq \Pr[1 \leftarrow \mathcal{G}_3] + 2(q_{\mathcal{A}} + q_{\mathcal{S}})\sqrt{\epsilon} .$$

$\mathcal{G}_3$ to $\mathcal{G}_4$ hop. Here we argue precisely as in the classical case: we relax the winning condition, and we do a syntactical change by introducing $\mathcal{C}$, which does the computation of h and the sampling of s and y locally, before it runs $\mathcal{B}$. Thus, also here

$$\Pr[1 \leftarrow \mathcal{G}_3] \leq \Pr[1 \leftarrow \mathcal{G}_4] .$$

$\mathcal{G}_4$ to $\mathcal{G}_5^i$ hop. In $\mathcal{G}_5^i$, the oracle for $\mathcal{D}$ is replaced by H_i^{Θ}, which implements H until just before the ith query, then measures that query in the computational basis to obtain $(m^i, \mathsf{pk}^i, s^i)$, and subsequently switches to $H[(m^i, \mathsf{pk}^i, s^i) \mapsto \Theta]$ either from the ith or the $(i+1)$th query onward, with this choice being made uniformly at random.

The goal here is to use the measure-and-reprogram technique from [DFM20] to control the effect of this change. For this purpose, we consider the oracle algorithm $\mathcal{E}^H(\mathcal{D},\mathcal{C})$, which simply runs $m \leftarrow \mathcal{D}^H$ followed by $(y',\mathsf{pk}',s') \leftarrow \mathcal{C}^{H[(m,\cdot,\cdot)\mapsto\perp]}(m)$.

We allow $\mathcal{E}$ conditional superposition query access to the random oracle H, which it uses to forward all queries from $\mathcal{D}$ unconditionally and queries x from $\mathcal{C}$ conditional on $x \neq (m,\cdot,\cdot)$, returning $\perp$ for $x = (m,\cdot,\cdot)$. $\mathcal{E}^H$ is thus an oracle algorithm with query complexity $q_0 + q_1$ and such that in its second phase the only query inputs with non-zero amplitude are of the form $x \neq (m,\cdot,\cdot)$. At the end of its run, $\mathcal{E}^H(\mathcal{D},\mathcal{C})$ outputs (x,z) with $x := (m,\mathsf{pk}',s')$ and $z := y'$.

Furthermore, we define the verification predicate $V(x,y,z)$ that is 1 if and only if $y = z$. Then, $V(x,H(x),z) = 1$ if and only if $H(m,\mathsf{pk}',s') = y'$, which is the verification condition in $\mathcal{G}_4$. Thus,

$$\Pr\left[V(x,H(x),z) = 1 : (x,z) \leftarrow \mathcal{E}^H(\mathcal{D},\mathcal{C})\right] = \Pr\left[1 \leftarrow \mathcal{G}_4(\mathcal{D},\mathcal{C})\right] .$$

We are now in a situation where we can apply a modified version of the measure-and-reprogram technique from Theorem A.4 in the Appendix, which ensures the existence of a "simulator" $\mathcal{S}^{\mathcal{E}}$ such that, for a random Θ,

$$\begin{aligned}&\frac{\Pr[V(x,H(x),z) = 1 : (x,z) \leftarrow \mathcal{E}^H]}{(2q+1)^2}\\ &\leq \Pr\left[V(x,\Theta,z) = 1 \wedge x = x' : (x',x,z,i) \leftarrow \langle \mathcal{S}^{\mathcal{E}}(Q),\Theta\rangle\right] ,\end{aligned}$$

where Q is a set of queries where $\mathcal{S}$ has non-zero probability of success, and $q = |Q|$. We need to describe $\mathcal{S}$ in a bit more detail before we can determine Q.

In the following, let $Q_0 := \{1,\ldots,q_{\mathcal{D}}\}$ and $Q_1 := \{q_{\mathcal{D}}+1,\ldots,q_{\mathcal{D}}+q_{\mathcal{A}}+q_{\mathcal{S}}\}$. The algorithm $\langle \mathcal{S}^{\mathcal{E}},\Theta\rangle$ works as follows: it measures the ith query of $\mathcal{E}^H$ for a random $i \in Q_0 \cup Q_1 \cup \{q_{\mathcal{D}} + q_{\mathcal{A}} + q_{\mathcal{S}} + 1\}$, with the measurement outcome being x', and then reprograms future queries to input x' by Θ (starting from this or the next query, chosen uniformly at random).[11] Finally, $\mathcal{S}$ outputs x' along with i and the final output (x,z) of $\mathcal{E}$.

In the case of our algorithm $\mathcal{E}$ it is easy to determine Q; $\mathcal{E}$ knows m by the end of its first phase, and by construction 1. never queries any input of the form $(m,\cdot,\cdot)$ from that point on and 2. at the end of its run outputs $x = (m,\mathsf{pk}',s')$. Hence, for $i \in Q_1$ we have $x \neq x'$ with certainty and thus

$$\Pr_{(x',x,z,i)\leftarrow\langle \mathcal{S}^{\mathcal{E}},\Theta\rangle}[V(x,\Theta,z) = 1 \wedge x = x' | i \in Q_1] = 0 .$$

It follows that $Q = Q_0$ and therefore $q = q_0$.

[11] The choice $i = q_{\mathcal{D}} + q_{\mathcal{A}} + q_{\mathcal{S}} + 1$ indicates that the final output (m,pk',s') of $\mathcal{E}$ is measured, instead of one of its queries.

Thus, *conditioned* on $i \in Q_0$, $\langle \mathcal{S}^{\mathcal{E}}, \Theta \rangle$ works as $\mathcal{D}^{H_i^{\Theta}}$ followed by $\mathcal{C}^{H[(m,\cdot,\cdot)\mapsto\perp]}(m)$ in $\mathcal{G}_5^i$ for a random $i \in Q_0$, and the event $V(x,\Theta,z) = 1 \wedge x = x'$ matches with the winning condition of $\mathcal{G}_5^i$.

Hence, omitting the specification $(x', x, z, i) \leftarrow \langle \mathcal{S}^{\mathcal{E}}, \Theta \rangle$ of the probability space and writing V as a shorthand for $V(x,\Theta,z)$ in the expressions to simplify notation, we obtain

$$\begin{aligned}
\Pr[V = 1 \wedge x = x'] &= \Pr[i \in Q_0] \Pr\left[V = 1 \wedge x = x' \mid i \in Q_0\right] \\
&\quad + \Pr[i \notin Q_0] \Pr\left[V = 1 \wedge x = x' \mid i \notin Q_0\right] \\
&= \frac{q_{\mathcal{D}}}{q_{\mathcal{D}}+1} \Pr\left[1 \leftarrow \mathcal{G}_5^i \mid i \in Q_0\right] \\
&\quad + \frac{1}{q_{\mathcal{D}}+1} \Pr\left[V = 1 \wedge x = x' \mid i \notin Q_0\right] . \qquad (4.8)
\end{aligned}$$

Finally, we argue that for $(x', x, z, i) \leftarrow \langle \mathcal{S}^{\mathcal{E}}, \Theta \rangle$

$$\Pr[V(x,\Theta,z) = 1 \wedge x = x' \mid i \notin Q_0] = \frac{1}{|\mathcal{Y}|} .$$

Consider $i \notin Q_0$, i.e. $\mathcal{S}$ measures the final output of $\mathcal{E}$. Then $\mathcal{C}$ learns no information on Θ before producing its output, and so $V(x,\Theta,z) = 1$ with probability $\frac{1}{|\mathcal{Y}|}$.

Putting all together, we obtain that

$$\frac{\Pr\left[1 \leftarrow \mathcal{G}_4\right]}{(2q_{\mathcal{D}}+1)^2} \leq \frac{q_{\mathcal{D}}}{q_{\mathcal{D}}+1} \Pr\left[1 \leftarrow \mathcal{G}_5^i \mid i \in Q_0\right] + \frac{1}{(q_{\mathcal{D}}+1) \cdot |\mathcal{Y}|} .$$

$\mathcal{G}_5^i$ to $\mathcal{G}_6^i$ hop. Let $i \in Q_0$ now be fixed. As in the classical case, due to the extra condition in $\mathcal{G}_5^i$, we may replace the occurrence of m with m^i without affecting the output of the game. Thus

$$\Pr\left[1 \leftarrow \mathcal{G}_5^i\right] = \Pr\left[1 \leftarrow \mathcal{G}_6^i\right] .$$

Similarly (but not identically) to the classical case, we can argue the latter probability to be small. Indeed, we may assume that $\mathcal{D}$ stops after having produced the ith query, which is then measured. This then means, given that $H[(m^i,\cdot,\cdot) \mapsto \perp])$ blocks the query that would reveal Θ, the output (y', pk', s') produced by $\mathcal{C}$ is independent of Θ. Thus, the probability that $y' = \Theta$ is at most $1/|\mathcal{Y}|$, showing that

$$\Pr\left[1 \leftarrow \mathcal{G}_6^i\right] = 1/|\mathcal{Y}| \qquad \text{for all fixed } i \in Q_0.$$

Substituting terms in Equation 4.8, we obtain that

$$\Pr\left[1 \leftarrow \mathcal{G}_4\right] \leq \frac{(2q_{\mathcal{D}}+1)^2}{|\mathcal{Y}|} \ .$$

Combining the above bounds concludes the proof. □

Given that, in typical applications, $\mathcal{D}$ is not adversarially chosen but determined by the environment, and typical applications take place in a classical environment, it makes sense to also consider the "semi-quantum case" in (4.6) where we restrict $\mathcal{D}$ to classical queries, but we still allow $\mathcal{A}$ to be quantum. By an appropriate mix-and-match of the classical proof (of (4.7)) and the fully quantum proof (of (4.5)), we immediately conclude (4.6).

4.4.4 Negative Results with *Computational* Entropy

Below, we show that our positive result on the salted BUFF transform in the ROM does not carry over to the computational setting when considering the entropy requirement to be computational, i.e., captured by the above notion of HILL entropy in the ROM. Concretely, we show that under the computational Diffie-Hellman (CDH) assumption, there exists a (contrived) signature scheme that is secure in the standard sense (and thus a meaningful signature scheme), but for which the salted BUFF transformation does not provide the computational variant of $\mathsf{NR}^{H,\perp}$. Formally, this is summarized in Theorem 4.17.

Theorem 4.17. *Let H be a random oracle with co-domain $\mathcal{Y}$. Assuming CDH is hard, there exists a signature scheme $\mathcal{S}^H = (\mathsf{KGen}, \mathsf{Sign}^H, \mathsf{Vrfy}^H)$ in the ROM, with a key generation that has no access to H, for which $\$\text{-}\mathsf{BUFF}[\mathcal{S}, H]$, obtained by applying the salted BUFF transform (see Fig. 4.9), satisfies the following:*

There exists a PPT adversary $\mathcal{A}^H_{\mathsf{NR}} = (\mathcal{D}^H, \mathcal{A}^H)$ given query access to H, and a PPT function aux without any query to H such that

$$\mathsf{HILL}^{H}_{\infty}_{\substack{(\mathsf{sk},\mathsf{pk})\leftarrow\$\text{-}\mathsf{BUFF}[\mathcal{S},H].\mathsf{KGen}(1^\lambda)\\ m\leftarrow\mathcal{D}^H(\mathsf{pk})}} \left(m \,\middle|\, \mathsf{pk}, \mathsf{aux}(m,\mathsf{pk})\right) \geq \log(|\mathcal{Y}|) \ ,$$

and yet they win the game $\mathsf{NR}^{H,\perp}$ against $\$\text{-}\mathsf{BUFF}[\mathcal{S}, H]$ with overwhelming probability, i.e.,

$$\mathbf{Adv}^{\mathsf{NR}^{H,\perp}}_{\$\text{-}\mathsf{BUFF}[\mathcal{S},H]}(\mathcal{A}_{\mathsf{NR}}, \mathsf{aux}) \geq 1 - \mathsf{negl}(\lambda) \ .$$

Moreover, $\mathcal{S}^H$ is strongly unforgeable under chosen message attacks.

Let $S^H_\circ$ be an arbitrary CDH-based (strongly unforgeable) signature scheme, with a key generation that does not query H. Define S to be as $S_\circ$, but modified

as follows. The key generation additionally produces a pair (a, g^a), and attaches a to the secret key and g^a to the public key. Furthermore, signing attaches a to the actual signature (but will be ignored by the verification). Then, we consider an attacker that produces the message m as $m := (H(g^{ab}), g^b)$, and the auxiliary function $\mathsf{aux}(m, \mathsf{pk}) := g^b$. Then we have

$$\mathsf{HILL}^{H}_{\infty}\big(m \,\big|\, \mathsf{pk}, \mathsf{aux}(m, \mathsf{pk})\big) \geq \mathsf{HILL}^{H}_{\infty}\big(H(g^{ab}) \,\big|\, g^a, g^b\big)\,,$$

which is at least as large as $\log(|\mathcal{Y}|)$ by the CDH assumption; yet when given the signature of m, which includes a (be it BUFF transformed or not), the attacker can compute all of m and so produce a new signature by freshly signing m, which breaks the $\mathsf{NR}^{H,\perp}$ security of \$-$\mathsf{BUFF}[\mathcal{S}, H]$.

4.5 BUFF and sNR

4.5.1 Secret-key Non-resignability (sNR)

In this section, we consider a new variant of *non-resignability*, called secret-key non-resignability and denoted by $\mathsf{sNR}^{H,\perp}$. It is similar in spirit as $\mathsf{NR}^{H,\perp}$ introduced in Section 4.4; in particular, a crucial aspect is that aux is not given access to H, but we additionally provide the adversary with the secret key sk, and we adjust the entropy condition correspondingly (see below for a more detailed comparison). The security game is shown in Fig. 4.13. It is played by randomized oracle algorithms[12],

$$\mathcal{D}^H : \mathcal{SK} \to \mathcal{M} \qquad \text{and} \qquad \mathcal{A}^H : \mathcal{SK} \times \mathcal{SGN} \times \mathcal{AUX} \to \mathcal{PK} \times \mathcal{SGN}$$

given query access to H, referred to as *adversaries*, and a randomized algorithm $\mathsf{aux} : \mathcal{SK} \times \mathcal{M} \to \mathcal{AUX}$ with no access to H, referred to as *hint function.*[13]

$\mathsf{sNR}^{H,\perp}_{\mathcal{S}}(\mathcal{D}, \mathcal{A}, \mathsf{aux})$:
1: $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}^H$
2: $m \leftarrow \mathcal{D}^H(\mathsf{sk})$
3: $\sigma \leftarrow \mathsf{Sign}^H(\mathsf{sk}, m)$
4: $(\mathsf{pk}', \sigma') \leftarrow \mathcal{A}^H(\mathsf{sk}, \sigma, \mathsf{aux}(m, \mathsf{sk}))$
5: **return** $\mathsf{pk} \neq \mathsf{pk}' \wedge \mathsf{Vrfy}^H(\mathsf{pk}', m, \sigma') = 1$

Figure 4.13: Our new variant of the non-resignability game $\mathsf{sNR}^{H,\perp}$.

[12] Here and in the remainder, we borrow from set notation to indicate the input and output space of (oracle) algorithms. In case of an algorithm that takes no input, we write the singleton set $\{\perp\}$ as domain.

[13] The hint function may be randomized, but we refer to it as a function for convenience.

While playing $\mathsf{sNR}^{H,\perp}$, we consider restricted ($\mathcal{S}$-dependent) classes of adversaries with a give bound h on the entropy

$$\mathsf{H}_{\infty}^{\substack{(\mathsf{sk},\mathsf{pk})\leftarrow\mathsf{KGen}^{H} \\ m\leftarrow\mathcal{D}^{H}(\mathsf{sk})}}(m \mid H, \mathsf{sk}, \mathsf{aux}(\mathsf{sk}, m)) \geq h\,. \tag{4.9}$$

For now we only consider the statistical variant, where we take an arbitrary but fixed security parameter for $\mathcal{S}$, where $\mathcal{D}$, $\mathcal{A}$ and aux may be computationally unbounded and we only limit their query complexity, and where the entropy requirement holds statistically, i.e., as in (4.9). The computational setting is handled later in Section 4.8; there, $\mathcal{D}, \mathcal{A}$ and aux are restricted to be (uniform or non-uniform) PPT algorithms, and the entropy requirement is expressed via HILL entropy.

Informally, we say that a signature scheme $\mathcal{S} = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vrfy})$ is *non-resignable* if for all $\mathcal{D}$, $\mathcal{A}$ and any hint function aux that satisfy the statistical entropy condition (4.9) for sufficiently large h, the probability of winning the $\mathsf{sNR}^{H,\perp}$ game, i.e.,

$$\mathbf{Adv}^{\mathsf{sNR}^{H,\perp}}(\mathcal{D}, \mathcal{A}, \mathsf{aux}) := \Pr\left[1 = \mathsf{sNR}_{\mathcal{S}}^{H,\perp}(\mathcal{D}, \mathcal{A}, \mathsf{aux})\right]\,,$$

is small.

The recent developments have shown that formalizing non-resignability is a non-trivial task, and different weaker variants of the original (unachievable) version have been proposed. We quickly discuss here how $\mathsf{sNR}^{H,\perp}$ relates to those variants; namely, we show that is stronger than the versions proposed in Section 4.4 and [CDF+23].

Comparison with Non-Resignability from Section 4.4. The difference to $\mathsf{NR}^{H,\perp}$ as defined in Section 4.4 is that $\mathsf{sNR}^{H,\perp}$ provides the $\mathcal{D}$, $\mathcal{A}$ and the hint function aux with the secret key sk, whereas $\mathsf{NR}^{H,\perp}$ only provides the public key pk (recall that we assume that pk can be computed from sk). This of course gives more power to the adversary. The other difference lies in the entropy requirement: for $\mathsf{NR}^{H,\perp}$, the message is required to have high entropy conditioned on pk (and aux) only, i.e.,

$$\mathsf{H}_{\infty}(m \mid H, \mathsf{pk}, \mathsf{aux}(\mathsf{pk}, m)) \geq h$$

whereas $\mathsf{sNR}^{H,\perp}$ requires (4.9) to hold, which conditions on sk instead; this seems to be a stronger restriction, but we observe that for $m \leftarrow \mathcal{D}(\mathsf{pk})$, produced by a $\mathcal{D}$ that only gets the public key as input (as in $\mathsf{NR}^{H,\perp}$),

$$\mathsf{H}_{\infty}(m \mid H, \mathsf{pk}, \mathsf{aux}(\mathsf{pk}, m)) = \mathsf{H}_{\infty}(m \mid H, \mathsf{sk}, \mathsf{aux}(\mathsf{pk}, m))$$

since $\mathsf{sk} \to (H, \mathsf{pk}, \mathsf{aux}(\mathsf{pk}, m)) \to m$ forms a Markov chain then. This implies that any attack against $\mathsf{NR}^{H,\perp}$ can be cast as an attack against $\mathsf{sNR}^{H,\perp}$ with the same entropy bound, making the latter a stronger security notion.

Comparison with Non-Resignability from [CDF+23]. We first note that [CDF+23] defines non-resignability only in the computational setting, so we compare it with the computational version of $\mathsf{sNR}^{H,\perp}$. While we have postponed the exact definition to Section 4.8, the high level reasoning can still be understood. First of all, in [CDF+23] the side information on m (given by aux in our case) is required to be *computationally independent* of m, which is equivalent to allowing *no side information at all* when considering computationally bounded adversaries. Furthermore, in line with $\mathsf{sNR}^{H,\perp}$, the entropy condition (though phrased in terms of HILL entropy) is required to hold when conditioning on the secret key sk. But on the other hand and in the spirit of $\mathsf{NR}^{H,\perp}$, the adversaries are only given pk as input, and not sk. Altogether, this makes their notion weaker than our computational version of $\mathsf{sNR}^{H,\perp}$, which provided sk as input to the adversaries.

4.5.2 The Hide-and-Seek Game

To prove the non-resignability (in the sense of $\mathsf{sNR}^{H,\perp}$) of the BUFF transform, which signs a message m by signing $\mathcal{F}(\mathsf{pk}, m)$, with the hash value then appended to the signature, it must *necessarily* be hard to recover m from $H(m, \mathsf{pk})$. This hardness may look trivial at first glance, since $\mathcal{F}$ is (typically) compressing, and modelled as a random oracle; however, it turns out to be not trivial at all. The reason is that in the $\mathsf{sNR}^{H,\perp}$ game, m is produced arbitrarily and dependent on $\mathcal{F}$, with the only promise being that m is hard to guess from scratch (i.e., when $\mathcal{F}(\mathsf{pk}, m)$ is not given).

We formally capture (a particular formulation of) this hardness via a game, which we call *Hide-and-Seek*. Looking ahead, in Section 4.5.4 we will show that hardness of winning Hide-and-Seek is *sufficient* for proving the non-resignability of the BUFF transform (when the hash function is modelled as a random oracle). The main technical challenge in Section 4.5 then lies in proving that Hide-and-Seek is hard to win, which we do in Section 4.5.5.

Throughout the remainder of this section, let $\mathcal{X}, \mathcal{Y}$, and $\mathcal{Z}$ be finite non-empty sets, and let $\mathcal{F}^H \colon \mathcal{X} \to \mathcal{Y}$ be any hash function given query access to a random oracle H.

The Hide-and-Seek game is played by two adversaries $\mathcal{D}$ and $\mathcal{A}$: the (possibly query-unbounded) *hider* $\mathcal{D}^H \colon \{\perp\} \to \mathcal{X} \times \mathcal{Z}$, and the query-bounded *seeker* $\mathcal{A}^H \colon \mathcal{Y} \times \mathcal{Z} \to \mathcal{X}$ that is allowed to make at most q queries to H. First, $\mathcal{D}^H$ chooses a challenge $x \in \mathcal{X}$ together with a hint $z \in \mathcal{Z}$ and "hides" x as $\mathcal{F}^H(x)$, and then $\mathcal{A}^H$ is supposed to find x from $H(x)$ and z. The game is formally specified as follows:

$\mathsf{HnS}^H_{\mathcal{F}}(\mathcal{D}, \mathcal{A})$:
1: $(x, z) \leftarrow \mathcal{D}^H$
2: **return** $x = \mathcal{A}^H(\mathcal{F}^H(x), z)$

In line with the entropy condition in $\mathsf{sNR}^{H,\perp}$, we require x to be statistically hidden given H and z. I.e., we require that

$$\mathsf{guess}(x \mid H, z) \leq \epsilon \tag{4.10}$$

for some small $\epsilon > 0$. Informally, we say that the random oracle H satisfies the Hide-and-Seek property, or HnS^H for short, if for every such pair of $\mathcal{D}, \mathcal{A}$ as above, the winning probability, given as

$$\mathbf{Adv}^{\mathsf{HnS}^H}_{\mathcal{F}}(\mathcal{D}, \mathcal{A}) \coloneqq \Pr\left[1 = \mathsf{HnS}^H_{\mathcal{F}}(\mathcal{D}, \mathcal{A})\right] = \Pr_{(x,z)\leftarrow\mathcal{D}^H}\left[x = \mathcal{A}^H(\mathcal{F}^H(x), z)\right],$$

is small.

As mentioned above already, what is tricky about this game is that x (and z) may depend arbitrarily on H, subject to the bound (4.10) on the guessing probability. Because of this, known results on inverting the random oracle do not apply, and it may not be fully clear whether we can actually expect it to be hard to win, i.e., that there is no sneaky way to win the game. We discuss this in more detail in Section 4.5.5, where we then analyze Hide-and-Seek and prove that it *is* hard to win after all.

4.5.3 BUFF via Random Oracles is sNR

Our main goal in this section is to prove that BUFF satisfy the above notion of non-resignability, when modelling the underlying hash function $\mathcal{F}^H := H$ as a random oracle, which we formally state below.

Theorem 4.18. *Let $\mathcal{D}^H : \mathcal{SK} \to \mathcal{M}$ and $\mathcal{A}^H : \mathcal{SK} \times \mathcal{SGN} \times \mathcal{AUX} \to \mathcal{PK} \times \mathcal{SGN}$ be $\mathsf{sNR}^{H,\perp}$-adversaries against $\mathsf{BUFF}[\mathcal{S}, H]$ for some $\mathsf{aux} : \mathcal{SK} \times \mathcal{M} \to \mathcal{AUX}$, making at most q_D and q_A queries to H, respectively, where (4.9) is satisfied for h such that $0 < \epsilon := 2^{-h} \leq \frac{1}{2}$. Then*

$$\mathbf{Adv}^{\mathsf{sNR}^{H,\perp}}_{\mathsf{BUFF}[\mathcal{S},H]}(\mathcal{D}, \mathcal{A}, \mathsf{aux}) \leq 6(q_A + q_S + 1)^2 \log\left(\frac{|\mathcal{SK}| \cdot |\mathcal{AUX}|}{\epsilon}\right)\epsilon + q_K\epsilon + \frac{q_D + 1}{|\mathcal{Y}|}$$

and in the case where $\mathcal{A}$ makes quantum queries, we have $\mathbf{Adv}^{\mathsf{sNR}^{H,\perp}}_{\mathsf{BUFF}[\mathcal{S},H]}(\mathcal{D}, \mathcal{A}, \mathsf{aux})$

$$\leq 18\sqrt{\left(\log\frac{|\mathcal{SK}| \cdot |\mathcal{AUX}|}{\epsilon} + q_{\mathcal{A}} + q_S\right)(q_{\mathcal{A}} + q_S + 1)^3\,\epsilon} + q_K\epsilon + \frac{q_D + 1}{|\mathcal{Y}|}\,.$$

Remark 4.19. *In the case where $\mathcal{D}$ makes quantum queries to H, we expect a similar result (with adjusted bounds) holds, via the method described in the proof of* (4.5) *.*

4.5.4 Reducing sNR of BUFF to Hide-and-Seek

In the following statement, we reduce the $\mathsf{sNR}^{H,\perp}$ security of the BUFF transform $\mathsf{BUFF}[\mathcal{S}, H]$ of a signature scheme $\mathcal{S} = (\mathsf{KGen}^H, \mathsf{Sign}^H, \mathsf{Vrfy}^H)$ to the hardness of winning the Hide-and-Seek game HnS^H. In the lemma statement, the parameters q_K and q_S refer to (an upper bound on) the number of queries to H that KGen^H and Sign^H perform.

Lemma 4.20. *Let $\mathcal{D}^H : \mathcal{SK} \to \mathcal{M}$ and $\mathcal{A}^H : \mathcal{SK} \times \mathcal{SGN} \times \mathcal{AUX} \to \mathcal{PK} \times \mathcal{SGN}$ be $\mathsf{sNR}^{H,\perp}$-adversaries against $\mathsf{BUFF}[\mathcal{S}, H]$ for some $\mathsf{aux} : \mathcal{SK} \times \mathcal{M} \to \mathcal{AUX}$, making at most q_D and q_A queries to H, respectively. Then there exists a hider $\bar{\mathcal{D}} : \{\perp\} \to \mathcal{X} \times \mathcal{Z}$ and a seeker $\bar{\mathcal{A}} : \mathcal{Y} \times \mathcal{Z} \to \mathcal{X}$ with $\mathcal{Z} = \mathcal{SK} \times \mathcal{AUX}$, where $\bar{\mathcal{A}}$ makes at most $q_A + q_S$ queries to H, and such that*

$$\mathsf{H}_\infty\underset{(x,z)\leftarrow\bar{\mathcal{D}}^H}{}(x \mid H, z) = \mathsf{H}_\infty\underset{\substack{(\mathsf{sk},\mathsf{pk})\leftarrow\mathsf{KGen}^H \\ m\leftarrow\mathcal{D}^H(\mathsf{sk})}}{}(m \mid H, \mathsf{sk}, \mathsf{aux}(\mathsf{sk}, m)) \tag{4.11}$$

and $\mathbf{Adv}^{\mathsf{sNR}^{H,\perp}}_{\mathsf{BUFF}[\mathcal{S},H]}(\mathcal{D}, \mathcal{A}, \mathsf{aux})$

$$\leq (q_A + q_S) \cdot \mathbf{Adv}^{\mathsf{HnS}^H}_H(\bar{\mathcal{D}}, \bar{\mathcal{A}}) + q_K\epsilon + \frac{q_D + 1}{|\mathcal{Y}|}\,, \tag{4.12}$$

where $\epsilon := 2^{-\mathsf{H}_\infty(x|H,z)}$. In the case $\mathcal{A}$ makes quantum queries to H, then

$$\mathbf{Adv}^{\mathsf{sNR}^{H,\perp}}_{\mathsf{BUFF}[\mathcal{S},H]}(\mathcal{D}, \mathcal{A}, \mathsf{aux}) \leq 2(q_A + q_S) \cdot \sqrt{\mathbf{Adv}^{\mathsf{HnS}^H}(\bar{\mathcal{D}}, \bar{\mathcal{A}})} + q_K\epsilon + \frac{q_D + 1}{|\mathcal{Y}|}\,, \tag{4.13}$$

holds in place of (4.12)*, and $\bar{\mathcal{A}}$ then makes quantum queries as well.*

Furthermore, in the computational setting when considering a non-fixed security parameter and PPT algorithms $\mathcal{D}$ and $\mathcal{A}$, then $\bar{\mathcal{D}}$ and $\bar{\mathcal{A}}$ are PPT as well.

The intuition behind the proof is as follows. Consider the $\mathsf{sNR}^{H,\perp}$ game. Due to the assumed hardness of Hide-and-Seek, $\mathcal{A}$ cannot recover m from its input and thus makes no query to H that has m as suffix. But then it cannot gather any information on $H(\mathsf{pk}', m)$ for any pk', and thus it will not be able to output $y' = H(\mathsf{pk}', m)$ for any pk'. Formally, we have to make sure that $\mathcal{A}$ gets no information on y' via its input, which is controlled by KGen and $\mathcal{D}$, which may query H on $H(\mathsf{pk}', m)$ for any pk'. This is taken care of in our formal proof below.

Proof. In Fig. 4.14 we define a hybrid sequence reducing the $\mathsf{sNR}^{H,\perp}$ property of $\mathsf{BUFF}[\mathcal{S}, H]$ to the HnS^{H} property of H.

G_0:

1: $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}^{H}$; $m \leftarrow \mathcal{D}^{H}(\mathsf{sk})$
2: $(\mathsf{pk}', y') \leftarrow \mathcal{B}^{H}(\mathsf{sk}, y, \mathsf{aux}(\mathsf{sk}, m))$
3: **return** $H(\mathsf{pk}', m) = y' \ \wedge \ \mathsf{pk}' \neq \mathsf{pk}$

$\mathcal{B}^{H}(\mathsf{sk}, y, h)$:

1: $\sigma \leftarrow \mathsf{Sign}^{H}(\mathsf{sk}, y)$
2: $(\mathsf{pk}', y') \leftarrow \mathcal{A}^{H}(\mathsf{sk}, (\sigma, y), h)$
3: **return** (pk', y')

G_1:

1: $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}^{H}$; $m \leftarrow \mathcal{D}^{H}(\mathsf{sk})$
2: $(\mathsf{pk}', y') \leftarrow \mathcal{B}^{H[(\cdot, m) \mapsto \perp]}(\mathsf{sk}, H(m, \mathsf{pk}), \mathsf{aux}(\mathsf{sk}, m))$
3: **return** $H(\mathsf{pk}', m) = y' \ \wedge \ \mathsf{pk}' \neq \mathsf{pk}$

G_2:

1: $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}^{H}$; $m \leftarrow \mathcal{D}^{H}(\mathsf{sk})$
2: **abort** if KGen queried $H(m, \cdot)$
3: $(\mathsf{pk}', y') \leftarrow \mathcal{B}^{H[(\cdot, m) \mapsto \perp]}(\mathsf{sk}, H(m, \mathsf{pk}), \mathsf{aux}(\mathsf{sk}, m))$
4: **return** $H(\mathsf{pk}', m) = y' \ \wedge \ \mathsf{pk}' \neq \mathsf{pk}$

G_3^i:

1: $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}^{H}$; $m \leftarrow \mathcal{D}^{H}(\mathsf{sk})$
2: **abort** if KGen queried $H(m, \cdot)$
3: $(\mathsf{pk}', y') \leftarrow \mathcal{B}^{H[(\cdot, m) \mapsto \perp]}(\mathsf{sk}, H(m, \mathsf{pk}), \mathsf{aux}(\mathsf{sk}, m))$
4: **return** $H(\mathsf{pk}', m) = y' \ \wedge \ \mathsf{pk}' \neq \mathsf{pk} \ \wedge (\mathsf{pk}', m) = (\mathsf{pk}_i, m_i)$
5: {where (pk_i, m_i) is $\mathcal{D}$'s ith query.}

G_4^i:

1: $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}^{H}$; $m \leftarrow \mathcal{D}^{H}(\mathsf{sk})$
2: **abort** if KGen queried $H(\cdot, m_i)$
3: $(\mathsf{pk}', y') \leftarrow \mathcal{B}^{H[(\cdot, m_i) \mapsto \perp]}(\mathsf{sk}, H(\mathsf{pk}, m_i), \mathsf{aux}(\mathsf{sk}, m_i))$
4: **return** $H(\mathsf{pk}_i, m_i) = y' \ \wedge \ \mathsf{pk}_i \neq \mathsf{pk}$
5: {where (pk_i, m_i) is $\mathcal{D}$'s ith query.}

Figure 4.14: Hybrid steps reducing $\mathsf{sNR}^{H,\perp}$ of $\mathsf{BUFF}[\mathcal{S}, H]$ to HnS^{H} of H when $\mathcal{D}$ is classical, i.e., (4.13), (4.12). In the derivations below we drop the parameter k for notational convenience.

To start with, we note that the adversary $(\mathcal{D}, \mathcal{B})$ playing G_0 is identical to $(\mathcal{D}, \mathcal{A})$ playing $\mathsf{sNR}^{H,\perp}_{\mathsf{BUFF}[\mathcal{S},H]}$.

The G_0 *to* G_1 *hop.* The only difference between G_0 and G_1 is whether $\mathcal{B}$ is

given oracle access to the original random oracle H, or the reprogrammed oracle $H[(m,\cdot) \mapsto \perp]$ and replies with $\perp$ to any query that has suffix m.

Consider the hider $\bar{\mathcal{D}}$ and seeker $\bar{\mathcal{A}}$, where $\bar{\mathcal{D}}$ samples $(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KGen}^H$ and $m \leftarrow \mathcal{D}^H(\mathsf{sk})$ and returns

$$x := (\mathsf{pk}, m) \qquad \text{and} \qquad z := (\mathsf{sk}, \mathsf{aux}(\mathsf{sk}, m))\,,$$

and on input $H(x) = H(m,\mathsf{pk})$ and z, the seeker $\bar{\mathcal{A}}$ samples a random index $i \leftarrow [q_A + q_S]$, runs

$$\mathcal{B}^H(\mathsf{sk}, H(m,\mathsf{pk}), \mathsf{aux}(\mathsf{sk},m)) = \mathcal{A}^H\big(\mathsf{sk}, \big(H(m,\mathsf{pk}), \mathsf{Sign}^H(\mathsf{sk},y)\big), \mathsf{aux}(\mathsf{sk},m)\big)$$

internally, but then looks at / does a full measurement of the ith query to obtain (pk_i^*, m_i^*), and returns (pk, m_i^*). It is clear by construction that $z \in \mathcal{SK} \times \mathcal{AUX}$, and (4.11) immediately follows from the fact that pk can be derived from sk, and so

$$\mathsf{H}_\infty(x \mid H, z) = \mathsf{H}_\infty(\mathsf{pk}, m \mid H, \mathsf{sk}, \mathsf{aux}(\mathsf{sk},m)) = \mathsf{H}_\infty(m \mid H, \mathsf{sk}, \mathsf{aux}(\mathsf{sk},m)) \tag{4.14}$$

as claimed. It also follows from construction that $\bar{\mathcal{D}}$ and $\bar{\mathcal{A}}$ preserve the efficiency of $\mathcal{D}$ and $\mathcal{A}$. In terms of query complexity, $\bar{\mathcal{A}}$ makes at most $q_A + q_S$ queries to H.

In the case where $\mathcal{A}$ makes classical queries, there is no difference in the two games when $\mathcal{B}$ makes no query to a point where the two oracles differ, and thus

$$\begin{aligned}\Pr[1 \leftarrow \mathsf{G}_0] &\le \Pr[1 \leftarrow \mathsf{G}_1] + \Pr\left[\exists\, i \in [q_A+q_S] \text{ s.t. } m_i^* = m\right] \\ &\le \Pr[1 \leftarrow \mathsf{G}_1] + (q_A+q_S)\cdot \mathbf{Adv}^{\mathsf{HnS}^H}(\bar{\mathcal{D}}, \bar{\mathcal{A}})\,.\end{aligned}$$

In the quantum case, the same kind of guarantee follows from the O2H lemma [AHU19, Theorem 3], which gives us that

$$\begin{aligned}\Pr[1 \leftarrow \mathsf{G}_0] &\le \Pr[1 \leftarrow \mathsf{G}_1] + 2(q_A+q_S)\cdot \sqrt{\Pr[m_i^* = m]} \\ &\le \Pr[1 \leftarrow \mathsf{G}_1] + 2(q_A+q_S)\cdot \sqrt{\mathbf{Adv}^{\mathsf{HnS}^H}(\bar{\mathcal{D}}, \bar{\mathcal{A}})}\,.\end{aligned}$$

The G_1 to G_2 hop. The difference between G_1 and G_2 is that the latter aborts if KGen^H ever makes a query of the form $(\cdot, m)$. Given that m is produced given (H,sk) but independent of KGen's q_K queries conditioned on (H,sk), and m satisfies (4.9) for $h := \log(1/\epsilon)$, we have

$$\Pr[1 \leftarrow \mathsf{G}_1] \le \Pr[1 \leftarrow \mathsf{G}_2] + \Pr[\mathsf{G}_2 \text{ abort}] \le \Pr[1 \leftarrow \mathsf{G}_2] + q_K\epsilon\,.$$

The G_2 *to* G_3^i *hop.* Assume without loss of generality that $\mathcal{D}$ never repeats its queries $(k_1, m_1), \ldots, (k_{q_D}, m_{q_D})$. Note that the queries of $\mathcal{A}$ in G_1 are blocked at $(\cdot, m)$, and the game aborts if KGen ever queries $(\cdot, m)$. Since, conditioned on KGen not querying with $(\cdot, m)$, $k' \neq k$ and $(k', m) \neq (k_i, m_i)$ for all i, the output $H(k', m)$ is uniformly random and independent of $\mathcal{A}$'s input $(\mathsf{sk}, H(m, \mathsf{pk}), \mathsf{aux}(\mathsf{sk}, m))$ together with the oracle $H[(m, \cdot) \mapsto \perp]$ it has access to, we have

$$\begin{aligned}\Pr\left[1 \leftarrow \mathsf{G}_2\right] &\leq \Pr\left[\begin{array}{c}\exists i \in [q_D] \text{ s.t. } (k', m) = (k_i, m_i) \\ 1 \leftarrow \mathsf{G}_2\end{array}\right] \\ &\quad + \Pr\left[1 \leftarrow \mathsf{G}_2 \;\middle|\; \begin{array}{c}\mathsf{KGen} \text{ not querying } (\cdot, m) \\ (k', m) \neq (k_i, m_i)\ \forall i \in [q_D] \\ k' \neq k\end{array}\right] \\ &\leq \sum_{i \in [q_D]} \Pr\left[1 \leftarrow \mathsf{G}_3^i\right] + 1/|\mathcal{Y}| \,.\end{aligned}$$

The G_3^i *to* G_4^i *hop.* Because of the extra condition $(\mathsf{pk}', m') = (\mathsf{pk}_i, m_i)$ in G_3^i, replacing pk' with pk_i and m' with m_i as in G_4^i, does not change the winning probability. We further drop the condition $(\mathsf{pk}', m') = (\mathsf{pk}_i, m_i)$, which does not decrease the winning probability.

Finally, it remains to upper bound the winning probability of G_4^i for each $i \in [q_D]$. By a lazy sampling argument, we note that conditioned on KGen not querying with $(\cdot, m_i)$ and $\mathsf{pk}_i \neq \mathsf{pk}$, the output $H(k_i, m_i)$ is uniform and independent of $(H[(\cdot, m_i) \mapsto \perp], k, H(k, m_i), \mathsf{aux}(m_i))$, and hence, y' generated by $\mathcal{A}^{H[(\cdot, m_i) \mapsto \perp]}(k, H(k, m_i), \mathsf{aux}(m_i))$ is equal to $H(k_i, m_i)$ with probability at most $1/|\mathcal{Y}|$, i.e.

$$\Pr\left[1 \leftarrow \mathsf{G}_4^i\right] \leq 1/|\mathcal{Y}| \,,$$

which concludes (4.12), (4.13). □

Remark 4.21. *We point out that the claim on* $\bar{\mathcal{D}}$ *and* $\bar{\mathcal{A}}$ *being PPT if* $\mathcal{D}$ *and* $\mathcal{A}$ *are, fails to hold when aiming for a variant of Lemma 4.20 that considers* $\mathsf{NR}^{H,\perp}$ *instead of* $\mathsf{sNR}^{H,\perp}$*. The reason is that, on input* $H(m, \mathsf{pk})$ *and* z*, the seeker* $\bar{\mathcal{A}}$ *needs to run* $\mathcal{A}$ *on* a signature of $H(m, \mathsf{pk})$*, which it can do efficiently if given* sk *(which is part of* z *here, exploiting that* $\mathcal{D}$ *is given* sk*), but not if only given* pk*. This is the reason why in the* computational *setting, treated in Section 4.8, our proof for showing that BUFF satisfies* $\mathsf{sNR}^{H,\perp}$ *does not carry over to* $\mathsf{NR}^{H,\perp}$ *(in line with the counter example given in Section 4.4.4).*

4.5.5 Hide-and-Seek for Random Oracles

The following provides a bound on the Hide-and-Seek property of the random oracle.

Theorem 4.22 (The RO satisfies HnS^H)**.** *Let $\mathcal{D} : \{\perp\} \to \mathcal{X} \times \mathcal{Z}$ and $\mathcal{A} : \mathcal{Y} \times \mathcal{Z} \to \mathcal{X}$ be HnS^H-adversaries satisfying (4.10) for some $0 < \epsilon < 1$, where $\mathcal{A}$ makes q queries to H. Then for $\mathcal{A}^H$ making quantum queries in general, we have*

$$\mathbf{Adv}_H^{\mathsf{HnS}^H}(\mathcal{D}, \mathcal{A}) \leq 4(q+1)\big(\log |\mathcal{Z}| + \log(1/\epsilon) + 20q\big)\epsilon + 10\epsilon \tag{4.15}$$

In the case where $\mathcal{A}^H$ is restricted to classical queries, we have

$$\mathbf{Adv}_H^{\mathsf{HnS}^H}(\mathcal{D}, \mathcal{A}) \leq 2(q+1)\big(\log |\mathcal{Z}| + \log(1/\epsilon) + 1\big)\epsilon + \epsilon\,. \tag{4.16}$$

To prove the above statement, below, we show that a Hide-and-Seek seeker for any fixed hash function $\mathcal{F} : \mathcal{X} \to \mathcal{Y}$ is a successful adversary that can invert multiple hashes simultaneously.

Lemma 4.23. *Let $\mathcal{F} : \mathcal{X} \to \mathcal{Y}$ be a fixed function, $\mathcal{D} : \{\perp\} \to \mathcal{X} \times \mathcal{Z}$ and $\mathcal{A}$ be any Hide-and-Seek adversaries against $\mathcal{F}$. Then for every $(k, T) \in \mathbb{Z}_{>0} \times \mathbb{R}_{>0}$, for $(x, z) \leftarrow \mathcal{D}$ and for $x_1^u, \ldots, x_k^u \leftarrow \mathcal{Y}$ sampled uniformly and independently, we have*

$$\Pr\left[\, x = \mathcal{A}(\mathcal{F}(x), z) \,\right] \leq T \cdot \mathsf{guess}(x \mid z) + \left(\frac{|\,\mathcal{X}\,|}{T}\right)^k \Pr\left[x_i^u = \mathcal{A}(\mathcal{F}(x_i^u), z)\ \forall i \in [k]\right].$$

Proof. Here, for any $z^\circ \in \mathcal{Z}$, we define the following "weighted set"

$$S_{z^\circ}^* := \left\{ \big(x^\circ, w_{z^\circ}(x^\circ)\big) \;\middle|\; x^\circ \in \mathcal{X} \right\},$$

where each element x° comes with a weight, given by

$$w_{z^\circ}(x^\circ) := \Pr\big[x^\circ = \mathcal{A}(\mathcal{F}(x^\circ), z^\circ)\big]\,.$$

The total weight of $S_{z^\circ}^*$ is defined as $W(S_{z^\circ}^*) := \sum_{x^\circ} w_{z^\circ}(x^\circ)$.

Here, we consider the guesser $\mathcal{G}$ that, on input z, chooses its guess $\hat{x}$ by picking it from $\mathcal{X}$ according to the renormalized weights, i.e., according to the distribution

$$p_z(\hat{x}) := \frac{w_z(\hat{x})}{W(S_z^*)}\,.$$

First, we note that

$$\begin{aligned}\Pr[\hat{x} = x] &\geq \Pr[\hat{x} = x \wedge W(S_z^*) \leq T] \\ &\geq \frac{1}{T} \Pr[\mathcal{A}(\mathcal{F}(x), z) = x \wedge W(S_z^*) \leq T] \\ &\geq \frac{1}{T} \big(\Pr[\mathcal{A}(\mathcal{F}(x), z) = x] - \Pr[W(S_z^*) > T] \big) ,\end{aligned}$$

where here, for the second inequality, we exploit that for any fixed choices of x and z, if $W(S_z^*) \leq T$ then $\Pr[\hat{x} = x] = p_z(x) \geq w_z(x)/T$, and so the inequality is obtained by averaging over these choices. Rearranging the terms, we have

$$\mathbf{Adv}^{\mathsf{HnS}^H}(\mathcal{D}, \mathcal{A}) \leq T \cdot \mathsf{guess}(x \mid z) + \Pr[W(S_z^*) > T] . \tag{4.17}$$

Observe that, for every $k \in \mathbb{Z}_{>0}$ and for $x^u, x_1^u, \ldots, x_k^u \leftarrow \mathcal{X}$ sampled uniformly and independently,

$$\left(\frac{W(S_{z^\circ}^*)}{|\mathcal{X}|} \right)^k = \left(\Pr\left[x^u = \mathcal{A}(\mathcal{F}(x^u), z^\circ) \right] \right)^k = \Pr\left[x_i^u = \mathcal{A}(\mathcal{F}(x_i^u), z^\circ) \; \forall i \in [k] \right] . \tag{4.18}$$

Hence, by averaging over $z^\circ \sim z$, for every $T \in \mathbb{R}_{>0}$, we immediately obtain

$$\Pr\left[W(S_z^*) > T \right] \leq \frac{\mathbb{E}\left[|\, W(S_z^*) \,|^k \right]}{T^k} \leq \frac{\Pr\left[x_i^u = \mathcal{A}(\mathcal{F}(x_i^u), z) \; \forall i \in [k] \right]}{T^k} ,$$

where the first inequality follows Markov's bound. Plugging the above back to (4.17), we conclude the proof. □

Lemma 4.24. *Let $\mathcal{F} : \mathcal{X} \to \mathcal{Y}$ be a fixed function, and $\mathcal{A}$ be any algorithm. Then, for $(x^u, y^u) \leftarrow \mathcal{X} \times \mathcal{Y}$ sampled uniformly,*

$$\Pr\left[x^u = \mathcal{A}(\mathcal{F}(x^u)) \right] \leq \frac{|\, \mathcal{Y} \,| \cdot \Pr\left[y^u = \mathcal{F}\left(\mathcal{A}(y^u)\right) \right]}{|\, \mathcal{X} \,|} .$$

Proof. Without loss of generality, let $\mathcal{A}$ be deterministic. Observe that, by symmetry, for every $y^\circ \in \mathcal{F}(\mathcal{X}) \subseteq \mathcal{Y}$,

$$\begin{aligned}\Pr\left[x^u = \mathcal{A}(y^\circ) \;\middle|\; \mathcal{F}(x^u) = y^\circ \right] &= \begin{cases} 1/\left|\, \mathcal{F}^{-1}(y^\circ) \,\right| & \text{if } \mathcal{F}(\mathcal{A}(y^\circ)) = y^\circ \\ 0 & \text{otherwise} \end{cases} \\ &= \frac{\mathbb{1}_{\mathcal{F}(\mathcal{A}(y^\circ)) = y^\circ}}{|\, \mathcal{F}^{-1}(y^\circ) \,|} .\end{aligned}$$

Averaging both sides over $y^\circ \sim \mathcal{F}(x^u)$, we obtain

$$\begin{aligned}
\Pr\left[\, x^u = \mathcal{A}(\mathcal{F}(x^u)) \,\right] &= \sum_{y^\circ \in \mathcal{F}(\mathcal{X})} \Pr\left[\, \mathcal{F}(x^u) = y^\circ \,\right] \cdot \frac{\mathbb{1}_{\mathcal{F}(\mathcal{A}(y^\circ))=y^\circ}}{|\, \mathcal{F}^{-1}(y^\circ) \,|} \\
&= \sum_{y^\circ \in \mathcal{F}(\mathcal{X})} \frac{\left|\, \mathcal{F}^{-1}(y^\circ) \,\right|}{|\, \mathcal{X} \,|} \cdot \frac{\mathbb{1}_{\mathcal{F}(\mathcal{A}(y^\circ))=y^\circ}}{|\, \mathcal{F}^{-1}(y^\circ) \,|} \\
&\leq \frac{|\, \mathcal{Y} \,|}{|\, \mathcal{X} \,|} \sum_{y^\circ \in \mathcal{Y}} \frac{\mathbb{1}_{\mathcal{F}(\mathcal{A}(y^\circ))=y^\circ}}{|\, \mathcal{Y} \,|} \\
&= \frac{|\, \mathcal{Y} \,| \cdot \Pr\left[y^u = \mathcal{F}\left(\mathcal{A}(y^u)\right)\right]}{|\, \mathcal{X} \,|} \,.
\end{aligned}$$

This concludes the proof. □

Substituting $\mathcal{F}$ with $\mathcal{F}^k(x_1, \ldots, x_k) := \big(\mathcal{F}(x_1), \ldots, \mathcal{F}(x_k)\big)$, and $\mathcal{A}$ with $(y_1, \ldots, y_k) \mapsto \big(\mathcal{A}(y_1, z), \ldots, \mathcal{A}(y_k, z)\big)$ for z as in the right-hand-side of Lemma 4.23, we immediately obtain the following.

Corollary 4.25. *Let $\mathcal{F} : \mathcal{X} \to \mathcal{Y}$ be a fixed function, $\mathcal{D} : \{\bot\} \to \mathcal{X} \times \mathcal{Z}$ and $\mathcal{A}$ be any Hide-and-Seek adversaries against $\mathcal{F}$. Then for every $(k, T) \in \mathbb{Z}_{>0} \times \mathbb{R}_{>0}$, for $(x, z) \leftarrow \mathcal{D}$ and for $y_1^u, \ldots, y_k^u \leftarrow \mathcal{Y}$ sampled uniformly and independently, we have*

$$\Pr\left[\, x = \mathcal{A}(\mathcal{F}(x), z) \,\right] \leq T \cdot \mathsf{guess}(x \mid z) + \left(\frac{|\, \mathcal{Y} \,|}{T}\right)^k \Pr\left[y_i^u = \mathcal{F}\big(\mathcal{A}(y_i^u, z)\big)\ \forall i \in [k]\right] .$$

Finally, it suffices to bound the probability of simultaneously finding RO preimages of multiple uniformly and independently chosen target. We prove it for the classical case here, and refer to [CGLQ20] for the quantum case.[14]

Lemma 4.26. *Let $\mathcal{A}^H$ be an oracle algorithm making at most q classical queries to H. Then for $y_1^u, \ldots, y_k^u \leftarrow \mathcal{Y}$ sampled uniformly and independently, we have*

$$\Pr\left[H\left(\mathcal{A}^H(y_i^u)\right) = y_i^u\ \forall i \in [k]\right] \leq \big(k(q+1)/|\, \mathcal{Y} \,|\big)^k .$$

Proof. Let $\mathcal{A}$ be deterministic, and $\mathcal{B}^H(y_1^u, \ldots, y_k^u)$ be running every instance of $x_i \leftarrow \mathcal{A}^H(y_i^u)$, making one more query per instance to check if the produced output is valid $H(x_i) \stackrel{?}{=} y_i^u$ and output a bit that is 1 if and only if all checks are passed. Then, of course

$$\Pr\left[H\left(\mathcal{A}^H(y_i^u)\right) = y_i^u\ \forall i \in [k]\right] = \Pr\left[\mathcal{B}^H(y_1^u, \ldots, y_k^u) = 1\right] .$$

[14] Strictly speaking, since we cannot fully extract and verify the bound as in [CGLQ20], we use a bound that we obtain by using a similar approach.

Toward bounding the right-hand side, it suffices to consider the case where the domain of H is of size at least $k(q+1)$ and $\mathcal{A}$ makes exactly $k(q+1)$ distinct queries $x_1, \ldots, x_{k(q+1)}$ to H. Observe that $y_1^u, \ldots, y_k^u, H(x_1), \ldots, H(x_{k(q+1)})$ are mutually independent, and hence we have

$$\begin{aligned}
&\Pr[\mathcal{B}^H(y_1^u, \ldots, y_k^u) = 1] \leq \Pr\left[y_i^u \in \{H(x_1), \ldots, H(x_{k(q+1)})\}\ \forall i \in [k]\right] \\
&= \mathop{\mathbb{E}}_{\substack{y_i^\circ \sim H(x_i) \\ \forall i \in [k]}} \left[\Pr\left[y_i^u \in \{y_1^\circ, \ldots, y_{k(q+1)}^\circ\}\ \forall i \in [k] \;\middle|\; H(x_j) = y_j^\circ\ \forall j \in [k(q+1)]\right]\right] \\
&= \mathop{\mathbb{E}}_{\substack{y_i^\circ \sim H(x_i) \\ \forall i \in [k]}} \left[\prod_{i \in [k]} \Pr\left[y_i^u \in \{y_1^\circ, \ldots, y_{k(q+1)}^\circ\} \;\middle|\; H(x_j) = y_j^\circ\ \forall j \in [k(q+1)]\right]\right] \\
&\leq \mathop{\mathbb{E}}_{\substack{y_i^\circ \sim H(x_i) \\ \forall i \in [k]}} \left[\prod_{i \in [k]} k(q+1)/|\,\mathcal{Y}\,|\right] \leq \left(k(q+1)/|\,\mathcal{Y}\,|\right)^k .
\end{aligned}$$

□

Lemma 4.27 (A concrete version of [CGLQ20, Lemma 5.6])**.** *Let $\mathcal{A}^H$ be an oracle algorithm making at most q quantum queries to H. Then for $y_1^u, \ldots, y_k^u \leftarrow \mathcal{Y}$ sampled uniformly and independently, we have*

$$\Pr\left[H\left(\mathcal{A}^H(y_i^u)\right) = y_i^u\ \forall i \in [k]\right] \leq \left(\frac{2k(q+1) + 40q^2 + 4}{|\,\mathcal{Y}\,|}\right)^k .$$

Now we are ready to wrap up the proof of Theorem 4.22

Proof of Theorem 4.22. For every fixed choice H° of H, evoke Corollary 4.25 and we obtain

$$\begin{aligned}
&\Pr\left[x = \mathcal{A}^{H^\circ}(\mathcal{F}^{H^\circ}(x), z) \;\middle|\; H = H^\circ\right] \\
&\leq T \cdot \mathsf{guess}(x \mid z, H = H^\circ) + \left(\frac{|\,\mathcal{Y}\,|}{T}\right)^k \Pr\left[y_i^u = \mathcal{A}^{H^\circ}(y_i^u, z)\ \forall i \in [k] \;\middle|\; H = H^\circ\right] \\
&\leq T \cdot \mathsf{guess}(x \mid z, H = H^\circ) + \left(\frac{|\,\mathcal{Y}\,|}{T}\right)^k \sum_{z^\circ \in \mathcal{Z}} \Pr\left[y_i^u = \mathcal{A}^{H^\circ}(y_i^u, z^\circ)\ \forall i \in [k] \;\middle|\; H = H^\circ\right] .
\end{aligned}$$

Note that $\mathsf{guess}(x \mid z, H) \leq \epsilon$; averaging the above over $H^\circ \sim H$, we thus obtain

$$\Pr\left[\, x = \mathcal{A}^H(\mathcal{F}^H(x), z)\,\right] \leq T\epsilon + \left(\frac{|\,\mathcal{Y}\,|}{T}\right)^k \sum_{z^\circ \in \mathcal{Z}} \Pr\left[y_i^u = \mathcal{A}^H(y_i^u, z^\circ)\ \forall i \in [k]\right] \tag{4.19}$$

In the case where $\mathcal{A}$ makes classical queries only, substitute $\mathcal{A}$ as in Lemma 4.26 with $\mathcal{A}(\cdot, z^\circ)$ in (4.19); we obtain

$$\Pr\big[\, x = \mathcal{A}^H(\mathcal{F}^H(x), z)\,\big] \le T\epsilon + |\,\mathcal{Z}\,|\left(\frac{k(q+1)}{T}\right)^k .$$

Plugging in $T = 2k(q+1)$ and $k = \lceil \log|\,\mathcal{Z}\,| + \log(1/\epsilon)\rceil$, the above is bounded by

$$\le T\epsilon + |\,\mathcal{Z}\,| \cdot 2^{-k} \le T\epsilon + \epsilon \le 2\big(\log|\,\mathcal{Z}\,| + \log(1/\epsilon) + 1\big)(q+1) + \epsilon .$$

Similarly, in the case where $\mathcal{A}$ makes quantum queries in general, substitute $\mathcal{A}$ as in Lemma 4.27 with $\mathcal{A}(\cdot, z^\circ)$ in (4.19); we obtain

$$\Pr\big[\, x = \mathcal{A}^H(\mathcal{F}^H(x), z)\,\big] \le T\epsilon + |\,\mathcal{Z}\,|\left(\frac{2k(q+1) + 40q^2 + 4}{T}\right)^k .$$

Plugging in $T = 2(2k(q+1) + 40q^2 + 4)$ and $k = \lceil \log|\,\mathcal{Z}\,| + \log(1/\epsilon)\rceil$ and simplifying the bound, the above is bounded by

$$\le T\epsilon + |\,\mathcal{Z}\,|\,2^{-k} \le T\epsilon + \epsilon \le 4(q+1)\big(\log|\,\mathcal{Z}\,| + \log(1/\epsilon) + 20q\big)\epsilon + 10\epsilon$$

This concludes the proof. □

4.5.6 Wrapping up the Proof of Theorem 4.18

Theorem 4.18. *Let $\mathcal{D}^H : \mathcal{SK} \to \mathcal{M}$ and $\mathcal{A}^H : \mathcal{SK} \times \mathcal{SGN} \times \mathcal{AUX} \to \mathcal{PK} \times \mathcal{SGN}$ be $\mathsf{sNR}^{H,\perp}$-adversaries against $\mathsf{BUFF}[\mathcal{S}, H]$ for some $\mathsf{aux} : \mathcal{SK} \times \mathcal{M} \to \mathcal{AUX}$, making at most q_D and q_A queries to H, respectively, where (4.9) is satisfied for h such that $0 < \epsilon := 2^{-h} \le \frac{1}{2}$. Then*

$$\mathbf{Adv}^{\mathsf{sNR}^{H,\perp}}_{\mathsf{BUFF}[\mathcal{S},H]}(\mathcal{D}, \mathcal{A}, \mathsf{aux}) \le 6(q_A + q_S + 1)^2 \log\left(\frac{|\mathcal{SK}| \cdot |\mathcal{AUX}|}{\epsilon}\right)\epsilon + q_K\epsilon + \frac{q_D + 1}{|\mathcal{Y}|}$$

and in the case where $\mathcal{A}$ makes quantum queries, we have $\mathbf{Adv}^{\mathsf{sNR}^{H,\perp}}_{\mathsf{BUFF}[\mathcal{S},H]}(\mathcal{D}, \mathcal{A}, \mathsf{aux})$

$$\le 18\sqrt{\left(\log\frac{|\mathcal{SK}| \cdot |\mathcal{AUX}|}{\epsilon} + q_{\mathcal{A}} + q_S\right)(q_{\mathcal{A}} + q_S + 1)^3\,\epsilon} + q_K\epsilon + \frac{q_D + 1}{|\mathcal{Y}|} .$$

Proof. In the case where $\mathcal{A}$ is restricted to classical queries, we combine (4.12)

and (4.16), simplify the bound, and get

$$\begin{aligned}
&\mathbf{Adv}^{\mathsf{sNR}^{H,\perp}}_{\mathsf{BUFF}[\mathcal{S},H]}(\mathcal{D},\mathcal{A},\mathsf{aux}) \\
&\leq (q_A+q_S)\cdot\mathbf{Adv}^{\mathsf{HnS}^H}_{H}(\bar{\mathcal{D}},\bar{\mathcal{A}})+q_K\epsilon+\frac{q_D+1}{|\mathcal{Y}|} \\
&\leq 6(q_A+q_S+1)^2\log\left(\frac{|\mathcal{SK}|\cdot|\mathcal{AUX}|}{\epsilon}\right)\epsilon+q_K\epsilon+\frac{q_D+1}{|\mathcal{Y}|}
\end{aligned}$$

where $\bar{\mathcal{D}}$ and $\bar{\mathcal{A}}$ are as specified in Lemma 4.20. Similarly, in the case where $\mathcal{A}$ is allowed to make quantum queries, we combine (4.13) and (4.15), simplify the bound, and get

$$\begin{aligned}
&\mathbf{Adv}^{\mathsf{sNR}^{H,\perp}}_{\mathsf{BUFF}[\mathcal{S},H]}(\mathcal{D},\mathcal{A},\mathsf{aux}) \\
&\leq 2(q_A+q_S)\cdot\sqrt{\mathbf{Adv}^{\mathsf{HnS}^H}(\bar{\mathcal{D}},\bar{\mathcal{A}})}+q_K\epsilon+\frac{q_D+1}{|\mathcal{Y}|} \\
&\leq 18\sqrt{\left(\log\frac{|\mathcal{SK}|\cdot|\mathcal{AUX}|}{\epsilon}+q_{\mathcal{A}}+q_S\right)(q_{\mathcal{A}}+q_S+1)^3\epsilon}+q_K\epsilon+\frac{q_D+1}{|\mathcal{Y}|}
\end{aligned}$$

where $\bar{\mathcal{D}}$ and $\bar{\mathcal{A}}$ are as specified in Lemma 4.20. □

4.6 BUFF via Iterative Hash Functions

4.6.1 sNR Relative to (Function-like) Oracles

We recall the *secret-key non-resignability (sNR)* defined in Section 4.5.1, slightly relaxed by not restricting to the random oracle model, but instead considering an oracle O that is to be instantiated with a function chosen according to an *arbitrary* distribution. The relevant examples are when $O = H$ is a uniformly random function with a given domain and range, for capturing the random oracle model, or $O = P^{\pm 1} : \{-1,1\}\times\mathcal{Y}\to\mathcal{Y}$, which maps $(d,x)\mapsto P^d(x)$ for a random permutation $P\leftarrow\mathsf{Sym}(\mathcal{Y})$, capturing the random-permutation model then. We may then write $\mathcal{A}^O$ in order to make the oracle access to O explicit.

Let $\mathcal{S} = (\mathsf{KGen}^O, \mathsf{Sign}^O, \mathsf{Vrfy}^O)$ be a signature scheme where the underlying algorithms are given query access to O. The security game $\mathsf{sNR}^{O,\perp}$ depicted in Fig. 4.15 is played by two oracle algorithms

$$\mathcal{D}^O : \mathcal{SK}\to\mathcal{M}\ ,\ \text{and}\quad \mathcal{A}^H : \mathcal{SK}\times\mathcal{SGN}\times\mathcal{AUX}\to\mathcal{PK}\times\mathcal{SGN}$$

each with bounded queries to O, and an arbitrary function $\mathsf{aux} : \mathcal{SK}\times\mathcal{M}\to\mathcal{AUX}$. It is crucial here that aux has no access to O; otherwise, the strong negative result from [DFHS24] applies.

$\mathsf{sNR}^{O,\perp}_{\mathcal{S}}(\mathcal{D},\mathcal{A},\mathsf{aux})$:
1: $(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KGen}^O$
2: $m \leftarrow \mathcal{D}^O(\mathsf{sk})$
3: $(\mathsf{pk}',\sigma') \leftarrow \mathcal{A}^O\left(\mathsf{sk}, \mathsf{Sign}^O(\mathsf{sk},m), \mathsf{aux}(m,\mathsf{sk})\right)$
4: **return** $\mathsf{pk}' \neq \mathsf{pk} \ \wedge \ \mathsf{Vrfy}^O(\mathsf{pk}',m,\sigma')$

Figure 4.15: The game $\mathsf{sNR}^{O,\perp}$.

In addition, for the game to be non-trivial, we have the entropy requirement

$$\mathsf{H}_\infty(m \mid \mathsf{sk}, \mathsf{aux}(m,\mathsf{sk}), O) \geq \log(1/\epsilon) \tag{4.20}$$

for some small $\epsilon > 0$ is satisfied. We informally say that the signature scheme $\mathcal{S}$ satisfies $\mathsf{sNR}^{O,\perp}$ if for every such $\mathcal{D},\mathcal{A},\mathsf{aux}$

$$\mathbf{Adv}^{\mathsf{sNR}^{O,\perp}}_{\mathcal{S}}(\mathcal{D},\mathcal{A},\mathsf{aux}) := \Pr\left[\mathsf{sNR}^{O,\perp}_{\mathcal{S}}(\mathcal{D},\mathcal{A},\mathsf{aux}) = 1\right]$$

is small. Later in Section 4.8, we will consider a computational variant of the entropy condition (4.20).

4.6.2 Iterative Hash Functions

Here and for the remainder, we set $\mathcal{X} := \{0,1\}^r$ and $\mathcal{Y} := \{0,1\}^{n_f}$ for parameters r and n_f, and let $f : \mathcal{X} \times \mathcal{Y} \to \mathcal{Y}$ be a function, referred to as the *round function*. An element $x \in \mathcal{X}$ is called a *block*, and a bit string in $x \in \{0,1\}^{rB}$ is said to have *block length* B. Clearly, a bit string x with block length B can be naturally parsed as $(x_1,\ldots,x_B) \in (\{0,1\}^r)^B$. Finally, we write $\mathcal{X}^{\leq B}$ for the set of non-empty strings with block length at most B.
For a bit string s, define

$$|\, s \,|_{\mathrm{bl}} := \lceil\, |\, s \,| \,/\, r \,\rceil,$$

as the number of blocks of length r that s takes up, rounded up when the last block is not full.

The following captures the general notion of an iterative hash function, to which our negative result applies.

Definition 4.28. *An* iterative hash function $\mathcal{F}\colon \{0,1\}^* \to \{0,1\}^{n_{\mathcal{F}}}$ *that is specified by a round function* $f : \mathcal{X} \times \mathcal{Y} \to \mathcal{Y}$*, an initialization vector* $\mathsf{IV} \in \mathcal{Y}$*, a padding function* $\mathsf{pad} : \{0,1\}^* \to \{0,1\}^{\leq 2r}$ *with the property that* $|x|+|pad(x)|$ *is a multiple of* r *for any* $x \in \{0,1\}^*$*, and (optionally) a post-processing function* $p : \mathcal{Y} \to \{0,1\}^{n_{\mathcal{F}}}$*.* $\mathcal{F}$ *is then defined to map* $x \in \{0,1\}^*$ *to* $\mathcal{F}(x) = p(z_B)$*, where*

z_i is iteratively given by

$$z_i = f(x_i, z_{i-1}) \qquad \text{and} \qquad z_0 \coloneqq \mathsf{IV}$$

and the x_i's are obtained by parsing the string $x \| \mathsf{pad}(x) \in \{0,1\}^{rB}$ naturally as $(x_1, \ldots, x_B) \in (\{0,1\}^r)^B$.

Remark 4.29. *We allow the round function f and the post-processing function p to be given as oracle algorithms, with access to an oracle O (for instance the random oracle, or a random permutation); we will then write f^O and p^O, as well as $\mathcal{F}^O$, to make this explicit. This does not apply to* pad *and* IV*.*

Instantiating the round function f by a cryptographic compression function (which is then typically modelled as a random oracle H), and a suitable padding function pad, we obtain the the Merkle-Damgård construction, which we will be referring to as $\mathsf{MD}^H_{\mathsf{pad}} : \{0,1\}^* \to \mathcal{Y}$ for short.

If we instantiate the round function as $f(x, y) = P\big((x \| 0^c) \oplus y\big)$ for $c = n_f - r$, where $P \in \mathsf{Sym}(\mathcal{Y})$ is a cryptographic permutation (typically modelled as a random permutation), and take a suitable padding function pad (that appends $10 \ldots 01$ with the appropriate number of 0s) and an appropriate post-processing, we recover the Sponge construction $\mathsf{SPNG}^{P^{\pm 1}}_{\mathsf{pad}} : \{0,1\}^* \to \{0,1\}^{n_{\mathsf{SPNG}}}$ with rate r and capacity c. We are considering the Sponge hash function with one squeezing round; this means that $p(z)$ merely outputs the first n_{SPNG} bits of z where for technical reasons we require that $n_{\mathsf{SPNG}} \leq c$.

In the case where no padding is performed and instead the domain is restricted an exact multiple of blocks, we denote the corresponding padding-free variants as $\mathsf{MD}^H_{\perp} : \mathcal{X}^* \to \mathcal{Y}$ and $\mathsf{SPNG}^{P^{\pm 1}}_{\perp} : \mathcal{X}^* \to \{0,1\}^{n_{\mathsf{SPNG}}}$ respectively.

4.6.3 BUFF via Iterative Hash Functions is not sNR

Let $\mathcal{F}^O$ be an iterative hash function, as in Definition 4.28, where the round function f^O has access to an oracle O (which is instantiated by a function chosen according to a given distribution, e.g. the random oracle). Furthermore, let $\mathcal{S}^O = (\mathsf{KGen}, \mathsf{Sign}^O, \mathsf{Vrfy}^O)$ be a signature scheme, where signing and verifying (but not KGen) may also have query access to O.

Proposition 4.30. *There are $\mathsf{sNR}^{O,\perp}$ adversaries $\mathcal{D}^O, \mathcal{A}^O$ and a function* aux *such that for every choice of the security parameter $\lambda \in \mathbb{Z}_{>0}$ we have*

$$\mathsf{H}_\infty\limits_{\substack{(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KGen}^O \\ m \leftarrow \mathcal{D}^O}} \Big(m \;\Big|\; \mathsf{sk}, \mathsf{aux}(m, \mathsf{sk}), O \Big) \geq (\lambda - 2) \cdot r - n_f \; . \tag{4.21}$$

yet

$$\mathbf{Adv}^{\mathsf{sNR}^{O,\perp}}_{\mathsf{BUFF}[\mathcal{S},\mathcal{F}]}(\mathcal{D}, \mathcal{A}, \mathsf{aux}) \geq 1 - \mathsf{guess}(\mathsf{pk}) \; ,$$

for $(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KGen}$. *Moreover the algorithms* $\mathcal{D}$ *and* $\mathcal{A}$ *make* $Q_{\mathcal{D}}$ *and* $Q_{\mathcal{A}}$ *queries to* O *where*

$$Q_{\mathcal{D}} + Q_{\mathcal{A}} \leq Q_{\mathcal{F}} + Q_S \ ,$$

where Sign *makes at most* Q_S *queries to* O*, and evaluating* $\mathcal{F}^O(m\|\mathsf{pk})$ *for* $(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KGen}$ *and* $m \leftarrow \mathcal{D}^O(\mathsf{sk})$ *takes at most* $Q_{\mathcal{F}}$ *queries to* O *in the worst case. If the round function* f *and post-processing* p *of the hash function* $\mathcal{F}$ *are polynomial-time computable, then so is* aux *and* $\mathcal{D},\mathcal{A}$ *are PPT.*

Proof. For simplicity, we give the proof for the case that the padding only depends on the length of the input, i.e. $\mathsf{pad}(x) = \mathsf{pad}(y)$ for all x, y with $|\,x\,| = |\,y\,|$.

We describe the adversaries $\mathcal{D}^O, \mathcal{A}^O$ and the auxiliary function aux. $\mathcal{D}^O$ first samples a message prefix $x = (x_1, \ldots, x_\lambda) \leftarrow \mathcal{X}^\lambda$ and then computes the intermediate digests z_i towards computing the hash of x, i.e., it sets $z_0 = \mathsf{IV}$ and $z_i = f^O(x_i, z_{i-1})$ for $i = 1, \ldots, \lambda$. In the end, it outputs the message $m := x\|z_\lambda$. Furthermore, for any message and secret key, the auxiliary function aux is defined to output the last n_f bits of the message, so that here $\mathsf{aux}(m,\mathsf{sk}) = z_\lambda$, which is sufficient information to compute that hash $\mathcal{F}^O(m\|\mathsf{pk}')$ for any pk'.

Thus, the adversary $\mathcal{A}^O(\mathsf{sk}, \sigma, \mathsf{aux}(m,\mathsf{sk}))$ simply parses $\mathsf{aux}(m,\mathsf{sk}) = z_\lambda$ and samples $(\mathsf{sk}',\mathsf{pk}') \leftarrow \mathsf{KGen}$ and aborts if $\mathsf{pk} = \mathsf{pk}'$. Then it computes the hash $y' = \mathcal{F}^O(m\|\mathsf{pk}')$ in the obvious way: First, it splits $\mathsf{pk}'\|\mathsf{pad}(m\|\mathsf{pk}')$ into blocks of length r.[15] Denote the number of blocks by ℓ and the ith block by b_i. $\mathcal{A}$ then sets $z'_0 = z_\lambda$ and computes $z'_i = f(b_i, z'_{i-1})$ for $i = 1, \ldots, \ell$. It sets $y' = z'_\ell$ (or in case of post-processing $y' = p(z'_\ell)$). Finally, it computes the signature $\sigma' = (\mathcal{S}.\mathsf{Sign}^O(\mathsf{sk}', y'), y')$ and outputs (pk', σ') to the challenger.

For the reader's convenience, we show how the adversaries $\mathcal{D}$ and $\mathcal{A}$ share the computation of the hash in Fig. 4.16.

It is easy to see that $\mathcal{D}^O$ calls the round function f λ times to compute the value z_λ, and $\mathcal{A}$ makes at most $\lceil(|\,\mathsf{pk}\,| + |\,\mathsf{pad}\,|)/r\rceil$ calls the round function and one call to the post-processing p to evaluate the rest of the function.

It is easy to see that all algorithms make at most one call to KGen (expected) and one call to Sign each and thus are efficient.

The min-entropy H_∞ can be computed by

$$\begin{aligned}
&\mathop{\mathsf{H}_\infty}_{\substack{(\mathsf{sk},\mathsf{pk})\leftarrow\mathsf{KGen}^O\\ m\leftarrow\mathcal{D}^O}} \Big(m \;\Big|\; \mathsf{sk}, \mathsf{aux}(m,\mathsf{sk}), O\Big) = \mathop{\mathsf{H}_\infty}_{\substack{(\mathsf{sk},\mathsf{pk})\leftarrow\mathsf{KGen}^O\\ m\leftarrow\mathcal{D}^O}} \big(x\|z_\lambda \big| \mathsf{sk}, z_\lambda, O\big) \\
\geq &\mathop{\mathsf{H}_\infty}_{\substack{(\mathsf{sk},\mathsf{pk})\leftarrow\mathsf{KGen}^O\\ m\leftarrow\mathcal{D}^O}} \big(x\big|O\big) - \big|\,\mathsf{aux}(m,\mathsf{sk})\|\mathsf{pad}(m\|\mathsf{pk}')\,\big| \geq k\cdot r - (n_f + 2r)
\end{aligned}$$

This proves the claim. □

[15] Here we use the assumption that $\mathsf{pad}(m\|\mathsf{pk}')$ can be computed from $|\,m\|\mathsf{pk}'\,|$ which is known to $\mathcal{A}$.

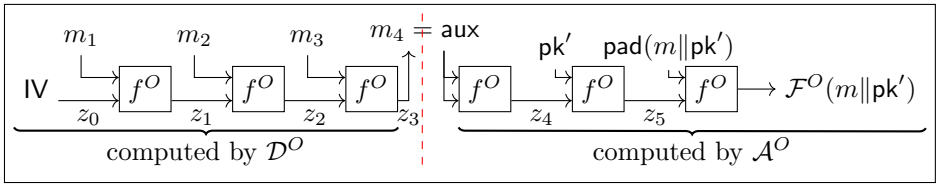


Figure 4.16: A simplified description of our attack for $\lambda = 3$ and without post-processing.

4.7 Sandwich BUFF (sBUFF) Transformation

Motivated by the above attack against the original BUFF transform, we introduce here Sandwich BUFF (sBUFF). The main technical challenge is to show that when instantiated with the Merkle-Damgård hash function, the Sandwich BUFF transform satisfies sNR. Motivated by the approach in [DFH+24], we first reduce the sNR property to the hardness of Hide-and-Seek for the considered hash function, and then we show the hardness of Hide-and-Seek in Sect Section 4.7.3.

For a signature scheme $\mathcal{S} = (\mathsf{KGen}^O, \mathsf{Sign}^O, \mathsf{Vrfy}^O)$ we define the Sandwich BUFF transform of $\mathcal{S}$ with hash function $\mathcal{F}^O$ to be signature scheme $\mathsf{sBUFF}[\mathcal{S}, \mathcal{F}] = (\mathsf{KGen}'^O, \mathsf{Sign}'^O, \mathsf{Vrfy}'^O)$ with algorithms as follows:

KGen'^O:	$\mathsf{Sign}'^O(\mathsf{sk}, m)$:	$\mathsf{Vrfy}'^O(\mathsf{pk}, m, \sigma')$:
1: $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}^O$	1: $y := \mathcal{F}^O(m\|\mathsf{pk}\|m)$	1: $(\sigma, y) := \sigma'$
2: **return** $(\mathsf{sk}, \mathsf{pk})$	2: $\sigma \leftarrow \mathsf{Sign}^O(\mathsf{sk}, y)$	2: **return** $\mathsf{Vrfy}^O(\mathsf{pk}, y, \sigma) \wedge$
	3: **return** (σ, y)	3: $\quad y = \mathcal{F}^O(m\|\mathsf{pk}\|m)$

Figure 4.17: The Sandwich BUFF Transformation

4.7.1 Sandwich BUFF via Merkle-Damgård is sNR

Recall that $\mathcal{X} := \{0,1\}^r$ and $\mathcal{Y} := \{0,1\}^{n_f}$. Towards, the main statement for $\mathsf{sBUFF}[\cdot, \mathsf{MD}]$, let $\mathcal{S} = (\mathsf{KGen}, \mathsf{Sign}^H, \mathsf{Vrfy}^H)$ be a signature scheme in the random oracle model, i.e., with access to a random oracle $H : \mathcal{X} \times \mathcal{Y} \to \mathcal{Y}$ (though, for simplicity, we assume that KGen does not query H), and let $\mathsf{MD}_\perp^H$ be the padding-free Merkle-Damgård hash function with H as the round-function. We allow Sign to make at most $q_\mathcal{S} \in \mathbb{Z}_{>0}$ queries for signing each message. For technical reasons, we restrict the domain of $\mathsf{MD}_\perp^H$ to $\mathcal{X}^{\leq \bar{B}}$ for an arbitrary but fixed bound $\bar{B}$. As a consequence, the message space $\mathcal{M}'$ of $\mathcal{S}' = \mathsf{sBUFF}[\mathcal{S}, \mathsf{MD}]$ is then restricted so that

$$m\|\mathsf{pk}\|m\|\mathsf{pad}(m\|\mathsf{pk}\|m) \in \mathcal{X}^{\leq \bar{B}}$$

for all $m \in \mathcal{M}'$ and $\mathsf{pk} \in \mathcal{PK}' = \mathcal{PK}$. Additionally, we assume (without loss of generality) that any $\mathsf{pk} \in \mathcal{PK}'$ is at least r bits long. We then have the following statement on the sNR property of $\mathcal{S}' = \mathsf{sBUFF}[\mathcal{S}, \mathsf{MD}]$.

Theorem 4.31. *Let $\mathcal{D}^H$ and $\mathcal{A}^H$ be $\mathsf{sNR}^{H,\perp}$-adversaries against $\mathsf{sBUFF}[\mathcal{S}, \mathsf{MD}]$ making at most $q_{\mathcal{D}}$ and $q_{\mathcal{A}} > 0$ classical queries to H respectively such that (4.10) holds for some $0 < \epsilon < 1$; let $\mathsf{aux} : \mathcal{SK} \times \mathcal{M} \to \mathcal{AUX}$ be any (possibly randomized) function. Then for $k := \lceil \log |\, \mathcal{SK} \times \mathcal{AUX} \,| + \log(1/\epsilon) \rceil$ and $q_{\mathcal{B}} := q_{\mathcal{A}} + q_{\mathcal{S}}$ we have*

$$\mathbf{Adv}^{\mathsf{sNR}^{H,\perp}}_{\mathsf{sBUFF}[\mathcal{S},\mathsf{MD}_{\mathsf{pad}}]}(\mathcal{D}, \mathcal{A}) \leq \frac{(q_{\mathcal{D}}^2 \cdot r + 1) \cdot \bar{B}\bar{L} + q_{\mathcal{D}} + 2\bar{L}^2}{|\, \mathcal{Y} \,|} + 6kr^2 q_{\mathcal{B}}(q_{\mathcal{B}} + \bar{B})\epsilon \ ,$$

where $\bar{L} = q_{\mathcal{A}} + q_{\mathcal{D}} + q_{\mathcal{S}} + 2\bar{B}$.

Proof. We put together Lemma 4.32, which reduces sNR of Sandwich BUFF to the harness of Hide-and-Seek, and Theorem 4.37 (with $\mathcal{Z} = \mathcal{SK} \times \mathcal{AUX}$ and $q = q_{\mathcal{A}}$), which shows the hardness of Hide-and-Seek, to obtain the bound. □

4.7.2 Reducing sNR of Sandwich BUFF to Hide-and-Seek

We reduce the sNR property for Sandwich BUFF with MD to the hardness of Hide-and-Seek for MD.

Lemma 4.32. *Let $\mathcal{D}^H$ and $\mathcal{A}^H$ be $\mathsf{sNR}^{H,\perp}$-adversaries against $\mathsf{sBUFF}[\mathcal{S}, \mathsf{MD}]$, making at most $q_{\mathcal{D}}$ and $q_{\mathcal{A}} \in \mathbb{Z}_{>0}$ classical queries to H respectively; let $\mathsf{aux} : \mathcal{SK} \times \mathcal{M} \to \mathcal{AUX}$ be any (possibly randomized) function. Then there exists a hider $\bar{\mathcal{D}} : \{\perp\} \to \mathcal{X}^{\leq \bar{B}} \times \mathcal{Z}$ and a seeker $\bar{\mathcal{A}} : \mathcal{Y} \times \mathcal{Z} \to \mathcal{X}^{\leq \bar{B}}$ and $\mathcal{Z} = \mathcal{SK} \times \mathcal{AUX}$, where $\bar{\mathcal{A}}$ makes at most $q_{\mathcal{B}} := q_{\mathcal{A}} + q_{\mathcal{S}}$ and $\bar{\mathcal{D}}$ makes at most $q_{\mathcal{D}}$ queries to H, and such that*

$$\mathsf{H}^{\infty}_{(x,z) \leftarrow \bar{\mathcal{D}}^H}(x \mid H, z) = \mathsf{H}^{\infty}_{\substack{(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KGen}^H \\ m \leftarrow \mathcal{D}^H(\mathsf{sk})}}(m \mid H, \mathsf{sk}, \mathsf{aux}(\mathsf{sk}, m)) \tag{4.22}$$

and $\mathbf{Adv}^{\mathsf{sNR}^{H,\perp}}_{\mathsf{sBUFF}[\mathcal{S},\mathsf{MD}]}(\mathcal{D}, \mathcal{A}, \mathsf{aux})$

$$\leq 2q_{\mathcal{B}} \cdot r^2 \cdot \mathbf{Adv}^{\mathsf{HnS}^H}_{\mathsf{MD}_{\perp}}(\bar{\mathcal{D}}, \bar{\mathcal{A}}) + \frac{(q_{\mathcal{D}}^2 \cdot r + 1) \cdot \bar{B}\bar{L} + q_{\mathcal{D}} + 2\bar{L}^2}{|\, \mathcal{Y} \,|}. \tag{4.23}$$

where $\bar{B}$ and $q_{\mathcal{S}}$ are as described in Section 4.7.1 and $\bar{L} = q_{\mathcal{D}} + q_{\mathcal{A}} + q_{\mathcal{S}} + 2\bar{B}$. Moreover, if aux is polynomial-time computable and $\mathcal{D}, \mathcal{A}$ are PPT, then so are $\bar{\mathcal{D}}, \bar{\mathcal{A}}$.

Proof. We present here a slightly simplified proof, where we omit the padding, i.e., consider the padding-free $\mathsf{MD}^H_{\perp}$, and instead assume that the message m

output by $\mathcal{D}$ is a multiple of the block length r of the hash function; furthermore, we assume that all public keys have the same length, also a multiple of r. The full proof can be found in Appendix A.3, and we state state the auxiliary claims in the simplified proof in full generality, but mark in colour the parts that are only relevant in the general case.

First, we note that

$$\mathbf{Adv}^{\mathsf{sNR}^{H,\perp}}_{\mathsf{sBUFF}[\mathcal{S},\mathsf{MD}_\perp]}(\mathcal{D},\mathcal{A},\mathsf{aux}) \le \Pr\left[\mathsf{MD}^H_\perp(m\|\mathsf{pk}'\|m) = y' \wedge \mathsf{pk}' \neq \mathsf{pk}\right]$$

with the random variables $\mathsf{pk}, \mathsf{pk}', m$ and y defined by the experiment

$$(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KGen},\ m \leftarrow \mathcal{D}^H(\mathsf{sk}),\ (\mathsf{pk}',y') \leftarrow \mathcal{B}^H\left(\mathsf{sk}, \mathsf{MD}^H_\perp(m\|\mathsf{pk}\|m), \mathsf{aux}(\mathsf{sk},m)\right)$$

where $\mathcal{B}(\mathsf{sk},y,a) := \mathcal{A}^H\left(\mathsf{sk},(\mathsf{Sign}^H(\mathsf{sk},y),y),a\right)$. We note that the random choice of H is understood and left implicit. We recall that $\mathcal{D}$ and $\mathcal{B}$ make at most $q_\mathcal{D}$ and $q_\mathcal{B} := q_\mathcal{A} + q_\mathcal{S}$ queries to the random oracle respectively. We introduce the following additional random variables, implicitly defined by the above experiment.

We denote by B the block number of $m\|\mathsf{pk}\|m$ and $m\|\mathsf{pk}'\|m$. We denote by B' the block number of $m\|\mathsf{pk}'$. We denote by m_i the ith block of the message m and by $B'' = |\,m\,|_{\mathrm{bl}}$, that is $m = m_1\|\ldots\|m_{B''}$.

We define

$$z'_1 := \mathsf{MD}^H_\perp(m\|\mathsf{pk}) \qquad \text{and} \qquad z'_{i+1} := \mathsf{MD}^H_\perp(m\|\mathsf{pk}\|m_1\|\ldots\|m_i) = H(m_i, z'_i)$$

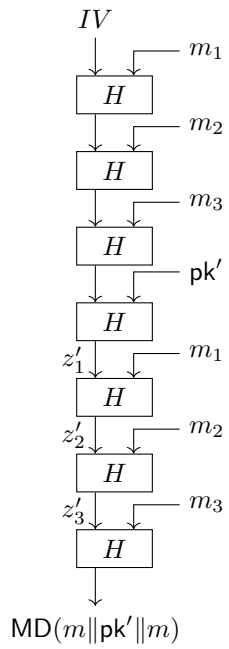
for $i = 1,\ldots,B''$, with $z_{B''+1} = \mathsf{MD}^H_\perp(m\|\mathsf{pk}'\|m)$ then. The z_i's thus form the intermediate digests in the second part of the hash chain when computing $\mathsf{MD}^H_\perp(m\|\mathsf{pk}'\|m)$; for illustration see Fig. 4.18a.

Finally, we let $\tau_1,\ldots,\tau_L$ with $L = q_\mathcal{D} + B + q_\mathcal{B} + B$ be the list of inputs to all the hash computations performed during the experiment, listed in the performed order; see Fig. 4.18b. Hence, $Q_\mathcal{D} = \{\tau_1,\ldots,\tau_{q_\mathcal{D}}\}$ consists of the hash queries made by $\mathcal{D}$, $Q_{\mathsf{MD}_\perp(m\|\mathsf{pk}\|m)} = \{\tau_{q_\mathcal{D}+1},\ldots,\tau_{q_\mathcal{D}+B}\}$ consists of the hash queries made by the challenger when computing $\mathsf{MD}_\perp(m\|\mathsf{pk}\|m)$, $Q_\mathcal{B} = \{\tau_{q_\mathcal{D}+B+1},\ldots,\tau_{q_\mathcal{D}+B+q_\mathcal{B}}\}$ of the queries made by $\mathcal{B}$, and the remaining τ_ℓ's are the inputs to the hash computations done towards computing $\mathsf{MD}^H_\perp(m\|\mathsf{pk}'\|m)$, in particular $\tau_{q_\mathcal{D}+B+q_\mathcal{B}+B'+1} = (m_1, z'_1)$, $\tau_{q_\mathcal{D}+B+q_\mathcal{B}+B'+2} = (m_2, z'_2)$, etc. For any τ_ℓ with $\ell \in [L]$, we write $\mathsf{R}(\ell)$ for the right component of τ_ℓ, i.e., $\mathsf{R}(q_\mathcal{D}+q_\mathcal{B}+B+1) = z'_1$, etc.

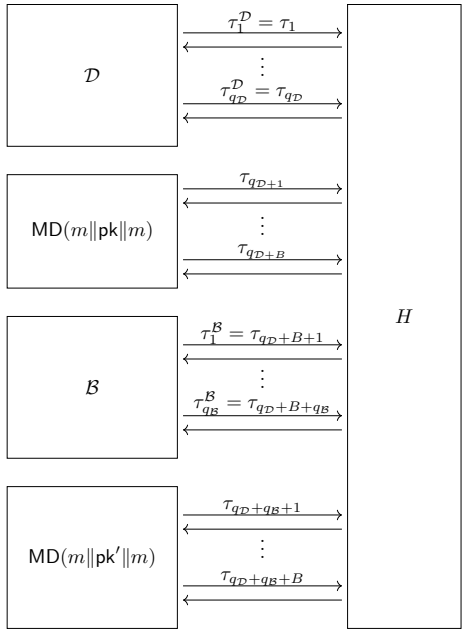
As explained, we are interested in upper-bounding the probability $\Pr[\Sigma]$ of the event

$$\Sigma := \left[\mathsf{MD}^H_\perp(m\|\mathsf{pk}'\|m) = y' \wedge \mathsf{pk}' \neq \mathsf{pk}\right].$$

We do this by introducing a sequence of further events, Γ, Λ and Δ, with the property that $\Pr[\Sigma]$ is close to $\Pr[\Sigma \wedge \Gamma \wedge \Lambda \wedge \Delta]$, assuming $\mathbf{Adv}^{\mathsf{HnS}^H}_{\mathsf{MD}_\perp}(\bar{\mathcal{D}},\bar{\mathcal{A}})$ is



(a) The hash chain for $\mathsf{MD}(m\|\mathsf{pk}'\|m)$ with three message blocks and one pk' block.



(b) Interaction between $\mathcal{D}, \mathcal{B}$, and the random oracle H, along with the queries made by the game in the end when computing the MD hash for the Sandwich BUFF transform.

Figure 4.18: Our notation for the proof of Lemma 4.32

small (for suitable choices of $\bar{\mathcal{D}}$ and $\bar{\mathcal{A}}$), and such that we can upper bound the latter probability.

We start off avoiding some atypical behavior of H. Formally, we consider the good event $\Gamma := \Gamma_1 \wedge \Gamma_2 \wedge \Gamma_3$ with

$$\Gamma_1 := \left[\forall\, \ell, \ell' \in [L] : H(\tau_\ell) = H(\tau_{\ell'}) \;\Rightarrow\; \tau_\ell = \tau_{\ell'}\right]$$
$$\Gamma_2 := \left[\forall\, \ell, \ell' \in [L] : H(\tau_\ell) = \mathsf{R}(\ell') \;\Rightarrow\; (\exists\, \ell_\circ < \ell' : \tau_\ell = \tau_{\ell_\circ})\right] \qquad \text{and}$$
$$\Gamma_3 := \left[\forall\, \ell \in [q_{\mathcal{D}}] : H(\tau_\ell) \neq IV\right].$$

Informally, Γ_1 states that there are no collisions for the points that H was queried on, Γ_2 states that a hash output does not "bump into" a previous hash input, thus retroactively connecting hash chains, and lastly, Γ_3 states that the initialization vector is never a hash output (this will be helpful later on to identify the start of a hash chain).

Claim 4.33. *It holds that* $\Pr[\Sigma] \leq \Pr[\Sigma \wedge \Gamma] + q_{\mathcal{D}}/|\mathcal{Y}| + 2\bar{L}^2/|\mathcal{Y}|$.

Proof. It follows from the collision resistance bound and the zero-preimage resistance bound that $\Pr[\neg\Gamma_1] \leq \bar{L}^2/|\mathcal{Y}|$ and $\Pr[\neg\Gamma_3] \leq q_{\mathcal{D}}/|\mathcal{Y}|$. Also, we immediately obtain $\Pr[\neg\Gamma_2] \leq \bar{L}^2/|\mathcal{Y}|$. Hence we have

$$\Pr[\neg\Gamma] \leq q_{\mathcal{D}}/|\mathcal{Y}| + 2\bar{L}^2/|\mathcal{Y}| \ , \tag{4.24}$$

and thus $\Pr[\Sigma] \leq \Pr[\Sigma \wedge \Gamma] + \Pr[\neg\Gamma] \leq \Pr[\Sigma \wedge \Gamma] + q_{\mathcal{D}}/|\mathcal{Y}| + 2\bar{L}^2/|\mathcal{Y}|$. □

Next, exploiting hide-and-seek, we argue that it is unlikely that $\mathcal{B}$ has queried the entire MD hash chain for $m\|\mathsf{pk}'\|m$ *and* provides the correct output $y' = \mathsf{MD}^H_\perp(m\|\mathsf{pk}'\|m)$. Formally, we consider the event

$$\Lambda := \left[\exists\, i \in [B''] : (m_i, z'_i) \notin Q_{\mathcal{B}}\right]$$

that $\mathcal{B}$ has not made a hash query to one of the high-order intermediate digests z'_i, together with the corresponding message block m_i.

Claim 4.34. *There exist hide-and-seek adversaries* $\bar{\mathcal{D}}, \bar{\mathcal{A}}$ *such that*

$$\Pr[\Sigma \wedge \Gamma \wedge \neg\Lambda] \leq q_{\mathcal{B}} \cdot 2r^2 \cdot \mathbf{Adv}^{\mathsf{HnS}^H_{\mathsf{MD}_\perp}}(\bar{\mathcal{D}}, \bar{\mathcal{A}}) \ , \tag{4.25}$$

where $\bar{\mathcal{A}}$ *makes at most* $q_{\mathcal{B}}$ *and the value* x *chosen by* $\bar{\mathcal{D}}$ *preserves the entropy:*

$$\mathsf{H}_\infty(x \mid z, H) = \mathsf{H}_\infty(m \mid H, \mathsf{sk}, \mathsf{aux}(\mathsf{sk}, m)) \ .$$

Moreover, aux *is polynomial-time computable and* $\mathcal{D}, \mathcal{A}$ *are PPT, then so are* $\bar{\mathcal{D}}$ *and* $\bar{\mathcal{A}}$.

Simplified Proof. [16] We construct adversaries $\bar{\mathcal{D}}$ and $\bar{\mathcal{A}}$ against hide and seek. First, the adversary $\bar{\mathcal{D}}$ simulates the sNR game to $\mathcal{D}$ by sampling a key pair $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}$ and giving sk to $\mathcal{D}$. It forwards all queries and responses by $\mathcal{D}$ to the random oracle and back. When $\mathcal{D}$ outputs a message m, the adversary $\bar{\mathcal{D}}$ outputs $x = m\|\mathsf{pk}\|m$ and $z = (\mathsf{sk}, \mathsf{aux}(\mathsf{sk}, m))$ as its output.

The adversary $\bar{\mathcal{A}}$ takes as input the hash y and $z = \mathsf{sk}, \mathsf{aux}(\mathsf{sk}, m)$. It parses z into sk and $\mathsf{aux}(\mathsf{sk}, m)$ and runs $\mathcal{B}$ on $\mathsf{sk}, y, \mathsf{aux}(\mathsf{sk}, m)$. It forwards all queries to H and their responses. Here, we observe that if Γ and Σ hold but Λ does not hold, then $\bar{\mathcal{A}}$ is able to restore the entire message m (and thus win the corresponding hide-and-seek game against $\mathsf{MD}_\perp$ by recomputing $m\|\mathsf{pk}\|m$), via inspecting $\mathcal{B}$'s queries to H and its output y'. Indeed, $z'_{B''+1} = y'$ (by Σ), and $\mathcal{B}$ has queried $(m_{B''}, z'_{B''})$ such that $H(m_{B''}, z'_{B''}) = z'_{B''+1} = y'$ (by $\neg\Lambda$), and $(m_{B''}, z'_{B''})$ is unique with that property (by Γ_1), and so $\bar{\mathcal{A}}$ can find it. By the same argument, $\bar{\mathcal{A}}$ can then find $(m_{B''-1}, z'_{B''-1}), (m_{B''-2}, z'_{B''-2}), \ldots, (m_1, z'_1)$

[16] This proof is simplified with the assumption that there is no padding and the messages and keys line up with the blocks. For the unsimplified version see Appendix A.3.

in the queries if it knows B'', which is at most $q_{\mathcal{B}}$ and can be guessed with probability $1/q_{\mathcal{B}}$.

Finally, in terms or computational efficiency, $\bar{\mathcal{D}}$ is PPT as long as $\mathcal{D}$ is PPT and aux is polynomial-time computable. Also, if $\mathcal{A}$ is PPT, then $\mathcal{B}$ is PPT and hence so is $\bar{\mathcal{A}}$. □

It remains to to bound the success probability of the adversaries in the case that Λ holds. For this purpose we define m_0 to be the last block of pk' in the sandwich, m_{-1} the second last block etc. and z'_i for $i = 0, -1, -2 \ldots$ accordingly. To this end, consider the events

$$\Delta := \left[\exists\, i \geq -\left|\,\mathsf{pk}\,\right|_{\mathrm{bl}} + 1 : (m_i, z'_i) \notin Q_{\mathcal{B}} \cup Q_{\mathcal{D}} \cup Q_{\mathsf{MD}_\perp(m\|\mathsf{pk}\|m)}\right]$$

and

$$\Delta'_k := \left[\exists\, i \in [B''] : \tau k = (m_i, z'_i) \notin Q_{\mathcal{B}} \cup Q_{\mathsf{MD}_\perp(m\|\mathsf{pk}\|m)} \cup \{\tau k'\}_{k'<k}\right]$$

for $k \in \{1, \ldots, q_{\mathcal{D}}\}$. It is not too hard to see that $\Delta \vee \Delta'_1 \vee \ldots \vee \Delta'_{q_{\mathcal{D}}} \Leftarrow \Sigma \wedge \Lambda \wedge \Gamma$, and thus by basic manipulations

$$\begin{aligned}\Pr[\Sigma] &= \Pr[\Sigma \wedge \Gamma \wedge \Lambda] + \Pr[\Sigma \wedge \Gamma \wedge \neg\Lambda] + \Pr[\Sigma \wedge \neg\Gamma] \\ &\leq \Pr[\Sigma \wedge \Delta] + \sum_k \Pr[\Sigma \wedge \Gamma \wedge \Delta'_k] + \Pr[\Sigma \wedge \Gamma \wedge \neg\Lambda] + \Pr[\neg\Gamma]\,,\end{aligned}$$

where we already have bounds for the last two terms.

First, we argue that $\Pr[\Sigma \wedge \Delta]$ is small. For that purpose, we introduce

$$\Delta^i := \left[i \leq B'' + \left|\,\mathsf{pk}'\,\right|_{\mathrm{bl}} \;\wedge\; (m_{B''-i+1}, z'_{B''-i+1}) \notin Q_{\mathcal{B}} \cup Q_{\mathcal{D}} \cup Q_{\mathsf{MD}_\perp(m\|\mathsf{pk}\|m)}\right]$$

for $i \in [\bar{B}]$.[17] Furthermore, we write $\Delta^{>i} = \Delta^{i+1} \vee \Delta^{i+1} \vee \ldots \vee \Delta^{\bar{B}}$. The crucial observation now is that conditioned on Δ^i, the hash value of $z'_{B''-i+2} = H(m_{B''-i+1}, z'_{B''-i+1})$ is uniformly random and independent of y' (and of prior hash queries etc.), which makes it unlikely for Σ to be satisfied. We formalize this in Claim 4.35 below, and can then conclude that

$$\begin{aligned}&\Pr[\Sigma \wedge \Delta] \\ \leq &\sum_i \Pr\left[\Sigma \wedge \Delta^i \wedge \neg\Delta^{>i}\right] = \sum_i \Pr\left[\Delta^i \wedge \neg\Delta^{>i}\right] \cdot \Pr\left[\Sigma \;\middle|\; \Delta^i \wedge \neg\Delta^{>i}\right] \\ \leq &\sum_i \Pr\left[\Delta^i \wedge \neg\Delta^{>i}\,\right] \cdot \frac{\bar{B}\bar{L}}{|\,\mathcal{Y}\,|} \;\leq\; \frac{\bar{B}\bar{L}}{|\,\mathcal{Y}\,|}\,,\end{aligned}$$

[17] We note that B'' is a random variable (given that m may have variable size), and so the condition $i \leq B''$ ensures $(m_{B''-i+1}, z'_{B''-i+1})$ to be well defined, or else the event is not satisfied.

where it is understood that the sum is over all $i \in [\bar{B}]$ with $\Pr\left[\Delta^i \wedge \neg\Delta^{>i}\right] > 0$, and where $\bar{L} := q_{\mathcal{D}} + \bar{B} + q_{\mathcal{B}} + \bar{B}$; the last inequality follows by the disjointness of $\Delta^i \wedge \neg\Delta^{>i}$ across $i \in [\bar{B}]$.

Claim 4.35. *It holds for every $i \in [\bar{B}]$ with $\Pr[\Delta^i \wedge \neg\Delta^{>i}] > 0$ that*

$$\Pr\left[\Sigma \;\middle|\; \Delta^i \wedge \neg\Delta^{>i}\right] \leq (i-1) \cdot \frac{\bar{L}}{|\mathcal{Y}|} + \frac{1}{|\mathcal{Y}|} \leq \frac{\bar{B}\bar{L}}{|\mathcal{Y}|}\,,$$

where $\bar{L} := q_{\mathcal{D}} + \bar{B} + q_{\mathcal{B}} + \bar{B}$.

Proof. We compute the probability

$$\begin{aligned}
\Pr\left[\Sigma \;\middle|\; \Delta^i \wedge \neg\Delta^{>i}\right] &\leq \Pr\left[\Sigma \wedge \Delta^{i-1} \middle| \Delta^i \wedge \neg\Delta^{>i}\right] + \Pr\left[\neg\Delta^{i-1} \middle| \Delta^i \wedge \neg\Delta^{>i}\right] \\
&\overset{(*)}{\leq} \Pr\left[\Sigma \;\middle|\; \Delta^{i-1} \wedge \Delta^i \wedge \neg\Delta^{>i}\right] + \frac{\bar{L}}{\mathcal{Y}} \\
&\leq \Pr[\Sigma \wedge \Delta^{i-2} | \Delta^{i-1} \wedge \Delta^i \wedge \neg\Delta^{>i}] + \Pr[\neg\Delta^{i-2} | \Delta^{i-1} \wedge \Delta^i \wedge \neg\Delta^{>i}] + \frac{\bar{L}}{\mathcal{Y}} \\
&\leq \Pr[\Sigma \mid \Delta^{i-2} \wedge \Delta^{i-1} \wedge \Delta^i \wedge \neg\Delta^{>i}] + 2 \cdot \frac{\bar{L}}{\mathcal{Y}} \\
&\leq \cdots \\
&\leq \Pr\left[\Sigma \;\middle|\; \Delta^1 \wedge \ldots \wedge \Delta^i \wedge \neg\Delta^{>i}\right] + (i-1) \cdot \frac{\bar{L}}{\mathcal{Y}} \\
&\leq \Pr\left[y' = H(m_{B''}, z'_{B''}) \;\middle|\; \Delta^1 \wedge \ldots \wedge \Delta^i \wedge \neg\Delta^{>i}\right] + (i-1) \cdot \frac{\bar{L}}{\mathcal{Y}} \\
&\overset{(**)}{\leq} \frac{1}{|\mathcal{Y}|} + (i-1) \cdot \frac{\bar{L}}{|\mathcal{Y}|}\,.
\end{aligned}$$

The statement $(*)$ follows the fact that $\neg\Delta^{i-1}$ implies $H(m_{B''-i+1}, z'_{B''-i+1}) \in \{\mathsf{R}(\tau) \mid \tau \in Q_{\mathcal{B}} \cup Q_{\mathcal{D}} \cup Q_{\mathsf{MD}_\perp(m\|\mathsf{pk}\|m)}\}$, which only happens with probability $\bar{L}/\mathcal{Y}$ since $H(m_{B''-i+1}, z'_{B''-i+1})$ is uniform and independent of $Q_{\mathcal{B}} \cup Q_{\mathcal{D}} \cup Q_{\mathsf{MD}_\perp(m\|\mathsf{pk}\|m)}$ conditioned on $\Delta^i \wedge \neg\Delta^{>i}$. For $(**)$ we used the same kind of argument, exploiting that $H(m_{B''}, z'_{B''})$ is uniformly random and independent of y' if $(m_{B''}, z'_{B''})$ has not been queried yet.

In the above series of inequalities we assume the conditions always have non-zero probability. Otherwise, the claim is trivial. In the case that $i \leq 2$ the claim follows directly from the last two rows. □

Towards controlling $\Pr[\Sigma \wedge \mathsf{GD} \wedge \Delta'_k]$, the obstacle is that $\mathcal{D}$ has made a hash query to (m_i, z'_i) and can thus, potentially, make its output m dependent on the hash (e.g., by choosing $m_{i+1} := H(m_i, z'_i)$ then), and so $H(m_i, z'_i)$ may not be "freshly random" anymore.[18] However, by our "sandwich structure" of

[18] Indeed, that is what our attack in Section 4.6.3 exploits.

the hash computation, this is actually not possible. Indeed, since z'_i is a point in *the second part of* the hash chain, all the points in the first part of the chain, i.e., $z_2 := H(m_1, \mathsf{IV})$, $z_3 := H(m_2, z_2)$ up to $z_{B''+1} := H(m_{B''}, z_{B''})$, must be determined already, and hence all of m as well, *before* $\mathcal{D}$ learns the hash of (m_i, z'_i).
We bound the probability in the following claim.

Claim 4.36. $\Pr[\Sigma \wedge \mathsf{GD} \wedge \Delta'_k] \leq q_{\mathcal{D}} \cdot r \cdot \bar{B}\bar{L}/|\mathcal{Y}|$.

Simplified Proof. [19] Formally, for a fixed choice of k, we consider the following procedure to (try to) extract m from the first k queries made by $\mathcal{D}$ and the replies to the first $k-1$ of these queries: Start with the kth query τk and look for a query within $\{\tau 1, \ldots, \tau k - 1\}$ that hashes into $\mathsf{R}(k)$, and then continuing iteratively with that query, until no further such query exists. By construction, this procedure finds $n \leq k$ and $j_1 < \ldots < j_n = k$ such that $H(\tau j_1) = \mathsf{R}(j_2)$, $H(\tau j_2) = \mathsf{R}(j_3), ..., H(\tau j_{n-1}) = \mathsf{R}(j_n)$. The procedure then guesses a value $B^\circ \leftarrow [q_{\mathcal{D}}]$ for the message length. The output of the procedure is then defined to be $\hat{m} := (\mathsf{L}(j_1), \ldots, \mathsf{L}(j_{B^\circ}))$.

We note that $\Sigma \wedge \Gamma \wedge \Delta'_k$ implies that m is a prefix of $\mathsf{L}(j_1)\| \ldots \|\mathsf{L}(j_n)$, and thus, as $n \leq q_{\mathcal{D}}$, it holds that

$$\Pr[m = \hat{m} | \Sigma \wedge \Gamma \wedge \Delta'_k] \geq \frac{1}{q_{\mathcal{D}}}. \tag{4.26}$$

Indeed, Δ'_k implies that $\tau k = (m_i, z'_i)$ for some i, and so $\mathsf{R}(k) = z'_i = \mathsf{MD}^H_\perp(m_1\| \ldots \|m_B\|\mathsf{pk}'\|m_1\| \ldots \|m_{i-1}) = H(m_{i-1}, z'_{i-1}) = H(\tau q_{\mathcal{D}} + q_{\mathcal{B}} + B + i + 1)$. Hence, by Γ_2, there exists $j_{n-1} < k$ so that $\tau j_{n-1} = (m_{i-1}, z'_{i-1})$, and thus $H(\tau j_{n-1}) = \mathsf{R}(k)$. Furthermore, $\mathsf{R}(j_{n-1}) = z'_{i-1} = \mathsf{MD}^H_\perp(m_1\| \ldots \|m_B\|\mathsf{pk}'\|m_1\| \ldots \|m_{i-2})$, and so by repeating the argument, the procedure extracts, in this reversed order, $m_i, m_{i-1}, \ldots, m_1, \mathsf{pk}, m_{B''}, \ldots, m_1$, until $\tau j_1 = (m_1, \mathsf{IV})$, which is when the procedure stops (by Γ_3).

Now we make the following "game hop", by replacing the experiment

$$(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen},\ m \leftarrow \mathcal{D}^H(\mathsf{sk}),\ (\mathsf{pk}', y') \leftarrow \mathcal{B}^H\big(\mathsf{sk}, \mathsf{MD}^H_\perp(m\|\mathsf{pk}\|m), \mathsf{aux}(\mathsf{sk}, m)\big),$$

which defined all the above random variables and probabilities, by

$$(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen},\ \hat{m} \leftarrow \hat{\mathcal{D}}^H(\mathsf{sk}),\ (\mathsf{pk}', y') \leftarrow \mathcal{B}^H\big(\mathsf{sk}, \mathsf{MD}^H_\perp(\hat{m}\|\mathsf{pk}\|\hat{m}), \mathsf{aux}(\mathsf{sk}, \hat{m})\big),$$

where $\hat{\mathcal{D}}$ runs $\mathcal{D}$, but then stops before sending the kth query to H and instead tries to extract m by means of the above procedure from the prior queries. Correspondingly, we denote its output by $\hat{m}$. We stress that $\hat{\mathcal{D}}$ has now query

[19] We give a simplified proof here, for the full version see Appendix A.3.

complexity $q_{\hat{\mathcal{D}}} = k - 1$. The crucial observation is that

$$\Pr[\Sigma \wedge \Delta'_k \wedge \hat{m} = m] \leq \widehat{\Pr}[\Sigma \wedge \Delta]$$

where we use $\widehat{\Pr}$ to denote probabilities in the new experiment. Indeed, in case $\hat{m} = m$ there is no difference in the new experiment, except that now $\hat{\mathcal{D}}$ stops before doing the kth query, and so if Δ'_k is satisfied in the original experiment then Δ is satisfied in the new one. Thus, we can recycle the bound from above. Using the bound $\widehat{\Pr}[\Sigma \wedge \Delta] \leq \bar{B}\bar{L}/|\mathcal{Y}|$ from Claim 4.35 (which holds for any choice of $\mathcal{D}$ and thus also for $\hat{\mathcal{D}}$) we obtain using the above (4.26) that

$$\Pr[\Sigma \wedge \mathsf{GD} \wedge \Delta'_k] \leq q_{\mathcal{D}} \cdot \Pr[\Sigma \wedge \Gamma \wedge \Delta'_k \wedge \hat{m} = m] \leq q_{\mathcal{D}} \cdot \widehat{\Pr}[\Sigma \wedge \Delta] \leq q_{\mathcal{D}} \bar{B}\bar{L}/|\mathcal{Y}|,$$

which concludes the proof of Claim 4.36. □

We wrap up the proof by adding up the probabilities

$$\begin{aligned}
\Pr[\Sigma] \leq& \Pr[\Sigma \wedge \Delta] + \sum_{k=1}^{q_{\mathcal{D}}} \Pr[\Sigma \wedge \Gamma \wedge \Delta'_k] + \Pr[\Sigma \wedge \Gamma \wedge \neg\Lambda] + \Pr[\neg\Gamma] \\
\leq& \frac{\bar{B}\bar{L}}{|\mathcal{Y}|} + \frac{q_{\mathcal{D}}^2 \cdot r \cdot \bar{B}\bar{L}}{|\mathcal{Y}|} + q_{\mathcal{B}} \cdot 2r^2 \cdot \mathbf{Adv}^{\mathsf{HnS}^H}_{\mathsf{MD}_\perp}(\bar{\mathcal{D}}, \bar{\mathcal{A}}) + \frac{q_{\mathcal{D}} + 2\bar{L}^2}{|\mathcal{Y}|} \\
=& q_{\mathcal{B}} \cdot 2r^2 \cdot \mathbf{Adv}^{\mathsf{HnS}^H}_{\mathsf{MD}_\perp}(\bar{\mathcal{D}}, \bar{\mathcal{A}}) + \frac{(q_{\mathcal{D}}^2 \cdot r + 1) \cdot \bar{B}\bar{L} + q_{\mathcal{D}} + 2\bar{L}^2}{|\mathcal{Y}|}
\end{aligned}$$

which shows the claimed bound. □

4.7.3 Hide-and-Seek for Merkle-Damgård

In this section, we provide a (classical) bound on the Hide-and-Seek property of Merkle-Damgård hash functions. For the purpose of using our bounds together with Lemma 4.32, it is sufficient to consider the padding-free versions $\mathsf{MD}^H_\perp : \mathcal{X}^{\leq \bar{B}} \to \mathcal{Y}$ with a bounded number of (at most $\bar{B}$) input blocks, where $\mathcal{X} = \{0,1\}^r$.

Theorem 4.37 ($\mathsf{MD}^H_\perp$ satisfies HnS^H)**.** *Let $\mathcal{D} : \{\perp\} \to \mathcal{X}^{\leq \bar{B}} \times \mathcal{Z}$ and $\mathcal{A} : \mathcal{Y} \times \mathcal{Z} \to \mathcal{X}^{\leq \bar{B}}$ be $\mathsf{HnS}^H_{\mathsf{MD}_\perp}$-adversaries satisfying (4.10) for some $0 < \epsilon < 1$, where $\mathcal{A}$ makes q classical queries to H. Then for $k := \lceil \log|\mathcal{Z}| + \log(1/\epsilon) \rceil$ we have*

$$\mathbf{Adv}^{\mathsf{HnS}^H}_{\mathsf{MD}_\perp}(\mathcal{D}, \mathcal{A}) \leq 2k(q + \bar{B})\epsilon + \epsilon \leq 3k(q + \bar{B})\epsilon\,.$$

To prove the above statement, we recall Corollary 4.25, which reduces the Hide-and-Seek security of any hash function $\mathcal{F}$, to the probability of simultaneously finding preimages of multiple randomly chosen targets. Plugging in $\mathcal{F} = \mathsf{MD}^H_\perp$, we then control the latter in Lemma 4.38 further below, which can

be obtained from the observation that inverting $\mathsf{MD}_{\perp}^{H}$ necessarily inverts H at the last round.

Corollary 4.25. *Let $\mathcal{F} : \mathcal{X} \to \mathcal{Y}$ be a fixed function, $\mathcal{D} : \{\perp\} \to \mathcal{X} \times \mathcal{Z}$ and $\mathcal{A}$ be any Hide-and-Seek adversaries against $\mathcal{F}$. Then for every $(k, T) \in \mathbb{Z}_{>0} \times \mathbb{R}_{>0}$, for $(x, z) \leftarrow \mathcal{D}$ and for $y_1^u, \ldots, y_k^u \leftarrow \mathcal{Y}$ sampled uniformly and independently, we have*

$$\Pr\left[\, x = \mathcal{A}(\mathcal{F}(x), z)\,\right] \le T \cdot \mathsf{guess}(x \mid z) + \left(\frac{|\,\mathcal{Y}\,|}{T}\right)^{k} \Pr\left[y_i^u = \mathcal{F}\big(\mathcal{A}(y_i^u, z)\big)\ \forall i \in [k]\right].$$

Lemma 4.38. *Let $\mathcal{A}^H$ be an oracle algorithm making at most q classical queries to H. Then for $y_1^u, \ldots, y_k^u \leftarrow \mathcal{Y}$ sampled uniformly and independently, we have*

$$\Pr\left[\mathsf{MD}_{\perp}^{H}\left(\mathcal{A}^H(y_i^u)\right) = y_i^u\ \forall i \in [k]\right] \le \big(k(q+\bar{B})/\,|\,\mathcal{Y}\,|\big)^{k}.$$

Proof. For every $i \in [k]$, let $x_i' \| x_i'' := \mathcal{A}^H(y_i^u)$ where $x_i'' \in \mathcal{X}$. Then, with the convention that $\mathsf{MD}_{\perp}^{H}(x_i') := \mathsf{IV}$ if $x_i' = \perp$ is the empty string,

$$\begin{aligned}\Pr\left[\mathsf{MD}_{\perp}^{H}\left(\mathcal{A}^H(y_i^u)\right) = y_i^u\ \forall i \in [k]\right] &\le \Pr\left[H(x_i'', \mathsf{MD}_{\perp}^{H}(x_i')) = y_i^u\ \forall i \in [k]\right] \\ &\le \big(k(q+\bar{B})/\,|\,\mathcal{Y}\,|\big)^{k},\end{aligned}$$

where the second inequality is by Lemma 4.26. □

We wrap up the proof of the Hide-and-Seek property for $\mathsf{MD}_{\perp}^{H}$ (Theorem 4.37) below.

Proof of Theorem 4.37. Combining Corollary 4.25 (for $\mathcal{F} = \mathsf{MD}_{\perp}^{H}$ with $\mathcal{X}_{\mathcal{F}} := \mathcal{X}^{\le B}$ and $\mathcal{Y}_{\mathcal{F}} = \mathcal{Y}$) and Lemma 4.38 immediately gives us

$$\begin{aligned}\mathbf{Adv}_{\mathsf{MD}_{\perp}}^{\mathsf{HnS}^H}(\mathcal{D}, \mathcal{A}) &\le T \cdot \epsilon + \frac{|\,\mathcal{Y}\,|^k}{T^k} \cdot \sum_{z^\circ \in \mathcal{Z}} \Pr\left[\mathsf{MD}_{\perp}^{H}(\mathcal{A}(y_i^u, z^\circ)) = y_i^u\ \forall i \in [k]\right], \\ &\le T \cdot \epsilon + |\,\mathcal{Z}\,| \cdot \left(\frac{k(q+\bar{B})}{T}\right)^{k} \qquad (4.27)\end{aligned}$$

for every $k, T \in \mathbb{Z}_{>0}$, where $y_1^u, \ldots, y_k^u \leftarrow \mathcal{Y}_f$ are sampled uniformly and independently. Plugging in $T = 2k(q+\bar{B})$ and $k = \lceil \log |\,\mathcal{Z}\,| + \log(1/\epsilon) \rceil$, the above is then bounded by

$$\le 2k(q+\bar{B})\epsilon + |\,\mathcal{Z}\,| \cdot 2^{-k} \le 2k(q+\bar{B})\epsilon + \epsilon\,,$$

which concludes the proof. □

4.8 Positive Results with Computational Entropy

Here, we consider the computational variant of $\mathsf{sNR}^{H,\perp}$, where we restrict $\mathcal{D}^H, \mathcal{A}^H$ and aux to be PPT (oracle) algorithms. Furthermore, the entropy requirement (4.9) is replaced by

$$\mathsf{HILL}^{H}_{\infty} \underset{\substack{(\mathsf{sk},pk)\leftarrow\mathsf{KGen}^H \\ m\leftarrow\mathcal{D}^H(\mathsf{sk})}}{(m \mid \mathsf{sk}, \mathsf{aux}(\mathsf{sk}, m))} \geq \omega(\log \lambda)\,, \tag{4.28}$$

and we then naturally demand that the game $\mathsf{sNR}^{H,\perp}$ can be won with negligible probability $\mathsf{negl}(\lambda)$ only.

Remark 4.39. *Interestingly, and maybe somewhat surprisingly, in the computational setting* $\mathsf{sNR}^{H,\perp}$ *does not imply* $\mathsf{NR}^{H,\perp}$*, in contrast to the statistical setting, as explained in Section 4.5.1. Indeed, in Section 4.4.4 we showed that the BUFF transform* $\mathsf{BUFF}[\mathcal{S}, H]$ *does in general not satisfy* $\mathsf{NR}^{H,\perp}$ *in the computational setting, while below we show that it does satisfy* $\mathsf{sNR}^{H,\perp}$*. See Remark 4.21 for why our proof does not carry over to* $\mathsf{NR}^{H,\perp}$*. We suspect that the two notions are incomparable in the computational setting.*

4.8.1 BUFF via RO is sNR, Computationally

We get the following positive result on the computational $\mathsf{sNR}^{H,\perp}$ security of the BUFF transform $\mathsf{BUFF}[\mathcal{S}, H]$ when using a random oracle H.

Theorem 4.40. *Let* $\mathcal{S} = (\mathsf{KGen}, \mathsf{Sign}^H, \mathsf{Vrfy}^H)$ *be a signature scheme in ROM, where* KGen *makes no query to* H*, and let* $\mathsf{BUFF}[\mathcal{S}, H]$ *be the signature scheme obtained by applying the BUFF transform that uses* H*. Then for every PPT hint function* aux*, and for any PPT adversaries* $\mathcal{D}^H, \mathcal{A}^H$ *against* $\mathsf{sNR}^{H,\perp}_{\mathsf{BUFF}[\mathcal{S},H]}$ *that satisfy* (4.28)*, we have*

$$\mathbf{Adv}^{\mathsf{sNR}^{H,\perp}}_{\mathsf{BUFF}[\mathcal{S},H]}(\mathcal{D}, \mathcal{A}, \mathsf{aux}) \leq \mathsf{negl}(\lambda)\,.$$

In spirit, we can recycle Lemma 4.20 to reduce the computational variant of $\mathsf{sNR}^{H,\perp}$ to the computational variant of Hide-and-Seek, and then we show in Lemma Lemma 4.41 that the latter is hard as well, which follows rather directly from the statistical hardness and the definition of the HILL entropy.

Proof. Take $(\bar{\mathcal{D}}, \bar{\mathcal{A}})$ as in Lemma 4.20, for which

$$\mathbf{Adv}^{\mathsf{sNR}^{H,\perp}}_{\mathsf{BUFF}[\mathcal{S},H]} \leq \mathsf{poly}(\lambda) \cdot \mathbf{Adv}^{\mathsf{HnS}^H}(\bar{\mathcal{D}}, \bar{\mathcal{A}}) + \mathsf{negl}(\lambda)\,,$$

where we exploit that the numbers of queries made by Sign, $\mathcal{D}$, and $\mathcal{A}$ are bounded by their (polynomial) running times, respectively, and that the addi-

tive term $q_K\epsilon$ in (4.13) vanishes due to the assumption that KGen makes no query to H. Hence it suffices to control the HnS^H advantage of $(\bar{\mathcal{D}}, \bar{\mathcal{A}})$.

Towards this end, we first note that, by inspecting the construction of $\bar{\mathcal{D}}$ with $x = (\mathsf{pk}, m)$ and $z = (\mathsf{sk}, \mathsf{aux}(\mathsf{sk}, m))$, the HILL entropy variant of (4.11) follows:

$$\mathsf{HILL}^H_{\infty,(x,z)\leftarrow\bar{\mathcal{D}}^H}(x \mid z) \geq k(\lambda) \iff \mathsf{HILL}^H_{\infty,\substack{(\mathsf{sk},pk)\leftarrow\mathsf{KGen}^H \\ m\leftarrow\mathcal{D}^H(\mathsf{sk})}}(m \mid \mathsf{sk}, \mathsf{aux}(\mathsf{sk}, m)) \geq k(\lambda) \ ,$$

where the equivalence is due to the public key pk being *efficiently derivable* from its corresponding secret key sk, and so (4.14) also holds for the HILL entropy. Combining the above with (4.28), we obtain

$$\mathsf{HILL}^H_{\infty,(x,z)\leftarrow\bar{\mathcal{D}}^H}(x \mid z) \geq \omega(\log\lambda) \ .$$

Moreover, by Lemma 4.20, $\bar{\mathcal{D}}\colon \{\bot\} \to \mathcal{X} \times \mathcal{Z}$ with $\mathcal{Z} = \mathcal{SK} \times \mathcal{AUX}$. Hence

$$\log|\mathcal{Z}| = \log|\mathcal{SK}| + \log|\mathcal{AUX}| \leq \mathsf{poly}(\lambda)$$

due to both KGen and aux being poly-time. Finally, Lemma 4.20 ensures that $\bar{\mathcal{A}}$ is PPT whenever $\mathcal{A}$ is, which is satisfied by assumption. Thus, the assumptions for Lemma 4.41 below (the hardness of Hide-and-Seek in the computational setting) are all satisfied, and so

$$\mathbf{Adv}^{\mathsf{HnS}^H}(\bar{\mathcal{D}}, \bar{\mathcal{A}}) \leq \mathsf{negl}(\lambda) \ ,$$

which concludes the proof. □

The following provided the computational hardness of Hide-and-Seek.

Lemma 4.41. *Let $\mathcal{D}^H\colon \{\bot\} \to \mathcal{X} \times \mathcal{Z}$ and $\mathcal{A}^H : \mathcal{Y} \times \mathcal{Z} \to \mathcal{X}$ be adversaries against HnS^H, with $\mathcal{A}$ being PPT, $\log|\mathcal{Z}| < \mathsf{poly}(\lambda)$, and*

$$\mathsf{HILL}^H_{\infty,(x,z)\leftarrow\mathcal{D}^H}(x \mid z) \geq \omega(\log\lambda) \ .$$

Then $\mathbf{Adv}^{\mathsf{HnS}^H}(\mathcal{D}, \mathcal{A}) \leq \mathsf{negl}(\lambda)$.

Proof. Let $(x, z) \leftarrow \mathcal{D}^H$. Via the entropy condition, there is an (H-dependent) random variable $x^* \in \mathcal{X}$ such that $\mathsf{guess}(x^* \mid H, z) \leq \mathsf{negl}(\lambda)$ and moreover (x^*, z) and (x, z) are computationally indistinguishable. Without loss of generality, we may assume (x^*, z) is sampled via a (possibly unbounded) hider $\mathcal{D}^{*H}$. Now, inspect the displayed games $\mathsf{HnS}^H(\mathcal{D}, \mathcal{A})$ and $\mathsf{HnS}^H(\mathcal{D}^*, \mathcal{A})$ below.

$\mathsf{HnS}^H(\mathcal{D}, \mathcal{A})$
1: $(x, z) \leftarrow \mathcal{D}^H$
2: **return** $x = \mathcal{A}^H(H(x), z)$

$\mathsf{HnS}^H(\mathcal{D}^*, \mathcal{A})$:
1: $(x^*, z) \leftarrow \mathcal{D}^{*H}$
2: **return** $x^* = \mathcal{A}^H(H(x^*), z)$

By the computational indistinguishability, it follows that

$$|\mathbf{Adv}^{\mathsf{HnS}^H}(\mathcal{D}, \mathcal{A}) - \mathbf{Adv}^{\mathsf{HnS}^H}(\mathcal{D}^*, \mathcal{A})| \leq \mathsf{negl}(\lambda)\,.$$

Finally, we can apply Theorem 4.22 to the HnS^H adversaries $\mathcal{D}^*$ and $\mathcal{A}$, which satisfy the statistical entropy condition, and so we have $\mathbf{Adv}^{\mathsf{HnS}}(\mathcal{D}^*, \mathcal{A}) \leq \mathsf{negl}(\lambda)$. This concludes the proof. □

Remark 4.42. *Interestingly, towards proving* $\mathsf{sNR}^{H,\perp}$ *of the BUFF transform in the* statistical *setting, as we did earlier in the thesis, it would have been sufficient to show that the random oracle satisfies (the statistical variant of)* HnS *for a* query bounded *hider* $\mathcal{D}$*. However, for the above line of reasoning in the computational setting, it is essential that Theorem 4.22 holds for a query unbounded hider; indeed, above,* x^* *may be arbitrarily dependent on* H*, and so might not be producible by a query bounded hider* $\mathcal{D}^*$*.*

4.8.2 Sandwich BUFF via MD is sNR, Computationally

We obtain a similar positive result on the computational $\mathsf{sNR}^{H,\perp}$ security of the Sandwich BUFF $\mathsf{sBUFF}[\mathcal{S}, \mathsf{MD}]$ when using a Merkle-Damgård hash function. Recall that, we denote by $\mathsf{MD}^H : \{0,1\}^* \to \mathcal{Y}$ the Merkle-Damgård hash function that is allowed to query a random oracle $H : \mathcal{X} \times \mathcal{Y} \to \mathcal{Y}$ as its round function, where $\mathcal{X} = \{0,1\}^r$ and $\mathcal{Y} = \{0,1\}^{n_f}$ satisfy $\omega(\log \lambda) \leq r, n_f \leq \mathsf{poly}(\lambda)$.

Theorem 4.43. *Let* $\mathcal{S} = (\mathsf{KGen}, \mathsf{Sign}^H, \mathsf{Vrfy}^H)$ *be a signature scheme in ROM, where* KGen *makes no query to* H*, and let* $\mathsf{sBUFF}[\mathcal{S}, \mathsf{MD}]$ *be the signature scheme obtained by applying the Sandwich BUFF that uses* MD^H*. Then for every PPT hint function* aux*, and* $\mathsf{sNR}^{H,\perp}$ *adversaries* $\mathcal{D}$ *and* $\mathcal{A}$ *that satisfy* (4.28)*, we have*

$$\mathbf{Adv}^{\mathsf{sNR}^{H,\perp}}_{\mathsf{sBUFF}[\mathcal{S},\mathsf{MD}]}(\mathcal{D}, \mathcal{A}, \mathsf{aux}) \leq \mathsf{negl}(\lambda)\,.$$

Similarly, once we have Lemma 4.32 and Theorem 4.37, the proof follows line by line as that of Theorem 4.40. Indeed, the reductions in Lemma 4.32 carry the HILL entropy from the $\mathsf{sNR}^{H,\perp}$ game to the $\mathsf{HnS}^H_{\mathsf{MD}}$ game, and the proven Hide-and-Seek property in Theorem 4.37 carries over to the computational setting.

Proof of Theorem 4.43. Since $\mathcal{D}, \mathcal{A}$ runs in polynomial-time, for $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}$ and $m \leftarrow \mathcal{D}^O(\mathsf{sk})$ the message length $|m| \leq \mathsf{poly}(\lambda)$ is polynomially bounded. Also we assume (up to a negligible security loss) that m is always

non-empty via (4.28). Hence we consider the domain of MD only consists of non-empty bit strings that are at most $\bar{B}$ blocks long, for some $\bar{B} \leq \mathsf{poly}(\lambda)$.

Let $\bar{\mathcal{D}}$ and $\bar{\mathcal{A}}$ be the PPT algorithms as specified in Lemma 4.32, so that

$$\begin{aligned}
&\mathbf{Adv}^{\mathsf{sNR}^{H,\perp}}_{\mathsf{sBUFF}[\mathcal{S},\mathsf{MD}]}(\mathcal{D},\mathcal{A},\mathsf{aux}) \\
&\leq 2q_{\mathcal{B}}{\cdot}r^2 \cdot \mathbf{Adv}^{\mathsf{HnS}^H_{\mathsf{MD}_\perp}}(\bar{\mathcal{D}},\bar{\mathcal{A}}) + \frac{(q_{\mathcal{D}}^2{\cdot}r+1)\cdot \bar{B}\bar{L} + q_{\mathcal{D}} + 2\bar{L}^2}{|\,\mathcal{Y}\,|}. \\
&\leq \mathsf{poly}(\lambda) \cdot \mathbf{Adv}^{\mathsf{HnS}^H_{\mathsf{MD}_\perp}}(\bar{\mathcal{D}},\bar{\mathcal{A}}) + \mathsf{negl}(\lambda) \qquad (4.29)
\end{aligned}$$

where parameters $\bar{L}, q_{\mathcal{B}}, k$ are as specified in the lemma, and the last inequality exploits the fact that $r, c, \bar{B}, \bar{L}, q_{\mathcal{B}}, q_{\mathcal{D}}, q_{\mathcal{A}}, q_{\mathcal{S}} \leq \mathsf{poly}(\lambda)$ and that $|\,\mathcal{Y}\,| \geq \lambda^{\omega(1)}$. Moreover by inspecting the construction of $\bar{\mathcal{D}}$ it follows that

$$\mathsf{HILL}^H_{\infty}_{(x,z)\leftarrow\bar{\mathcal{D}}^O} \geq k(\lambda) \iff \mathsf{HILL}^H_\infty(m \mid \mathsf{sk}, \mathsf{aux}(\mathsf{sk}, m)) \geq k(\lambda)\ .$$

Combining the above with (4.28), we thus have

$$\mathsf{HILL}^H_{\infty}_{(x,z)\leftarrow\bar{\mathcal{D}}^H}(x \mid z) \geq \lambda^{\omega(1)}\ ,$$

which implies the existence of an H-dependent random variable x^* for $(x,z) \leftarrow \mathcal{D}^H$ such that

$$\mathsf{HILL}_\infty(x^* \mid z, H) \geq \omega(\log\lambda)\ ,$$

and yet (x^*, z) and (x, z) are computationally indistinguishable for oracle algorithms querying H. Without loss of generality, let $\mathcal{D}^*$ be the Hide-and-Seek seeker that samples x^*, and note that the seeker $\bar{\mathcal{A}}$ is PPT and makes only polynomially many queries to H. Hence,

$$\mathbf{Adv}^{\mathsf{HnS}^H_{\mathsf{MD}_\perp}}(\bar{\mathcal{D}},\bar{\mathcal{A}}) \leq \mathbf{Adv}^{\mathsf{HnS}^H_{\mathsf{MD}_\perp}}(\mathcal{D}^*,\bar{\mathcal{A}}) + \mathsf{negl}(\lambda) \leq \mathsf{negl}(\lambda)\ ,$$

where the first inequality follows from the indistinguishability between (x^*, z) and (x, z), and the last inequality follows from Theorem 4.37 respectively in the Merkle-Damgård and the Sponge case. Putting things together, we conclude the proof. □

Chapter 5

KEM Combiners via Split-key PRFs

5.1 Introduction

In the upcoming transition to post-quantum secure cryptographic standards, *combiners* play an important role. A combiner can be used to compile several cryptographic schemes into a new, hybrid scheme, which offers the same (or a similar) functionality, and so that the new scheme is secure as long as *at least one* of the original schemes is secure. For example, combining a well-established but quantum-insecure scheme with a believed-to-be quantum-secure (but less well studied) scheme then offers the best of both worlds: it offers security against quantum attacks, should there really be a quantum computer in the future, but it also offers some protection in case the latter scheme turns out to be insecure (or less secure than expected) even against classical attacks. In other words, using a combiner in this context ensures that we are not making things less secure by trying to aim for quantum security.

In [GHP18], Giacon, Heuer, and Poettering proposed a family of KEM combiners that works as follows. To produce the combined ciphertext $C := (c_1, \ldots, c_n)$, one concatenates all component ciphertexts $c_1, \ldots, c_n$ produced by the component KEMs, and to produce the combined session key $K := F(k_1, \ldots, k_n, C)$, one feeds the component session keys $k_1, \ldots, k_n$ and C into a key-derivation function F. They then show that instantiating F with a *split-key pseudorandom function* (skPRF) would preserve the standard IND-CCA security as a KEM combiner. Roughly speaking, an skPRF is a function $F(k_1, \ldots, k_n, x)$ with multiple keys $k_1, \ldots, k_n$ that behaves like a PRF if at least one of the keys k_i is kept secret.[1] A particularly efficient skPRF proposed

[1] For technical reasons, one typically restricts the PRF attacker from querying F with the same input x twice.

by [GHP18] is

$$F(k_1, \ldots, k_n, x) := H(g(k_1, \ldots, k_n), x) \ ,$$

where H is a cryptographic hash function, and where we assume $g(k_1, \ldots, k_n)$ has high min-entropy as long as one of the keys k_i is freshly chosen. The classical security of this skPRF has been proven in the ROM (i.e. modelling H as a random oracle) considering classical attackers, which implies the classical security of the aforementioned KEM combiner.

5.1.1 Our Contributions

Given its relevance for constructing hybrid schemes from both pre-quantum and post-quantum schemes, the post-quantum security of F, and hence that of the corresponding KEM combiner, has remained an important open question prior to our work [DFH22], on which this chapter is based.

Quantum security of a skPRF. In Section 5.4, we close this gap via showing that F is a secure skPRF, considering quantum attackers as well. More specifically, we show that for every attacker $\mathcal{A}$ making at most q_H queries to H and q_F queries to F (with one of the keys k_i sampled and kept secret at the beginning), $\mathcal{A}$ cannot distinguish F from a truly random function R chosen independently of H, with the following concrete bound on the advantage

$$\mathbf{Adv}_F^{\mathsf{skPRF}}(\mathcal{A}) := \left| \Pr\left[\, 1 \leftarrow \mathcal{A}^{F,H} \,\right] - \Pr\left[\, 1 \leftarrow \mathcal{A}^{R,H} \,\right] \right| \leq 4\sqrt{2q_F^2 q_H \epsilon} + 4\sqrt{2q_H^2 q_F \epsilon} \ ,$$

where ϵ is a parameter close to zero, depending on the function g. This confirms the quantum security of F as an skPRF. Consequently, the corresponding KEM combiner as described above and in [GHP18] is also secure against quantum attackers.

By default, such an attacker $\mathcal{A}$ can then choose *adaptively*, i.e., depending on answers to previous queries, at what point to query *which oracle*. This is in contrast to a *static* $\mathcal{A}$ that has a predefined order of when it queries which oracle.[2] In certain cases, proving security for a static attacker is easier than proving security for a full fledged adaptive attacker, or taking care of adaptivity (naively) results in an unnecessary blow-up in the error term (see later).

Our security proof crucially exploits a generic compiler that we introduce in Section 5.3 to reduce every such adaptive attacker to a static one. Namely, in spirit, our security proof is a typical hybrid proof, where we replace, one by one, the queries to F by queries to R; however, the crux is that for each hybrid, corresponding to a particular function query that is to be replaced, the closeness of the current to the previous hybrid depends on the number of hash queries *between the current and the previously replaced function query*. In case

[2] In either case, we allow $\mathcal{A}$ to decide adaptively *what input* to query, when having decided (adaptively or statically) on which oracle to query.

of an adaptive $\mathcal{A}$, *each* such "window" of hash queries between two function queries could be as large as the total number of hash queries in the worst case, giving rise to a huge multiplicative blow-up when using this naive bound. Instead, for a static $\mathcal{A}$, each such window is bounded by a fixed number, with the sum of these numbers being the total number of hash queries.

By means of our compiler, we can turn the possibly adaptive $\mathcal{A}$ into a static one (almost) for free, and this way avoid an unnecessary blow-up, respectively bypassing additional complications that arise by trying to avoid this blow-up by other means.

Our adaptive-to-static compiler. More generally, we now consider the attackers given query access to n oracles $\mathcal{O}_1, \ldots, \mathcal{O}_n$. In light of the above, it is desirable to have a generic compiler that transforms any adaptive attacker $\mathcal{A}$ into a static attacker $\bar{\mathcal{A}}$ that is equally successful in the attack. And there is actually a simple, naive solution for that. Indeed, let $\mathcal{A}$ be an arbitrary oracle algorithm that makes adaptive queries to n oracles $\mathcal{O}_1, \ldots, \mathcal{O}_n$, and consider the static oracle algorithm $\bar{\mathcal{A}}$ defined as follows: $\bar{\mathcal{A}}$ simply runs $\mathcal{A}$, and at every point in time when $\mathcal{A}$ makes a query to one of $\mathcal{O}_1, \ldots, \mathcal{O}_n$ (but due to the adaptivity it will only become clear at the time of the query *which* $\mathcal{O}_i$ is to be queried then), the algorithm $\bar{\mathcal{A}}$ makes n queries, one to every $\mathcal{O}_i$, and it relays $\mathcal{A}$'s query to the right oracle, while making dummy queries to the other oracles.

At first glance, this simple solution is not too bad. It certainly transforms any adaptive $\mathcal{A}$ into a static $\bar{\mathcal{A}}$ that will be equally successful, and the blow-up in the total query complexity is a factor n only, which is mild given that the typical case is $n = 2$. However, it turns out that in many situations, considering the blow-up in the total query complexity is not good enough.

For example, consider the case of an attacker against a public-key encryption scheme in the random oracle model. In this example, it is typically assumed that $\mathcal{A}$ may make many more queries to the random oracle than to the decryption oracle, i.e., $q_H \gg q_D$, where q_H and q_D are the numbers of queries to the two different oracles respectively. But then, applying the above simple compiler, $\bar{\mathcal{A}}$ makes the same number of queries to the random oracle and to the decryption oracle; namely $\bar{q}_H = \bar{q}_D = q_H + q_D$. Furthermore, the actual figure of merit, namely the advantage of an attacker $\bar{\mathcal{A}}$, is typically not (bounded by) a function of the total query complexity, but a function of the two respective query complexities q_H and q_D *individually*. For example, if one can show that the advantage of any *static* attacker $\bar{\mathcal{A}}$ with respective query complexities $\bar{q}_H$ and $\bar{q}_D$ is bounded by, say, $\bar{q}_H \bar{q}_D^2 \epsilon$ (for some small ϵ), then the above compiler gives a bound on the advantage of any *adaptive* attacker $\mathcal{A}$ with respective query complexities q_H and q_D of $q_H^3\epsilon + 2q_H^2 q_D \epsilon + q_H q_D^2 \epsilon + q_D^3 \epsilon$. If $q_H \gg q_D$ then this is significantly worse than $\approx q_H q_D^2 \epsilon$, which one might hope for given the bound for static $\bar{\mathcal{A}}$.

Our contribution in Section 5.3, is a compiler that transforms any *adaptive* oracle algorithm $\mathcal{A}$ that makes at most q_i queries to oracle $\mathcal{O}_i$ for $i = 1, \ldots, n$ into a *static* oracle algorithm $\bar{\mathcal{A}}$ that makes at most $\bar{q}_i = nq_i$ queries to oracle $\mathcal{O}_i$ for $i = 1, \ldots, n$. Thus, rather than controlling the blow-up in the total number of queries, we can control the blow-up in the number of queries for each oracle *individually*, yet still with the same factor n. Our result applies for *any* vector $\mathbf{q} = (q_1, \ldots, q_n) \in \mathbb{Z}^n_{\geq 0}$ and contains no hidden constants. Our compiler naturally depends on $\mathbf{q}$ (or, alternatively, needs $\mathbf{q}$ as input) but otherwise only requires straight-line black-box access to $\mathcal{A}$, and it preserves efficiency: the run time of $\bar{\mathcal{A}}$ is polynomial in $Q = q_1 + \cdots + q_n$, plus the time needed to run $\mathcal{A}$. Furthermore, the compiler is applicable to any classical or quantum oracle algorithm $\mathcal{A}$, where in the latter case the queries to the oracles $\mathcal{O}_1, \ldots, \mathcal{O}_n$ may be classical or quantum as well; however, the *choice* of the oracle for each query is assumed to be classical (so that individual query complexities are well defined).

In the above made-up example of a public-key encryption scheme with advantage bounded by $\bar{q}_H \bar{q}_D^2 \epsilon$ for any static $\bar{\mathcal{A}}$ with respective query complexities $\bar{q}_H$ and $\bar{q}_D$, we now get the bound $8 q_H q_D^2 \epsilon$ for any adaptive $\mathcal{A}$ with respective query complexities q_H and q_D.

Besides applying our adaptive-to-static compiler in the main contribution (i.e. for analyzing the skPRFs), we show the usefulness of our adaptive-to-static compiler in Section 5.3.4 by discussing two additional example results from the literature. One is the security proof by Alkim *et al.* [ABB+17] of the qTESLA signature scheme [ABB+20] in the quantum random oracle model; the other is the recent work by Alagic, Bai, Katz and Majenz [ABKM22] on the quantum security of the famous Even-Mansour cipher. In both these works, the adaptivity of the attacker was a serious obstacle and caused a significant overhead and additional complications in the proof. With our results, these complications could have been avoided without sacrificing much in the security loss (as would be the case with using a naive compiler).

Interestingly, all three example applications are in the realm of quantum security (of a classical scheme). This seems to suggest that the kind of adaptivity we consider here is not so much of a hurdle in the case of classical queries. Indeed, in that case, a typical argument works by inspecting the entire query transcript and identifying an event with the property that conditioned on this event, whatever needs to be shown holds *with certainty*, and then it remains to show that this event is very likely to occur. In the case of quantum queries, this kind of reasoning does not apply since one cannot "inspect" the query transcript anymore; instead, one then typically resorts to some sort of hybrid argument where queries are replaced one-by-one, and then adaptivity of the queries may — and sometimes does, as we discuss — form a serious obstacle.

5.2 Preliminaries

We consider oracle algorithms $\mathcal{A}^{\mathcal{O}_1,\ldots,\mathcal{O}_n}$ that make queries to (possibly unspecified) oracles $\mathcal{O}_1,\ldots,\mathcal{O}_n$, see Fig. 5.1 (left). Sometimes, and in particular when the oracles are not specified, we just write $\mathcal{A}$ and leave it implicit that $\mathcal{A}$ makes oracle calls. We allow $\mathcal{A}$ to be classical or quantum, and in the latter case we may also allow the queries (to some of the oracles) to be quantum; however, the choice of *which* oracle is queried is always classical. For the purpose of our work, we may assume $\mathcal{A}$ to have no input; any potential input could be hardwired into $\mathcal{A}$. For a vector $\mathbf{q} = (q_1,\ldots,q_n) \in \mathbb{Z}^n_{\geq 0}$, we say that $\mathcal{A}$ is a $\mathbf{q}$-*query* oracle algorithm if it makes at most q_i queries to the oracle $\mathcal{O}_i$.

In general, such an oracle algorithm $\mathcal{A}$ may decide *adaptively* which oracle to query at what step, dependent on previous oracle responses. In contrast to this, a *static* oracle algorithm has an arbitrary but pre-defined order in querying the oracles.

Our goal will be to transform any *adaptive* oracle algorithm $\mathcal{A}$ into a *static* oracle algorithm $\bar{\mathcal{A}}$ that is functionally equivalent, while keeping the blow-up in query complexity for each individual oracle, i.e., the blow-up for each individual q_i, small. By *functionally equivalent* (for certain oracle instantiations) we mean the respective executions of $\mathcal{A}^{O_1,\ldots,O_n}$ and $\bar{\mathcal{A}}^{O_1,\ldots,O_n}$ give rise to the same output distribution *for all* (the considered) instantiations $O_1,\ldots,O_n$ of the oracles $\mathcal{O}_1,\ldots,\mathcal{O}_n$. In case of *quantum* oracle algorithms, we require the output state to be the same.

For this purpose, we declare that an *interactive oracle algorithm* $\mathcal{B}$ is an interactive algorithm with two distinct interaction interfaces, one for the interaction with $\mathcal{A}$ (we call this the *simulation interface*), and one for the oracle queries (we call this the *oracle interface*), see Fig. 5.1 (middle). For any oracle algorithm $\mathcal{A}$, we then denote by $\mathcal{B}[\mathcal{A}]$ the oracle algorithm that is obtained by composing $\mathcal{A}$ and $\mathcal{B}$ in the obvious way. In other words, $\mathcal{B}[\mathcal{A}]$ runs $\mathcal{A}$ and answers all of $\mathcal{A}$'s oracle queries using its simulation interface; furthermore, $\mathcal{B}[\mathcal{A}]$ outputs whatever $\mathcal{A}$ outputs at the end of this run of $\mathcal{A}$, see Fig. 5.1 (right).[3]

In contrast to $\mathcal{A}$ (where, for our purpose, any input could be hardwired), we explicitly allow an interactive oracle algorithm $\mathcal{B}$ to obtain an input. Indeed, our transformation, which turns any adaptive oracle algorithm $\mathcal{A}$ into a static oracle algorithm $\bar{\mathcal{A}}$, needs to "know" $\mathbf{q}$, i.e., the number of queries $\mathcal{A}$ makes to the different oracles. Thus, this will be provided in the form of an input to $\mathcal{B}$; for reasons to be clear, it be provided in unary, i.e., as $1^{\mathbf{q}} := (1^{q_1},\ldots,1^{q_n})$.

We stress that we do not put any computational restriction on the oracle algorithms $\mathcal{A}$ (beyond bounding the queries to the individual oracles); however, we do want our transformation to preserve efficiency. Therefore, we say that

[3]Note, we silently assume consistency between $\mathcal{A}$ and $\mathcal{B}$, i.e. $\mathcal{A}$ should send a message when $\mathcal{B}$ expects one and the format of these messages should match the format of the messages that $\mathcal{B}$ expects (and vice versa), so that the above composition makes sense. Should $\mathcal{B}$ encounter some inconsistency, it will abort.

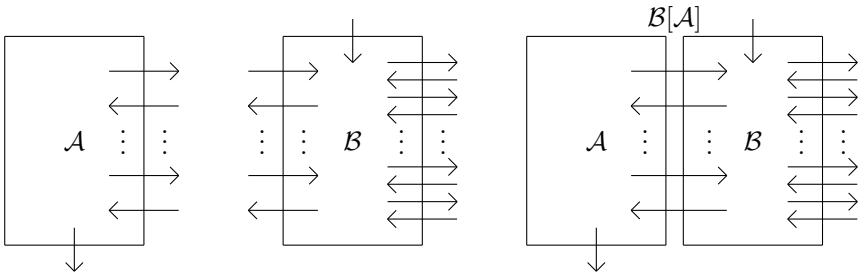


Figure 5.1: An oracle algorithm $\mathcal{A}$ (left), an interactive oracle algorithm $\mathcal{B}$ (middle), and the oracle algorithm $\mathcal{B}[\mathcal{A}]$ obtained by composing $\mathcal{A}$ and $\mathcal{B}$ (right).

an interactive oracle algorithm $\mathcal{B}$ is *polynomial-time* if the number of local computation steps it performs is bounded to be polynomial in its input size, and where we declare that copying an *incoming* message on the simulation interface to an *outgoing* message on the oracle interface, and vice versa, is unit cost (irrespectively of the size of the message). By providing $\mathbf{q}$ in unary, we thus ensure that $\mathcal{B}$ is polynomial-time in $q_1 + \cdots + q_n$.

5.3 A Generic Adaptive-to-static Compiler

5.3.1 Our Result

Let n be an arbitrary positive integer. We present here a generic adaptive-to-static compiler $\mathcal{B}$ that, on input a vector $\mathbf{q} \in \mathbb{Z}^n_{\geq 0}$, turns any *adaptive* $\mathbf{q}$-query oracle algorithm $\mathcal{A}^{\mathcal{O}_1,\ldots,\mathcal{O}_n}$ into a *static* $n\mathbf{q}$-query algorithm.

Theorem 5.1. *There exists a polynomial-time interactive oracle algorithm $\mathcal{B}$, such that for any $\mathbf{q} \in \mathbb{Z}^n_{\geq 0}$ and any* adaptive $\mathbf{q}$*-query oracle algorithm $\mathcal{A}^{\mathcal{O}_1,\ldots,\mathcal{O}_n}$, the oracle algorithm $\mathcal{B}[\mathcal{A}](1^{\mathbf{q}})$ is a* static $n\mathbf{q}$*-query oracle algorithm that is functionally equivalent to $\mathcal{A}$ for all* stateless *instantiations of the oracles $\mathcal{O}_1, \ldots, \mathcal{O}_n$.*

Remark 5.2. *As phrased, Theorem 5.1 applies to oracle algorithms $\mathcal{A}$ that have no input. This is merely for simplicity. In case of an oracle algorithm $\mathcal{A}$ that takes an input, we can simply apply the statement to the algorithm $\mathcal{A}(x)$ that has the input x hardwired, and so argue that Theorem 5.1 also applies in that case.*

Remark 5.3. *$\mathcal{B}[\mathcal{A}]$ is guaranteed to behave the same way as $\mathcal{A}$ for* stateless *(instantiations of the) oracles only. This is become most of the queries that $\mathcal{B}[\mathcal{A}]$ makes are actually dummy queries (i.e., queries on a default input and with the response ignored), which have no effect in case of stateless oracles,*

but may mess up things in case of stateful oracles. Theorem 5.1 extends to arbitrary stateful oracles if we allow $\mathcal{B}[\mathcal{A}]$ *to* skip *queries instead of making dummy queries (but the skipped queries would still count towards the query complexity).*

Given the vector $\mathbf{q} = (q_1, \ldots, q_n) \in \mathbb{Z}^n_{\geq 0}$, the core of the problem is to find a *fixed* sequence of $\mathcal{O}_i$'s in which each individual $\mathcal{O}_i$ occurs at most nq_i times, and so that *every* sequence of $\mathcal{O}_i$'s that contains each individual $\mathcal{O}_i$ at most q_i times can be embedded into the former. We consider and solve this abstract problem in the following section, and then we wrap up the proof of Theorem 5.1 in Section 5.3.3.

5.3.2 The Technical Core

Let Σ be an non-empty finite set of cardinality n. We refer to Σ as the *alphabet*. As is common, Σ^* denotes the set of finite strings over the alphabet Σ. In other words, the elements of Σ^* are the strings/sequences $s = (s_1, \ldots, s_\ell) \in \Sigma^\ell$ with arbitrary $\ell \in \mathbb{Z}_{\geq 0}$ (including $\ell = 0$).

Following standard terminology, for $s = (s_1, \ldots, s_\ell)$ and $s' = (s'_1, \ldots, s'_m)$ in Σ^*, the *concatenation* of s and s' is the string $s \| s' = (s_1, \ldots, s_\ell, s'_1, \ldots, s'_m)$, and s' is a *subsequence* of s, denoted $s' \sqsubseteq s$ if there exist integers $1 \leq j_1 < \ldots < j_m \leq \ell$ with $(s_{j_1}, \ldots, s_{j_m}) = (s'_1, \ldots, s'_m)$. Such an integer sequence $(j_1, \ldots, j_m)$ is then called an *embedding* of s' into s.[4]

Finally, for a function $q : \Sigma \to \mathbb{Z}_{\geq 0}, \sigma \mapsto q_\sigma$, we say that $s = (s_1, \ldots, s_\ell) \in \Sigma^*$ has *characteristic* (at most) q if $|\{i \text{ s.t. } s_i = \sigma\}| = q_\sigma$ $(\leq q_\sigma)$ for any $\sigma \in \Sigma$.

Lemma 5.4 (Embedding Lemma)**.** *Let* Σ *be an alphabet of size* n*, and let* $q : \Sigma \to \mathbb{Z}_{\geq 0}$*,* $\sigma \mapsto q_\sigma$*. Then, there exists a string* $s \in \Sigma^*$ *with characteristic* $n \cdot q : \sigma \mapsto n \cdot q_\sigma$ *such that any string* $s' \in \Sigma^*$ *with characteristic at most* q *is a subsequence of* s*, i.e.,* $s' \sqsubseteq s$*.*

The idea of the construction of the sequence s is quite simple: First, we evenly distribute $n \cdot q_\sigma$ copies of σ within the interval $(0, n]$ by "attaching" one copy of σ to every point in $(0, n]$ that is an integer multiple of $1/q_\sigma$ (see Fig. 5.2). Note that it may happen that different symbols are "attached" to the same point. Then, we walk along the interval from 0 and n and, one by one, collect the symbols we encounter in order to build up s' from left to right; in case we encounter a point with multiple symbols "attached" to it, we collect them in an arbitrary order.

It is then not too hard to convince yourself that this s indeed satisfies the claim. Namely, for any $s' = (s'_1, \ldots, s'_m)$ as considered, we can again walk along the interval from 0 and n, and we will then encounter all the symbols

[4] We use *string* and *sequence* interchangeably; however, following standard terminology, there is a difference between a *substring* and *subsequence*: namely, a substring is a subsequence that admits an embedding with $j_{i+1} = j_i + 1$.

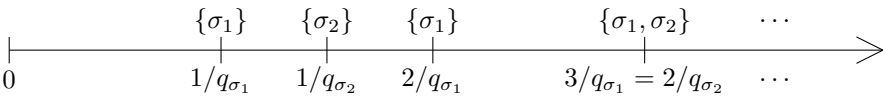


Figure 5.2: Constructing the string s by distributing the different symbols evenly within the interval $(0, n]$ (here with $3/q_{\sigma_1} = 2/q_{\sigma_2}$), and then collecting them from left to right.

of s', one by one: we will encounter the symbol s'_1 within the walk from 0 to $1/q_{s'_1}$, the symbol s'_2 then within the walk from $1/q_{s'_1}$ to $1/q_{s'_1} + 1/q_{s'_2}$, etc.[5]

Putting this idea into a formal proof is somewhat tedious, but in the end not too difficult. In order to formalize things properly, we generalize the standard notion of a sequence $s \in \Sigma^*$ in a way that allows us to talk about "attaching" a symbol to a point on $\mathbb{R}$, etc., in a rigorous way. Formally, we define a *line sequence* to be an arbitrary finite (possibly empty) subset $S \subseteq \mathbb{R} \times \Sigma$, i.e.,

$$S = \{(t_1, s_1), \ldots, (t_\ell, s_\ell)\} \in \mathcal{P}_{<\infty}(\mathbb{R} \times \Sigma)\,,$$

where w.l.o.g. we will always assume that $t_1 \leq \ldots \leq t_\ell$. We may think of the symbol s_i to "occur at the time" t_i.[6] For a subset $T \subset \mathbb{R}$, the set $\mathcal{P}_{<\infty}(T \times \Sigma)$ then obviously denotes the set of line sequences with $t_1, \ldots, t_\ell \in T$.

Assuming that the alphabet Σ is equipped with a total order $\leq$, any line sequence $S = \{(t_1, s_1), \ldots, (t_\ell, s_\ell)\}$ is naturally associated with the ordinary sequence

$$\pi(S) := (s_1, \ldots, s_\ell) \in \Sigma^*\,,$$

which is uniquely determined by the convention $t_1 \leq \ldots \leq t_\ell$ and insisting on $s_i \leq s_j$ whenever $t_i = t_j$ for $i < j$.

This *projection* $\pi : \mathcal{P}_{<\infty}(\mathbb{R} \times \Sigma) \to \Sigma^*$ preserves the characteristic of the sequence, i.e., if $s = (s_1, \ldots, s_\ell) = \pi(S)$ then

$$|\,\{t \text{ s.t. } (t, \sigma) \in S\}\,| = |\,\{i \text{ s.t. } s_i = \sigma\}\,| \tag{5.1}$$

for any $\sigma \in \Sigma$. Furthermore, for $T, T' \subset \mathbb{R}$ with $T < T'$ point-wise, and for $S \in \mathcal{P}_{<\infty}(T \times \Sigma)$ and $S' \in \mathcal{P}_{<\infty}(T' \times \Sigma)$, it is easy to see that $\pi(S \cup S') = \pi(S) \| \pi(S')$, from which it then follows that for ordinary sequences $s, s' \in \Sigma^*$

$$s \sqsubseteq \pi(S) \,\wedge\, s' \sqsubseteq \pi(S') \implies s \| s' \sqsubseteq \pi(S) \| \pi(S') = \pi(S \cup S')\,. \tag{5.2}$$

A final, simple observation, which follows directly from the definitions, is that

[5] To avoid the obvious issue of division by zero, we assume without loss of generality that each $q_\sigma > 0$.

[6] Note that we allow $t_i = t_j$ for $i \neq j$ while the definition prohibits $(t_i, s_i) = (t_j, s_j)$. If desired, one could allow the latter by letting S be a multi-set, but this is not necessary for us.

for $\sigma \in \Sigma$, i.e. a sequence of length $m = 1$, $\sigma \sqsubseteq \pi(S)$ holds if and only if there exists a time $t \in \mathbb{R}$ such that $(t, \sigma) \in S$.

Proof of Lemma 5.4. For any symbol $\sigma \in \Sigma$ let S_σ be a line sequence

$$S_\sigma := \left\{ \tfrac{1}{q_\sigma}, ..., \tfrac{nq_\sigma}{q_\sigma} \right\} \times \{\sigma\} \in \mathcal{P}_{<\infty}((0, n] \times \Sigma)\,,$$

and set $S := \bigcup_{\sigma \in \Sigma} S_\sigma$. We will show that $s := \pi(S)$ is as claimed.

The claim on the characteristic of s follows from the preservation of the characteristic under π, i.e. (5.1), and from $|\,\{t \text{ s.t. } (t, \sigma) \in S\}\,| = |\,S_\sigma\,| = n \cdot q_\sigma$, which holds by construction of S.

Let $s' = (s'_1, \ldots, s'_m) \in \Sigma^*$ be arbitrary with characteristic bounded by q. We consider the times $\tau_j := 1/q_{s'_1} + \cdots + 1/q_{s'_j}$ for $j \in \{1, \ldots, m\}$, and we let T_j be the interval

$$T_j := \big(\tau_{j-1}, \tau_j\big] = \big(\tau_{j-1}, \tau_{j-1} + \tfrac{1}{q_j'}\big] \subset \mathbb{R}\,,$$

and decompose $S = S_1 \cup \ldots \cup S_m$ with $S_j := S \cap (T_j \times \Sigma) \in \mathcal{P}_{<\infty}(T_j \times \Sigma)$. Here, we exploit that

$$\tau_m = \sum_{\sigma \in \Sigma} \frac{|\,\{i \text{ s.t. } s'_i = \sigma\}\,|}{q_\sigma} \leq \sum_{\sigma \in \Sigma} \frac{q_\sigma}{q_\sigma} = n\,,$$

and so the S_j's indeed cover all of $S \in \mathcal{P}_{<\infty}((0, n] \times \Sigma)$. Given that the interval $T_j \subset (0, n]$ has size $1/q_{s'_j}$, there exists a time $t_j \in T_j \cap \left\{ \tfrac{1}{q_\sigma}, ..., \tfrac{nq_\sigma}{q_\sigma} \right\}$. But then, $(t_j, s'_j) \in S_j$ by construction of S, and therefore $s'_j \sqsubseteq \pi(S_j)$. Finally, since $T_{j-1} < T_j$, property (5.2) implies that

$$s' = s'_1 \| \cdots \| s'_m \sqsubseteq \pi(S_1 \cup \ldots \cup S_m) = s$$

which was to be shown. □

While Lemma 5.4 above settles the existence question, the following two observations settle the corresponding efficiency questions. For concreteness, we assume $\Sigma = \{1, \ldots, n\}$ below, and thus can identify the function $q : \Sigma \to \mathbb{Z}_{\geq 0}$, $\sigma \mapsto q_\sigma$ with the vector $\mathbf{q} = (q_1, \ldots, q_n)$.

First, we observe that the line sequence S defined in the proof above, as well as its projection $s = \pi(S)$, can be computed in polynomial time in $q_1 + \cdots + q_n$; thus, we have the following.

Lemma 5.5. *There exists a polynomial-time algorithm that, on input $1^{\mathbf{q}}$, computes a string $s \in \Sigma^*$ as specified in the proof of Lemma 5.4.*

Furthermore, for any $s' \in \Sigma^*$ with characteristic at most q, for which we then know by Lemma 5.4 that s' can be embedded into s, the following ensures that this embedding can be computed efficiently and *on the fly*.

Lemma 5.6. *There exists a polynomial-time algorithm $\mathcal{E}$ such that for every string $s \in \Sigma^*$ and every subsequence $s' = (s'_1, \ldots, s'_m) \sqsubseteq s$, the following holds. Computing inductively $j_i \leftarrow \mathcal{E}(s, s'_i, j_{i-1})$ for every $i \in [m]$, where $j_0 := 0$, results in an increasing sequence $j_1 < \cdots < j_m$ with*

$$s' = (s_{j_1}, \ldots, s_{j_m}) \; .$$

The algorithm $\mathcal{E}$ simply follows the obvious greedy strategy: for each s'_i it looks for the next j_i for which $s'_i = s_{j_i}$. More formally:

Proof. The algorithm $\mathcal{E}(s, s'_i, j_{i-1})$ computes

$$j_i := \min \{k \in \mathbb{Z}_{>0} \,|\, j_{i-1} < k \leq m, s_k = s'_i\} \; . \tag{5.3}$$

It can be easily shown that the minimum is well-defined, i.e. taken over a non-empty set for each i by the assumption that s' is a subsequence of s, and thus by construction, every j_i is such that $s'_i = s_{j_i}$ while keeping $j_1 < \cdots < j_n$ increasing. This concludes the proof. □

5.3.3 Wrapping up the Proof of Theorem 5.1

The claimed interactive oracle algorithm $\mathcal{B}$ now works in the obvious way. On input $\mathbf{q}$ (provided in unary) and for any $\mathcal{A}$, $\mathcal{B}[\mathcal{A}]$ will make static oracle queries to $\mathcal{O}_{s_1}, \mathcal{O}_{s_2}, \ldots, \mathcal{O}_{s_{nQ}}$, where $s = (s_1, \ldots, s_{nQ}) \in \{1, \ldots, n\}^*$ is the string promised to exist by Lemma 5.4, with $Q = q_1 + \cdots + q_n$. In more detail, it first computes s using the algorithm from Lemma 5.5. Then, for the ith oracle query that $\mathcal{B}$ receives from $\mathcal{A}$ (starting with $i = 1$), and which consists of the identifier $s'_i \in \{1, \ldots, n\}$ of which oracle to query now and of the actual input to the oracle $\mathcal{O}_{s'_i}$, the algorithm $\mathcal{B}$ does the following: it computes $j_i \leftarrow \mathcal{E}(s, s'_i, j_{i-1})$ using the algorithm from Lemma 5.6, makes dummy queries to $\mathcal{O}_{s_{j_{i-1}+1}}, \ldots, \mathcal{O}_{s_{j_i-1}}$, and forwards $\mathcal{A}$'s query input to $\mathcal{O}_{s_{j_i}} = \mathcal{O}_{s'_i}$. The fact that $(j_1, \ldots, j_Q)$ computed this way forms an embedding of $s' = (s'_1, \ldots, s'_Q)$ into s ensures that $\mathcal{B}$ is able to forward all the queries that $\mathcal{A}$ makes to the right oracle, and so $\mathcal{A}$ will produce its output as in an ordinary run with direct adaptive access to the oracles.

5.3.4 Applications

To demonstrate the usefulness of our adaptive-to-static compiler, we briefly discuss three results from the literature. For two of them, the adaptivity of the attacker was explicitly declared as an obstacle in the security proof, and dealing with it complicated the proof substantially. These complications could be avoided/removed by means of our adaptive-to-static compiler. For the third one, we can immediately strengthen one of the results, which is restricted to

hold for static multi-oracle adversaries, by dropping this restriction via our compiler.

Quantum security of qTESLA. Our first application is in the context of qTESLA [ABB+20], which is a signature scheme that made it into the second round of the NIST post-quantum competition. Its security is based on the Ring-LWE problem, to which the authors of [ABB+17] give a reduction in the quantum random oracle model (QROM).[7] In the reduction, which starts from the security notion of *Unforgeability under Chosen Message Attack* (UF-CMA), the adversary can query a random oracle H as well as a signing oracle, where the order of oracle queries may be adaptive.

The reduction strategy of [ABB+17] applies only to a static adversary, with a fixed query pattern. Thus, the authors first compile the adaptive into a naive static attacker by letting it do q_H (the number of H-queries of the original adaptive adversary) H-queries between any two signing queries. Leaving it with this would blow up the number of H-queries to $q_S q_H$. In order to avoid that, they give the attacker a "*live-switch*", meaning that each query to H may be in superposition of making the query and not making the query, and the total "*query magnitude*" on actual H-queries is still restricted to q_H. Not so surprising, adding even more "quantumness" to the problem in this way, makes the analysis more complicated (compared to using standard "all-or-nothing" static queries and a standard classical bound on the query complexity), but it allows the authors to avoid the above blow-up in the (classical) query complexity to transpire into the security loss. The overall loss they obtain in the end is $O((q_S q_H^2 + q_S^3 + q_S^2 q_H) \cdot \epsilon)$ for small ϵ determined by the parameters of the scheme.

Since the security reduction in [ABB+17] intertwines the adaptive to static hurdle with other aspects of the proof, we cannot simply insert our Theorem 5.1 and then continue the proof as is. Still, by applying our result, we could obtain a static adversary with almost no cost in the number of H-queries, avoiding the need for the rather complicated "live-switch superposition" attacker, thus simplifying the overall proof significantly. Furthermore, looking ahead at Section 5.4, our result allows us to obtain the much better $O(\sqrt{q_O q_H^2 \epsilon} + \sqrt{q_O^2 q_H \epsilon})$ loss in a similar context — similar in the sense that it also involves two oracles where one reprograms the other at some high-entropy input. The adaptive to static reduction there allows us to apply some additional QROM tools that could potentially also be applied in the setting of qTESLA to improve the bound. However, actually doing this would require us to rewrite the entire proof of [ABB+17], which we consider outside the scope of this work.

[7] We note that some versions of qTESLA have been broken [LS19], but the attack only applies to an optimized variant that was developed for the NIST-competition, and does not apply to the scheme in [ABB+17] that we discuss here.

Quantum security of the function FX. Our second application is to [JST21], where the post-quantum security of the FX key-length extension is studied (which is a generalization of the Even-Mansour cipher). In a first part, post-quantum security of FX is shown under the restriction that the inputs to the queries are fixed in advance. In a second part, towards avoiding this restriction, the authors consider a variation of the FX construction, which they call FFX (for "function FX"), and they show in their Theorem 3 post-quantum security of FFX under the restriction that the attacker is "*order consistent*", as they call it in [JST21], which is precisely our notion of a *static* multi-oracle algorithm. Thus, by a direct application of our Theorem 5.1, this restriction can be dropped (almost) for free, i.e., with a small constant blow-up on the attackers advantage.

Quantum security of the Even-Mansour cipher. The work [ABKM22] shows full post-quantum security of the (unmodified) Even-Mansour cipher. As in the case of qTESLA, the fact that the attacker can choose adaptively whether to query the public permutation of the cipher complicates the proof. Indeed, as is explained on the 4th page in [ABKM22], this adaptivity issue forces the authors to extend the blinding lemma of Alagic *et al.* to a variant that gives a bound in terms of the *expected* number of queries. While the authors succeed in providing such an extended version of the blinding lemma (Lemma 3 in [ABKM22]), it further increases the complexity of an already involved proof.[8]

Thus, again, our Theorem 5.1 could be used to simplify the given proof by bypassing the complications that arise due to the attacker choosing adaptively which oracle to query at what point.

5.4 Quantum Security of a Split-key PRF

5.4.1 Hybrid Security and skPRFs

A *split-key pseudorandom function* (skPRF), as introduced in [GHP18], is a polynomial-time computable function $F : \mathcal{K}_1 \times \cdots \times \mathcal{K}_n \times \mathcal{X} \to \mathcal{Y}$ that is a pseudorandom function (PRF) in the standard sense *for every* $i \in [n]$ when considered as a keyed function with key space $\mathcal{K}_i$ and message space $\mathcal{K}_1 \times \cdots \times \mathcal{K}_{i-1} \times \mathcal{K}_{i+1} \times \cdots \times \mathcal{K}_n \times \mathcal{X}$, with the additional restriction that the distinguisher $\mathcal{A}$ (in the standard PRF security definition) must use a fresh $x \in \mathcal{X}$ in every query $(k_1, \ldots, k_{i-1}, k_{i+1}, \ldots, k_n, x)$.

[8] To be fully precise, Lemma 3 in [ABKM22] also generalizes the original blinding lemma in a different direction by allowing to reprogram to an arbitrary value instead of a uniformly random one; however, this generalization comes for free in that the original proof still applies up to obvious changes, while allowing an expected number of queries, which is needed to deal with the adaptivity issue, requires a new proof.

This restriction on the PRF distinguisher may look artificial, but is motivated by this definition of a skPRF being good enough for the intended purpose of a skPRF, namely to give rise to a *secure KEM combiner*. Indeed, [GHP18] shows that the naturally combined KEM, obtained by concatenating the individual ciphertexts to $C = (c_1, \ldots, c_n)$, and combining the individual session keys $k_1, \ldots, k_n$ using the above mentioned skPRF as

$$K = F(k_1, \ldots, k_n, C)\,,$$

is IND-CCA secure if at least one of the individual KEM's is IND-CCA secure.

The paper [GHP18] also proposes a particularly efficient hash-based construction, given by

$$F(k_1, \ldots, k_n, x) := H(g(k_1, \ldots, k_n), x) \tag{5.4}$$

where $g : \mathcal{K}_1 \times \cdots \times \mathcal{K}_n \to \mathcal{W}$ is a polynomial-time mapping with the property that, for some small ϵ,

$$\Pr_{k_i \leftarrow \mathcal{K}_i}[g(k_1, \ldots, k_n) = w] \leq \epsilon\,, \tag{5.5}$$

for every $i \in [n]$ and for every $k_1, \ldots, k_{i-1}, k_{i+1}, \ldots, k_n$ and every w; furthermore, $H : \mathcal{W} \to \mathcal{Y}$ is a cryptographic hash function. Simple choices for the function g are $g(k_1, \ldots, k_n) = (k_1, \ldots, k_n)$ and $g(k_1, \ldots, k_n) = k_1 + \cdots + k_n$.

It is shown in [GHP18] that this construction is a skPRF when H is modelled as a random oracle; indeed, it is shown that the distinguishing advantage is upper-bounded by $q_H\epsilon$, where q_H is the number of queries to the random oracle H.

Given the natural use of combiners in the context of the upcoming transition to post-quantum cryptography, it is natural — and well-motivated — to ask whether F can be proven to be a skPRF in the presence of a *quantum attacker*, i.e., when H is modeled as a *quantum* random oracle. Below, we answer this in the affirmative.

5.4.2 Quantum-security of the skPRF

The goal of this section is to show the security of the skPRF (5.4) in the quantum random oracle model. In essence, this requires proving that F is a PRF (in the quantum random oracle model) with respect to *any* of the k_i's being the key, subject to the restriction of asking a fresh x in each query.

To simplify the notation, we fix the index $i \in [n]$ and simply write k for k_i and x for $(k_1, \ldots, k_{i-1}, k_{i+1}, \ldots, k_n, x)$, and we abstract away the properties of the function g as follows. We let

$$F(k, x) := H(h(k, x))\,,$$

where $h : \mathcal{K} \times \mathcal{X} \to \mathcal{W}$ is an arbitrary function with the property that, for some parameter $\epsilon > 0$,

$$\Pr_{k \leftarrow \mathcal{K}}[h(k, x) = w] \leq \epsilon \tag{5.6}$$

for all $w \in \mathcal{W}$ and $x \in \mathcal{X}$. Furthermore, in the PRF security game, we restrict the attacker/distinguisher $\mathcal{A}$ to queries x with a fresh value of $h(k, x)$, no matter what k is.

More formally, let $\mathcal{A}^{\mathcal{H},\mathcal{O}}$ be an arbitrary quantum oracle algorithm, making quantum superposition queries to an oracle $\mathcal{H}$ and classical queries to another oracle $\mathcal{O}$, with the restriction that for every query x to $\mathcal{O}$ it holds that

$$h(\kappa, x) \neq h(\kappa, x') \ , \tag{5.7}$$

for any prior query x' to $\mathcal{O}$ and all $\kappa \in \mathcal{K}$. For any such oracle algorithm $\mathcal{A}^{\mathcal{H},\mathcal{O}}$, we consider the standard PRF security games

$$\mathsf{PR}^1 := \mathcal{A}^{H,F} \qquad \text{and} \qquad \mathsf{PR}^0 := \mathcal{A}^{H,R} \, ,$$

obtained by instantiating $\mathcal{H}$ with a random function H (the random oracle) in both games, and in one game we instantiate $\mathcal{O}$ with the pseudorandom function F, which we understand to return $F(k, x)$ on query x for a random $k \leftarrow \mathcal{K}$, chosen once and for all queries, and in the other we instantiate $\mathcal{O}$ with a truly random function R instead.

We show that the distinguishing advantage for these two games is bounded as follows.

Theorem 5.7. *Let $\mathcal{A}^{\mathcal{H},\mathcal{O}}$ be a (q_H, q_F)-query oracle algorithm satisfying (5.7). Then*

$$\left|\Pr\left[1 \leftarrow \mathsf{PR}^1\right] - \Pr\left[1 \leftarrow \mathsf{PR}^0\right]\right| \leq 4\sqrt{2q_F^2 q_H \epsilon} + 4\sqrt{2q_H^2 q_F \epsilon} \, .$$

We can now apply Theorem 5.7 to the function $h(k, x) := (g(k_1, \ldots, k_n), \tilde{x})$, where $k := k_i$ and $x := (k_1, \ldots, k_{i-1}, k_{i+1}, \ldots, k_n, \tilde{x})$. Indeed, the condition (5.5) on g implies the corresponding condition (5.7) on h, and the restriction on $\tilde{x}$ being fresh in the original skPRF definition implies the above restriction on $h(k, x)$ being fresh no matter what k is, i.e., 5.6). Thus, we obtain the following.

Corollary 5.8. *For any function g satisfying (5.5) for a given $\epsilon > 0$, the function $F(k_1, \ldots, k_n, x) := H(g(k_1, \ldots, k_n), x)$ is a skPRF in the quantum random oracle model with distinguishing advantage at most $4\sqrt{2q_F^2 q_H \epsilon} + 4\sqrt{2q_H^2 q_F \epsilon}$.*

5.4.3 Proof of Theorem 5.7

Proof (of Theorem 5.7). Let $\mathcal{A}^{\mathcal{H},\mathcal{O}}$ be an oracle algorithm as considered in the previous subsection. Thanks to Theorem 5.1, taking a factor-2 blow-up in the

query complexity into account, we may assume $\mathcal{A}$ to be a *static* (q_H, q_F)-query oracle algorithm. It will be convenient to write such a static algorithm as

$$\mathcal{A}^{[\mathbf{H}_0 \mathcal{O} \mathbf{H}_1 \mathcal{O} \mathbf{H}_2 \ldots \mathcal{O} \mathbf{H}_{q_F}]} ,$$

where each block $\mathbf{H}_i = \mathcal{H} \cdots \mathcal{H}$ consists of a (possibly empty) sequence of symbols $\mathcal{H}$ of length $q_i^{\mathcal{H}} = |\mathbf{H}_i|$, and with the understanding that $\mathcal{A}$ first makes $q_0^{\mathcal{H}}$ queries to $\mathcal{H}$, then a query to $\mathcal{O}$, then $q_1^{\mathcal{H}}$ queries to $\mathcal{H}$, etc., where, obviously, $q_0^{\mathcal{H}} + \cdots + q_{q_F}^{\mathcal{H}} = q_H$ then. Instantiating $\mathcal{H}$ with H, and $\mathcal{O}$ with F and R, respectively, we can then write

$$\mathsf{PR}^0 = \mathcal{A}^{[\mathbf{H}_0 R \mathbf{H}_1 \ldots R \mathbf{H}_{q_F}]} \qquad \text{and} \qquad \mathsf{PR}^1 = \mathcal{A}^{[\mathbf{H}_0 F \mathbf{H}_1 \ldots F \mathbf{H}_{q_F}]} .$$

For the proof, we introduce certain hybrid games. For this purpose, we introduce the following alternative (*stateful* and R-dependent) instantiation H' of $\mathcal{H}$. To start with, H' is set to be equal to H, but whenever R is queried on some input x, H' is *reprogrammed* at the point $h(k, x)$ to the value $H'(h(k, x)) := R(x)$. For any i, we now define the two hybrid games

$$\mathsf{PR}_i^2 := \mathcal{A}^{[\mathbf{H}_0 R \ldots R \mathbf{H}_i F \mathbf{H}'_{i+1} F \ldots F \mathbf{H}'_{q_F}]}$$
$$\widetilde{\mathsf{PR}}_i^2 := \mathcal{A}^{[\mathbf{H}_0 R \ldots R \mathbf{H}_i R \mathbf{H}'_{i+1} F \ldots F \mathbf{H}'_{q_F}]}$$

and also spell out

$$\mathsf{PR}_{i+1}^2 = \mathcal{A}^{[\mathbf{H}_0 R \ldots R \mathbf{H}_i R \mathbf{H}_{i+1} F \ldots F \mathbf{H}'_{q_F}]}$$

to emphasize its relation to $\widetilde{\mathsf{PR}}_i^2$. We note that in all of the above, the first occurrences of $\mathcal{H}$ and $\mathcal{O}$ are instantiated with R and H, respectively, but at some point we switch to R and H' instead.

The extreme cases match up the games we are interested in. Indeed,

$$\mathsf{PR}_0^2 = \mathcal{A}^{[\mathbf{H}_0 F \mathbf{H}'_1 \ldots F \mathbf{H}'_{q_F}]} = \mathcal{A}^{[\mathbf{H}_0 F \mathbf{H}_1 \ldots F \mathbf{H}_{q_F}]} = \mathsf{PR}^1 ,$$

where we exploit that there are no queries to R and thus H' remains equal to H, and, by definition,

$$\mathsf{PR}_{q_F}^2 = \mathcal{A}^{[\mathbf{H}_0 R \mathbf{H}_1 \ldots R \mathbf{H}_{q_F}]} = \mathsf{PR}^0 .$$

Our goal is to prove the closeness of the following games

$$\mathsf{PR}^1 = \mathsf{PR}_0^2 \approx \widetilde{\mathsf{PR}}_0^2 \approx \mathsf{PR}_1^2 \cdots \approx \mathsf{PR}_{q_F-1}^2 \approx \widetilde{\mathsf{PR}}_{q_F-1}^2 \approx \mathsf{PR}_{q_F}^2 = \mathsf{PR}^0 .$$

We do this by means of applying Lemma 5.9 and 5.10, which we state here and prove further down.

Lemma 5.9. *For each* $0 \le i < q_F$,

$$\left|\Pr\left[1 \leftarrow \mathsf{PR}_i^2\right] - \Pr\left[1 \leftarrow \widetilde{\mathsf{PR}}_i^2\right]\right| \le 2\sqrt{\sum_{1\le j\le i} q_j^{\mathcal{H}}\epsilon}\,.$$

Lemma 5.10. *For each* $0 \le i < q_F$,

$$\left|\Pr\left[1 \leftarrow \widetilde{\mathsf{PR}}_i^2\right] - \Pr\left[1 \leftarrow \mathsf{PR}_{i+1}^2\right]\right| \le 2q_{i+1}^{\mathcal{H}}\sqrt{q_F\epsilon}\,.$$

Indeed, by repeated applications of these lemmas, and additionally using that $q_0^{\mathcal{H}} + \cdots + q_i^{\mathcal{H}} \le q_H$ for all $0 \le i \le q_F$, we obtain

$$\begin{aligned}\left|\Pr\left[1 \leftarrow \mathsf{PR}^1\right] - \Pr\left[1 \leftarrow \mathsf{PR}^0\right]\right| &\le 2\sum_{i=0}^{q_F}\sqrt{\sum_{1\le j\le i} q_j^{\mathcal{H}}\epsilon} + 2\sum_{i=0}^{q_F} q_{i+1}^{\mathcal{H}}\sqrt{q_F\epsilon}\\ &\le 2\sqrt{q_F^2 q_H\epsilon} + 2\sqrt{q_H^2 q_F\epsilon}\end{aligned}$$

which concludes the claim of Theorem 5.7 when incorporating the factor-2 increase in q_H and q_F due to switching to a static $\mathcal{A}$. □

It remains to prove Lemma 5.9 and 5.10, which we do below. In both proofs, we use the *gentle measurement lemma* [Wil11, Lemma 9.4.1], which states that if a projective measurement has a very likely outcome then the measurement causes only little disturbance on the state. More formally, for any density operator ρ and any projector P, where $p := \mathsf{tr}(P\rho P)$ then is the probability to observe the outcome associated with P when measured using the measurement $\{P, \mathbb{I} - P\}$, the trace distance between the original state ρ and the post-measurement state $\rho' := P\rho P/p$ is bounded by $\sqrt{1-p}$. This in turn implies that ρ and ρ' can be distinguished with an advantage $\sqrt{1-p}$ only.

The proof of Lemma 5.9 additionally makes use of Zhandry's compressed oracle technique [Zha19]. It is out of scope of this work to give a self-contained description of this technique; we refer to the original work [Zha19] instead, or to [CFHL21], which offers an alternative concise description. At the core is the observation that one can *purify* the random choice of the function H and then, by switching to the Fourier basis and doing a suitable measurement, one can check whether a certain input x has been "recorded" in the database (mind though that such a measurement disturbs the state). If the outcome is negative then the oracle is still in a uniform superposition over all possible hash values for x, and as a consequence, when removing the purification by doing a full measurement of H (in the computational basis), $H(x)$ is ensured to be a "fresh" uniformly random value, with no information on $H(x)$ having been leaked in prior queries.

In the proof of Lemma 5.9, we use this technique to check whether *prior*

to the crucial query, which is to F in one and to R in the other game, there was a query to H that would reveal the difference, and we use (5.6) to argue that it is unlikely that such a query occurred. Since this measurement has a likely outcome, it is also ensured by the gentle measurement lemma that this measurement causes little disturbance.

Proof (of Lemma 5.9). For convenience, we refer to the *crucial query* as the respective query to F and R that differs between

$$\mathsf{PR}_i^2 = \mathcal{A}^{[\mathbf{H}_0 R \ldots R \mathbf{H}_i F \mathbf{H}'_{i+1} F \ldots F \mathbf{H}'_{q_F}]} \quad \text{and} \quad \widetilde{\mathsf{PR}}_i^2 = \mathcal{A}^{[\mathbf{H}_0 R \ldots R \mathbf{H}_i R \mathbf{H}'_{i+1} F \ldots F \mathbf{H}'_{q_F}]}.$$

Furthermore, we let x be the input to that query, and we set $w := h(k, x)$, with k being the key chosen and used by F. Note that up to this very query, the two games are identical. Also, by (5.7) it is ensured that for any prior query x' to R it holds that $h(k, x') \neq w$.

First, we consider the games $\mathbf{G}^1$ and $\widetilde{\mathbf{G}}^1$ that work exactly as PR_i^2 and $\widetilde{\mathsf{PR}}_i^2$, respectively, except that, at the beginning of the games we set up the compressed oracle and answer all queries made to H prior to the crucial query using the compressed oracle. Then, once x is received during the crucial query, we do a full measurement of the purified (i.e. uncompressed) oracle in order to obtain the function H, which is then to be used in the remainder of the games. We note that setting up the function H' is then necessarily also deferred to after this measurement, where H' is then set to be equal to H, except that for any prior query x' to R it is reprogrammed to $H'(h(k, x')) := R(x')$. Only once H has been measured and H' set up as above, is the crucial query then actually answered.

It follows from basic properties of the compressed oracle that the respective output distributions of $\mathbf{G}^1$ and $\widetilde{\mathbf{G}}^1$ match with those of PR_i^2 and $\widetilde{\mathsf{PR}}_i^2$.

Then, we define $\mathbf{G}^2$ and $\widetilde{\mathbf{G}}^2$ from $\mathbf{G}^1$ and $\widetilde{\mathbf{G}}^1$, respectively, by introducing one more measurement. Namely, right after x is sent by $\mathcal{A}$ and before H is measured, we measure in the compressed oracle whether the input $w = h(k, x)$ has been recorded in the database, and in case of a positive outcome, the game aborts. By the gentle measurement lemma (and basic properties of the trace distance),

$$\left|\Pr\left[1 \leftarrow \mathbf{G}^1\right] - \Pr\left[1 \leftarrow \mathbf{G}^2\right]\right| \leq \sqrt{\Pr\left[\mathbf{G}^2 \text{ aborts}\right]}$$

and similarly for $\widetilde{\mathbf{G}}^1$ and $\widetilde{\mathbf{G}}^2$, where $\widetilde{\mathbf{G}}^2$ aborts with the same probability as $\mathbf{G}^2$.

By basic properties, after $t := q_0^{\mathcal{H}} + \cdots + q_i^{\mathcal{H}}$ queries to the compressed oracle, no more than t values have been recorded. I.e., if we were to measure, for the sake of the argument, the entire compressed oracle to obtain the full database D, it would hold that $\mathrm{supp}(D) := \{u \,|\, D(u) \neq \bot\}$ has cardinality

at most t. Since k has not been used yet and so is still freshly random (i.e., independent of x and D), the high-entropy condition (5.6) then ensures that

$$\Pr\big[\widetilde{\mathbf{G}}^2 \text{ abort}\big] = \Pr\big[\mathbf{G}^2 \text{ abort}\big] = \Pr\big[w \in \mathrm{supp}(D)\big] \leq \sum_{j<i} q_j^{\mathcal{H}} \epsilon\,.$$

It remains to show that $\mathbf{G}^2$ and $\widetilde{\mathbf{G}}^2$ behave identically conditioned on not aborting. The only difference between the two games is that in $\mathbf{G}^2$ the crucial query is answered with $y := H(h(k,x)) = H(w)$ and H' is *not* reprogrammed at the point w, while in $\widetilde{\mathbf{G}}^2$ the crucial query is answered with $y := R(x)$ and H' *is* reprogrammed at the point w to $H'(w) := R(x)$. We argue that this difference is not noticeable by $\mathcal{A}$.

First, we note that y is a fresh random value in both games. In the former game it is because, conditioned on not aborting, the compressed oracle at the register $h(k,x)$ is $\perp$, and so when uncompressing and measuring to obtain H, the hash value $H(w)$ will be a fresh random value. In the latter game it is because $R(x)$ is a truly random function and, due to (5.7), x has not been queried to R before.

Second, we observe that $y = H'(w)$ in both games. Indeed, in $\widetilde{\mathbf{G}}^2$ this holds by definition; in $\mathbf{G}^2$ it holds because $H'(w) = H(w)$, which follows from the fact that H' is reprogrammed only at points $w' = h(k,x')$ with x' being a prior query to R, but then (5.7) ensures that $w' \neq w$.

Thus, in both games, from $\mathcal{A}$'s perspective, the tuple $(k, y, H', H\backslash w)$ of random variables has the same distribution, where $H\backslash w$ refers to the function (table of) H but with the value at the point w removed. The only difference is that in one game $H'(w) = H(w)$ and in the other not (necessarily). However, the future behavior of $\mathcal{A}$ in both games only depends on $(k, y, H', H\backslash w)$, and thus $\mathcal{A}$ behaves the same way in both games. Here we are exploiting that the future hash queries by $\mathcal{A}$ are to H' (and not to H anymore), and, once more, we are using the restriction (5.7), here to ensure that for any future F-query x' by $\mathcal{A}$, it holds that $h(k,x') \neq w$, and thus the response does not depend on $H(w)$. Thus, $H(w)$ does indeed not affect $\mathcal{A}$'s behavior after the crucial query.

Exploiting that $\mathsf{PR}_i^2 = \mathbf{G}^1 \approx \mathbf{G}^2 = \widetilde{\mathbf{G}}^2 \approx \widetilde{\mathbf{G}}^1 = \widetilde{\mathsf{PR}}_i^2$, with the approximations bounded as discussed further up, we obtain the claimed closeness claim. This concludes the proof. □

Proof of Lemma 5.10. In order to show the closeness between $\widetilde{\mathsf{PR}}_i^2$ and PR_{i+1}^2, we define the intermediate games

$$\mathbf{G}_{i,j} := \mathcal{A}^{[\mathbf{H}_0 R \ldots \mathbf{H}_i R \mathbf{H}'_{i,j} \mathbf{H}_{i,j} F \ldots F \mathbf{H}'_{q_F}]}$$

for $0 \leq j \leq m := q_{i+1}^{\mathcal{H}}$, where $\mathbf{H}'_{i,j}$ and $\mathbf{H}_{i,j}$ consists of j and $m-j$ copies of

H' and H respectively. Note that for the extreme cases we have

$$\mathbf{G}_{i,0} = \widetilde{\mathsf{PR}}_i^2 \qquad \text{and} \qquad \mathbf{G}_{i,m} = \mathsf{PR}_{i+1}^2 \,.$$

Thus, it suffices to show closeness between $\mathbf{G}_{i,j}$ and $\mathbf{G}_{i,j+1}$ for any $0 \leq j < m$. Note that they only differ at one query, which is either to H' or to H, which we will refer to as the *crucial query* for convenience. In the remainder, i and j are arbitrary (in the considered ranges) but fixed.

Define the games $\widetilde{\mathbf{G}}^1$ and $\mathbf{G}^1$ from $\mathbf{G}_{i,j}$ and $\mathbf{G}_{i,j+1}$ respectively as follows. Let X be the set of queries x made to R prior to the crucial query, and set $S := \{h(k,x) \,|\, x \in X\}$. We then measure the crucial query, which may be in a superposition, with the binary measurement that checks whether the crucial query is an element of S, and we abort if this is the case.

In case of a negative outcome, i.e., the crucial query is *not* in S, there is no difference between the reply provided by H and by H', and thus there is no difference between the two games — and in case of a positive outcome, they both abort. In order to argue that this measurement causes little disturbance, we again use the gentle measurement lemma to argue that

$$\left|\Pr\left[1 \leftarrow \mathbf{G}^1\right] - \Pr\left[1 \leftarrow \mathbf{G}_{i,j+1}\right]\right| \leq \sqrt{\Pr\left[\mathbf{G}^1 \text{ abort}\right]} \,,$$

and correspondingly for $\mathbf{G}_{i,j}$ and $\widetilde{\mathbf{G}}^1$. So it remains to bound the abort probability. For the purpose of the argument, let us do a full measurement of the query, and let w be the outcome. We note that k has not been used yet, and thus remains a fresh random key, independent of w and X. Thus, using (5.6),

$$\Pr\left[\mathbf{G}^1 \text{ abort}\right] = \Pr\left[\widetilde{\mathbf{G}}^1 \text{ abort}\right] = \Pr\left[w \in S\right] \leq \sum_{x \in X} \Pr\left[w = h(k,x)\right] \leq q_F \epsilon \,.$$

Adding up this error term over the sequence $\mathbf{G}_{i,0} \approx \cdots \approx \mathbf{G}_{i,m}$ of approximations, the proof is concluded. □

Conclusions

Provable security is at the heart of modern cryptography; it ensures that breaking the considered scheme is at least as difficult as solving a well-studied hard computational problem. However, this guarantee stands and falls with the correctness of the proof. Thus, it is essential that security proofs undergo sufficient scrutiny — and even then human errors may remain. For instance, the flaw in the original FSwA analysis not only remained unnoticed for over a decade, it reappeared in later, modified variants of the analysis.

In a field like cryptography (and maybe theoretical computer science in general), often there is no fully rigorous formalism available. Meanwhile, it has become standard to put major proofs into the appendices of submissions, which reviewers do not have to verify (and it remains unclear who will ever carefully verify them). For the two cases we discuss in this thesis (FSwA and BUFF), we were lucky enough to find alternative, correct proofs (albeit with a worse security loss and an adjustment to the security definition, respectively); in other cases we may not be as fortunate and possibly face "proven-secure" insecure schemes, if we do not pay sufficient attention to verifying security proofs.

In this context, it is also worth mentioning the importance of verifying security proofs using more reliable methods, e.g. mechanized proof checkers such as EasyCrypt. This has been a direction with many ongoing research efforts. In fact, the FSwA flaw, which reappeared in HSwA, was initially discovered in such a mechanized verification project, and parts of the new, fixed security proofs, as presented in Chapter 3, have also been verified using EasyCrypt.

Another takeaway is that, especially for new notions of security, formulating the "right" definition can be very non-trivial. For instance, in Chapter 4, we have spent a significant amount of effort investigating the achievability of different definitions of non-resignability (NR). Along the way, evidence is gradually accumulated regarding which definitions are reasonable, and which ones are not. However, there may potentially be other relevant aspects that we have not (extensively) treated in this thesis, such as practical applications of NR. Therefore, strictly speaking, the final call as to which definition is "right," is not ours to make. Like any newly established notion of security, it must be shaped and tested, over time, by the cryptographic community as a whole.

Appendix

A.1 Breaking a Weakened Phi-Non-Malleability

Following the (updated) definition in [CDF+23, Def. 2.4], a hash function H is Φ-non-malleable if for every pair $(\mathcal{D}, \mathcal{A})$ of PPT algorithms for which the HILL entropy $\mathsf{HILL}_\infty(x \mid \mathsf{st})$ is sufficiently large for $(\mathcal{X}, \mathsf{st}) \leftarrow \mathcal{D}$ and $x \leftarrow \mathcal{X}$, the probability of winning the game in Fig. A.1 is negligible. In case of a hash function family, the hash key is given as input to $\mathcal{D}$ and the HILL entropy is then also conditioned on the hash key. In case of H a random oracle, $\mathcal{D}$ is given query access to H and the entropy requirement is then on the statistical min-entropy $H_\infty(x \mid H, \mathsf{st})$, where one additionally conditions on the (function table of) the random oracle.

The relevant choice of Φ for the non-resignability claim in [CDF+23] is

$$\Phi = \{\phi_{\mathsf{pk}'} : (\mathsf{pk}, m) \mapsto (\mathsf{pk}', m) \,|\, \mathsf{pk}' \in \mathcal{PK}\}\,,$$

where $\mathcal{PK}$ is the space of all public keys. In more detail, [CDF+23, Lemma 5.7] shows that the considered NR property of the (original) BUFF transform is satisfied *if* the considered hash function H (or hash function family) satisfies the above notion of Φ-non-malleability for this particular choice of Φ.

$\Phi\text{-}\mathsf{NM}_0$:
1: $(\mathcal{X}, \mathsf{st}) \leftarrow \mathcal{D}$
2: $x \leftarrow \mathcal{X}$
3: $y := H(x)$
4: $(y', \phi) \leftarrow \mathcal{A}(y, \mathsf{st})$
5: **return** $(\, H(\phi(x)) = y' \,\wedge\, \phi(x) \neq x \,)$

Figure A.1: The Φ-non-malleability game, as considered in [CDF+23], but for a fixed hash function H. $\mathcal{X}$ is an efficiently sampleable distribution. The subscript in $\Phi\text{-}\mathsf{NM}_0$ here is meant to distinguish it from the original definition, which considers some additional auxiliary information.

We show here a simple attack against this notion of Φ-non-malleability for this choice of Φ. The attack applies to *any* hash function (family) H, including the random oracle, and so renders the non-resignability claim in [CDF+23, Lemma 5.7], and in [CDF+23, Theorem 5.5], vacuous.

The attack works as follows. $\mathcal{D}$ outputs the distribution $\mathcal{X}$ that samples a random $\mathsf{pk} \in \mathcal{PK}$ and outputs $x = (\mathsf{pk}, 0)$, and $\mathcal{A}$ ignores its input $y = H(\mathsf{pk}, 0)$ and simply outputs $(H(\mathsf{pk}', 0), \phi_{\mathsf{pk}'})$ for an arbitrary (fixed) $\mathsf{pk}' \in \mathcal{PK}$. Note that there is no state information st here, and the entropy condition is satisfied (assuming $\mathcal{PK}$ to be sufficiently large). Thus, this is a valid attack that succeeds with probability almost 1; it only fails when the random $\mathsf{pk} \in \mathcal{PK}$ happens to be pk'.

A.2 A Modified Measure-and-Reprogram Lemma

In [DFM20] we find the "measure-and-reprogram technique 2.0" (Theorem 2):

Theorem A.1 (Measure-and-reprogram)**.** *Let $\mathcal{X}$ and $\mathcal{Y}$ be finite non-empty sets. There exists a black-box two-stage quantum algorithm $\mathcal{S}$ with the following property. Let $\mathcal{A}$ be an arbitrary oracle quantum algorithm that makes q queries to a uniformly random $H : \mathcal{X} \to \mathcal{Y}$ and that outputs some $x \in \mathcal{X}$ and a (possibly quantum) output z. Then, the two-stage algorithm $\mathcal{S}^{\mathcal{A}}$ outputs some $x \in \mathcal{X}$ in the first stage and, upon a random $\Theta \in \mathcal{Y}$ as input to the second stage, a (possibly quantum) output z, so that for any $x_\circ \in \mathcal{X}$ and any (possibly quantum) predicate V:*

$$\Pr_{H,\Theta}\left[x = x_\circ \wedge V(x, \Theta, z) : (x, z) \leftarrow \langle \mathcal{S}^{\mathcal{A}}, \Theta \rangle\right] \geq \frac{1}{(2q+1)^2} \Pr_{H}\left[x = x_\circ \wedge V(x, H(x), z) : (x, z) \leftarrow \mathcal{A}^{H}\right].$$

Furthermore, $\mathcal{S}$ runs in time polynomial in q, $\log|\mathcal{X}|$, and $\log|\mathcal{Y}|$.

Here $\langle \mathcal{S}^{\mathcal{A}}, \Theta \rangle$ works as follows: First, one of the $q+1$ queries of $\mathcal{A}$ (also counting the final output in register X) is measured, and the measurement outcome x is output by (the first stage of) $\mathcal{S}$. Each of the q actual queries is picked with probability $\frac{2}{2q+1}$, while the final output is picked with probability $\frac{1}{2q+1}$. Then this very query of $\mathcal{A}$ is answered either using the original H *or* using the reprogrammed oracle $H{*}\Theta x$, with the choice being made at random[1], while all the remaining queries of $\mathcal{A}$ are answered using oracle $H{*}\Theta x$. Finally, (the second stage of) $\mathcal{S}$ outputs whatever $\mathcal{A}$ outputs.

The theorem follows directly from the above definition of $\mathcal{S}$ and a technical lemma. Let first $|\phi_i\rangle$ be defined as $\mathcal{A}$'s state right before making its $i+1$st

[1] If it is the final output that is measured then there is nothing left to reprogram, so no choice has to be made.

query — with the special case $|\phi_q\rangle$ denoting the final output state — to which we add the superscript $\mathcal{O} \in \{H, H*\Theta x\}$ when all previous queries have been answered using $\mathcal{O}$. Next, we use $\mathcal{A}^{\mathcal{O}}_{i\to j}$ to denote the unitary that brings $\mathcal{A}$ from $|\phi_i\rangle$ to $|\phi_j\rangle$, using $\mathcal{O}$ from the ith query on. Finally, we use the shorthand $X := |x\rangle\langle x|$. The lemma then reads:

Lemma A.2. *Let $\mathcal{A}$ be a q-query oracle quantum algorithm. Then, for any function $H : \mathcal{X} \to \mathcal{Y}$, any $x \in \mathcal{X}$ and $\Theta \in \mathcal{Y}$, and any projection $\Pi_{x,\Theta}$, it holds that*

$$\mathbb{E}_{i,b}\left[\left\|(X \otimes \Pi_{x,\Theta})\big(\mathcal{A}^{H*\Theta x}_{i+b\to q}\big)\big(\mathcal{A}^{H}_{i\to i+b}\big)X\left|\phi^H_i\right\rangle\right\|_2^2\right] \geq \frac{\left\|(X \otimes \Pi_{x,\Theta})\left|\phi^{H*\Theta x}_q\right\rangle\right\|_2^2}{(2q+1)^2},$$

where the expectation is over uniform $(i, b) \in (\{0, \ldots, q-1\} \times \{0, 1\}) \cup \{(q, 0)\}$.

Here the left-hand side corresponds to the success probability of $\mathcal{S}$ with respect to V and Θ, while (in expectation over Θ) the right-hand side is equal to the success probability of the adversary in a normal run, now with respect to V and the original oracle output $H(x)$.

A first observation is that the technical lemma actually proves something slightly stronger than Theorem A.1; If we let $\mathcal{S}$ additionally output the measurement outcome x, we get the condition $x = x'$ for free (since the same projector X is used on the query as well as the final output state). On the other hand, for our application it suffices to use a slightly weaker statement (in a different respect) that we obtain by summing over all $x_\circ \in \mathcal{X}$:

$$\begin{aligned}&\Pr_{\Theta}\big[x = x' \wedge V(x, \Theta, z) : (x, x', z) \leftarrow \langle \mathcal{S}^{\mathcal{A}}, \Theta\rangle\big]\\ &\geq \frac{1}{(2q+1)^2}\Pr_{H}\big[V(x, H(x), z) : (x, z) \leftarrow \mathcal{A}^H\big].\end{aligned}$$

Next, we will reduce q to account for only those queries where $\mathcal{S}$ has a non-zero probability of measuring the same $x' = x$ that will eventually be output by $\mathcal{A}$, while also satisfying the quantum predicate. The probability here is over the choice of H, Θ and the measurement outcome x', we thus define:

$$\begin{aligned}Q_{\min} := \{i \in \{0, \ldots, q-1\} \mid \exists H \in \mathcal{Y}^{\mathcal{X}}, \exists \Theta \in \mathcal{Y}, \exists x \in \mathcal{X}, \exists b \in \{0, 1\} \text{ s.t.}\\ \left\|(X \otimes \Pi_{x,\Theta})\big(\mathcal{A}^{H*\Theta x}_{i+b\to q}\big)\big(\mathcal{A}^{H}_{i\to i+b}\big)X\left|\phi^H_i\right\rangle\right\|_2 \neq 0\}.\end{aligned}$$

Let furthermore Q be any subset of queries such that $Q_{\min} \subseteq Q \subseteq \{0, \ldots, q-1\}$. It will now be easy to prove the following modified lemma:

Lemma A.3. *Let $\mathcal{A}$ be a q-query oracle quantum algorithm, with Q as defined above. Then, for any function $H\colon \mathcal{X} \to \mathcal{Y}$, any $x \in \mathcal{X}$ and $\Theta \in \mathcal{Y}$, and any*

projection $\Pi_{x,\Theta}$*, it holds that*

$$\mathbb{E}_{i,b}\left[\left\|(X\otimes\Pi_{x,\Theta})\left(\mathcal{A}^{H*\Theta x}_{i+b\to q}\right)\left(\mathcal{A}^{H}_{i\to i+b}\right)X\left|\phi^H_i\right\rangle\right\|_2^2\right]\geq\frac{\left\|(X\otimes\Pi_{x,\Theta})\left|\phi^{H*\Theta x}_q\right\rangle\right\|_2^2}{(2|Q|+1)^2}\,,$$

where the expectation is over uniform $(i,b)\in(Q\times\{0,1\})\cup\{(q,0)\}$*.*

Note that the only difference to Lemma A.2 is in the expectation on the left-hand side and the denominator on the right-hand side, as indicated in blue. The proof is largely taken from [DFM20], with a small modification which we highlight with a yellow background.

Proof. For any $0\leq i\leq q$, inserting a resolution of the identity and exploiting that

$$\left(\mathcal{A}^{H*\Theta x}_{i+1\to q}\right)\left(\mathcal{A}^{H}_{i\to i+1}\right)(\mathbb{I}-X)\left|\phi^H_i\right\rangle=\left(\mathcal{A}^{H*\Theta x}_{i\to q}\right)(\mathbb{I}-X)\left|\phi^H_i\right\rangle\,,$$

we can write

$$\begin{aligned}
&\left(\mathcal{A}^{H*\Theta x}_{i+1\to q}\right)\left|\phi^H_{i+1}\right\rangle\\
&\quad=\left(\mathcal{A}^{H*\Theta x}_{i+1\to q}\right)\left(\mathcal{A}^{H}_{i\to i+1}\right)(\mathbb{I}-X)\left|\phi^H_i\right\rangle &&+\left(\mathcal{A}^{H*\Theta x}_{i+1\to q}\right)\left(\mathcal{A}^{H}_{i\to i+1}\right)X\left|\phi^H_i\right\rangle\\
&\quad=\left(\mathcal{A}^{H*\Theta x}_{i\to q}\right)(\mathbb{I}-X)\left|\phi^H_i\right\rangle &&+\left(\mathcal{A}^{H*\Theta x}_{i+1\to q}\right)\left(\mathcal{A}^{H}_{i\to i+1}\right)X\left|\phi^H_i\right\rangle\\
&\quad=\left(\mathcal{A}^{H*\Theta x}_{i\to q}\right)\left|\phi^H_i\right\rangle-\left(\mathcal{A}^{H*\Theta x}_{i\to q}\right)X\left|\phi^H_i\right\rangle &&+\left(\mathcal{A}^{H*\Theta x}_{i+1\to q}\right)\left(\mathcal{A}^{H}_{i\to i+1}\right)X\left|\phi^H_i\right\rangle
\end{aligned}$$

Rearranging terms, applying $G^{\Theta}_x=(X\otimes\Pi_{x,\Theta})$ and using the triangle equality, we can thus bound

$$\begin{aligned}
\left\|G^{\Theta}_x\left(\mathcal{A}^{H*\Theta x}_{i\to q}\right)\left|\phi^H_i\right\rangle\right\|_2\leq{}&\left\|G^{\Theta}_x\left(\mathcal{A}^{H*\Theta x}_{i+1\to q}\right)\left|\phi^H_{i+1}\right\rangle\right\|_2\\
&+\left\|G^{\Theta}_x\left(\mathcal{A}^{H*\Theta x}_{i\to q}\right)X\left|\phi^H_i\right\rangle\right\|_2\\
&+\left\|G^{\Theta}_x\left(\mathcal{A}^{H*\Theta x}_{i+1\to q}\right)\left(\mathcal{A}^{H}_{i\to i+1}\right)X\left|\phi^H_i\right\rangle\right\|_2.
\end{aligned}$$

Summing up the respective sides of the inequality over $i=0,\ldots,q-1$, dropping (some of) the zero terms in the summation[2], we get

$$\left\|G^{\Theta}_x\left|\phi^{H*\Theta x}_q\right\rangle\right\|_2\;\leq\;\left\|G^{\Theta}_x\left|\phi^H_q\right\rangle\right\|_2+\sum_{\substack{i\in Q\\ b\in\{0,1\}}}\left\|G^{\Theta}_x\left(\mathcal{A}^{H*\Theta x}_{i+b\to q}\right)\left(\mathcal{A}^{H}_{i\to i+b}\right)X\left|\phi^H_i\right\rangle\right\|_2.$$

By squaring both sides, dividing by $2|Q|+1$ (i.e., the number of terms on the right-hand side), and using Jensen's inequality on the right-hand side, we

[2] At most (if $Q=Q_{\min}$) we drop all terms that are zero for every choice of b, H, Θ, and x.

obtain

$$\frac{\left\|G_x^\Theta\left|\phi_q^{H*\Theta x}\right\rangle\right\|_2^2}{2|Q|+1} \leq \left\|G_x^\Theta\left|\phi_q^H\right\rangle\right\|_2^2 + \sum_{\substack{0\leq i<q\\ b\in\{0,1\}}} \left\|G_x^\Theta\left(\mathcal{A}_{i+b\to q}^{H*\Theta x}\right)\left(\mathcal{A}_{i\to i+b}^{H}\right)X\left|\phi_i^H\right\rangle\right\|_2^2$$

and thus, noting that we can write $\left\|G_x^\Theta\left|\phi_q^H\right\rangle\right\|_2^2$ as

$$\left\|G_x^\Theta\left(\mathcal{A}_{i+b\to q}^{H*\Theta x}\right)\left(\mathcal{A}_{i\to i+b}^{H}\right)X\left|\phi_i^H\right\rangle\right\|_2^2$$

with $i=q$ and $b=0$,

$$\frac{\left\|G_x^\Theta\left|\phi_q^{H*\Theta x}\right\rangle\right\|_2^2}{(2|Q|+1)^2} \;\leq\; \mathbb{E}_{i\in Q,b}\left[\left\|G_x^\Theta\left(\mathcal{A}_{i+b\to q}^{H*\Theta x}\right)\left(\mathcal{A}_{i\to i+b}^{H}\right)X\left|\phi_i^H\right\rangle\right\|_2^2\right].$$

This concludes the proof. □

The corresponding theorem reads as follows:

Theorem A.4 (Measure-and-reprogram with stingy simulator)**.** *Let $\mathcal{X}$ and $\mathcal{Y}$ be finite non-empty sets. There exists a black-box two-stage quantum algorithm $\mathcal{S}$ with the following property. Let $\mathcal{A}$ be an arbitrary oracle quantum algorithm that makes q queries to a uniformly random $H:\mathcal{X}\to\mathcal{Y}$ and that outputs some $x\in\mathcal{X}$ and a (possibly quantum) output z, and let V be a (possibly quantum) predicate. For $i\in\{0,\ldots,q-1\}$, define the two-stage algorithm $\mathcal{S}_i^{\mathcal{A}}$ as follows: In the first stage $\mathcal{S}_i$ measures the ith query of $\mathcal{A}$, and outputs the measurement outcome x'. Then, upon a random $\Theta\in\mathcal{Y}$ as input to the second stage, this very query of $\mathcal{A}$ is answered either using the original H* or *using the reprogrammed oracle $H*\Theta x$, with the choice being made at random, while all the remaining queries of $\mathcal{A}$ are answered using oracle $H*\Theta x$. At the end of its run $\mathcal{S}_i$ then outputs whatever $\mathcal{A}$ outputs (along with i). Now let $Q\subseteq\{0,\ldots,q-1\}$ be such that for all $i\notin Q$ we have $\Pr_{H,\Theta}\left[x'=x\wedge V(x,\Theta,z):(x',x,z)\leftarrow\langle\mathcal{S}_i^{\mathcal{A}},\Theta\rangle\right]=0$. Define $\mathcal{S}(Q)$ to be the algorithm that with probability $\frac{2|Q|}{2|Q|+1}$ picks i uniformly at random from Q and then runs $\mathcal{S}_i$, and with probability $\frac{1}{2|Q|+1}$ chooses $i=q$ and just simulates $\mathcal{A}$ without any measurement or reprogramming, and again outputs whatever $\mathcal{A}$ outputs (along with $x':=x$ and i). We then have*

$$\begin{aligned}&\Pr_{H,\Theta}\left[x'=x\wedge V(x,\Theta,z):(x',x,z,i)\leftarrow\langle\mathcal{S}^{\mathcal{A}}(Q),\Theta\rangle\right]\\ &\geq\frac{1}{(2|Q|+1)^2}\Pr_H\left[V(x,H(x),z):(x,z)\leftarrow\mathcal{A}^H\right].\end{aligned}$$

Furthermore, $\mathcal{S}$ runs in time polynomial in q, $\log|\mathcal{X}|$, and $\log|\mathcal{Y}|$.

A.3 Unsimplified Proof of Lemma 4.32

In this section, we explain how the proof of Lemma 4.32 presented in Section 4.7.2 can be modified to take into account the padding as well as the possibility that the lengths of the message and the public keys may not be multiples of the block length. The parts that change in comparison to Section 4.7.2 are marked in colour.

Lemma 4.32. *Let $\mathcal{D}^H$ and $\mathcal{A}^H$ be $\mathsf{sNR}^{H,\perp}$-adversaries against $\mathsf{sBUFF}[\mathcal{S},\mathsf{MD}]$, making at most $q_\mathcal{D}$ and $q_\mathcal{A} \in \mathbb{Z}_{>0}$ classical queries to H respectively; let $\mathsf{aux} : \mathcal{SK} \times \mathcal{M} \to \mathcal{AUX}$ be any (possibly randomized) function. Then there exists a hider $\bar{\mathcal{D}} : \{\perp\} \to \mathcal{X}^{\leq \bar{B}} \times \mathcal{Z}$ and a seeker $\bar{\mathcal{A}} : \mathcal{Y} \times \mathcal{Z} \to \mathcal{X}^{\leq \bar{B}}$ and $\mathcal{Z} = \mathcal{SK} \times \mathcal{AUX}$, where $\bar{\mathcal{A}}$ makes at most $q_\mathcal{B} := q_\mathcal{A} + q_\mathcal{S}$ and $\bar{\mathcal{D}}$ makes at most $q_\mathcal{D}$ queries to H, and such that*

$$\mathsf{H}^{\infty}_{(x,z)\leftarrow\bar{\mathcal{D}}^H}(x \mid H, z) = \mathsf{H}^{\infty}_{\substack{(\mathsf{sk},\mathsf{pk})\leftarrow\mathsf{KGen}^H \\ m\leftarrow\mathcal{D}^H(\mathsf{sk})}}(m \mid H, \mathsf{sk}, \mathsf{aux}(\mathsf{sk}, m)) \tag{4.22}$$

and $\mathbf{Adv}^{\mathsf{sNR}^{H,\perp}}_{\mathsf{sBUFF}[\mathcal{S},\mathsf{MD}]}(\mathcal{D}, \mathcal{A}, \mathsf{aux})$

$$\leq 2q_\mathcal{B} \cdot r^2 \cdot \mathbf{Adv}^{\mathsf{HnS}^H}_{\mathsf{MD}_\perp}(\bar{\mathcal{D}}, \bar{\mathcal{A}}) + \frac{(q_\mathcal{D}^2 \cdot r + 1) \cdot \bar{B}\bar{L} + q_\mathcal{D} + 2\bar{L}^2}{|\mathcal{Y}|}. \tag{4.23}$$

where $\bar{B}$ and $q_\mathcal{S}$ are as described in Section 4.7.1 and $\bar{L} = q_\mathcal{D} + q_\mathcal{A} + q_\mathcal{S} + 2\bar{B}$. Moreover, if aux is polynomial-time computable and $\mathcal{D}, \mathcal{A}$ are PPT, then so are $\bar{\mathcal{D}}, \bar{\mathcal{A}}$.

Proof. We explain the notation needed for the generalized proof where the lengths of messages and keys do not necessarily line up with the blocks. First, we note that

$$\mathbf{Adv}^{\mathsf{sNR}^{H,\perp}}_{\mathsf{sBUFF}[\mathcal{S},\mathsf{MD}_\perp]}(\mathcal{D}, \mathcal{A}, \mathsf{aux}) \leq \Pr\left[\mathsf{MD}^H_\perp(m\|\mathsf{pk}'\|m) = y' \wedge \mathsf{pk}' \neq \mathsf{pk}\right]$$

with the random variables $\mathsf{pk}, \mathsf{pk}', m$ and y defined by the experiment

$$(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}, \quad m \leftarrow \mathcal{D}^H(\mathsf{sk}),$$
$$(\mathsf{pk}', y') \leftarrow \mathcal{B}^H\big(\mathsf{sk}, \mathsf{MD}^H(m\|\mathsf{pk}\|m), \mathsf{aux}(\mathsf{sk}, m)\big)$$

where $\mathcal{B}(\mathsf{sk}, y, a) := \mathcal{A}^H\big(\mathsf{sk}, (\mathsf{Sign}^H(\mathsf{sk}, y), y), a\big)$. We note that the random choice of H is understood and left implicit. We recall that $\mathcal{D}$ and $\mathcal{B}$ make at most $q_\mathcal{D}$ and $q_\mathcal{B} := q_\mathcal{A} + q_S$ queries to the random oracle respectively. We introduce the following additional random variables, implicitly defined by the above experiment.

Parsing $x = m\|\mathsf{pk}\|m\|\mathsf{pad}(m\|\mathsf{pk}\|m)$ as $x = (x_1, \ldots, x_{|x|_{\mathrm{bl}}})$ and $x' = m\|\mathsf{pk}'\|m\|\mathsf{pad}(m\|\mathsf{pk}'\|m)$ as $x' = (x'_1, \ldots, x'_{|x'|_{\mathrm{bl}}})$ and denoting by $B_{\mathsf{pk}'} =$

$|\,m\|\mathsf{pk}'\|m\|\mathsf{pad}(m\|\mathsf{pk}'\|m)\,|_{\mathrm{bl}}$ and by $B_{\mathsf{pk}} = |\,m\|\mathsf{pk}\|m\|\mathsf{pad}(m\|\mathsf{pk}\|m)\,|_{\mathrm{bl}}$, we let

$$B_1'' = |\,m\,|_{\mathrm{bl}}$$

and

$$B_2'' = \begin{cases} B_{\mathsf{pk}'} - |\,m\|\mathsf{pk}'\,|_{\mathrm{bl}} & \text{if } |\,m\|\mathsf{pk}'\,| = |\,m\|\mathsf{pk}'\,|_{\mathrm{bl}} \cdot r \\ B_{\mathsf{pk}'} - |\,m\|\mathsf{pk}'\,|_{\mathrm{bl}} + 1 & \text{otherwise} \end{cases}$$

i.e., B_1'' is the number of blocks that contain the first occurrence of m, and B_2'' is the number of blocks that contain the second occurrence of m in the sandwich $m\|\mathsf{pk}'\|m$.

For $i \in [B_1'']$ we define m_i to be the ith block of $m\|\mathsf{pk}'$, and for $i \in [B_2'']$ we define m_i' to be the ith block starting from the beginning of the second occurrence of m in the sandwich $m\|\mathsf{pk}'\|m\|\mathsf{pad}(m\|\mathsf{pk}'\|m)$. We define

$$z_1' := \mathsf{MD}_{\perp}^{H}(x_1'\|\ldots\|x'_{B-B_2''}) \text{ and } z'_{i+1} := \mathsf{MD}_{\perp}^{H}(x_1'\|\ldots\|x'_{B-B_2''+i}) = H(m_i', z_i')$$

for $i = 1, \ldots, B_2''$, with $z_{B_2''+1} = \mathsf{MD}^H(m\|\mathsf{pk}'\|m)$ then. The z_i's thus form the "high-order" intermediate digests towards computing $\mathsf{MD}^H(m\|\mathsf{pk}'\|m)$.[3]

Finally, we let $\tau_1, \ldots, \tau_L$ with $L = q_{\mathcal{D}} + B_{\mathsf{pk}} + q_{\mathcal{B}} + B_{\mathsf{pk}'}$ be the list of inputs to all the hash computations performed during the experiment, listed in the performed order; see Fig. 4.18b. Hence, $Q_{\mathcal{D}} = \{\tau_1, \ldots, \tau_{q_{\mathcal{D}}}\}$ consists of the hash queries made by $\mathcal{D}$, we denote by $Q_{\mathsf{MD}(m\|\mathsf{pk}\|m)}$ the B_{pk} queries made during the computation of $\mathsf{MD}(m\|\mathsf{pk}\|m)$ by the challenger, and $Q_{\mathcal{B}} = \{\tau_{q_{\mathcal{D}}+B_{\mathsf{pk}}+1}, \ldots, \tau_{q_{\mathcal{D}}+B_{\mathsf{pk}}+q_{\mathcal{B}}}\}$ of the queries made by $\mathcal{B}$, and the remaining τ_ℓ's are the inputs to the hash computations done towards computing $\mathsf{MD}^H(m\|\mathsf{pk}'\|m)$, in particular $\tau_{q_{\mathcal{D}}+B_{\mathsf{pk}}+q_{\mathcal{B}}+B_{\mathsf{pk}'}-B_2''+1} = (m_1', z_1')$, and $\tau_{q_{\mathcal{D}}+B_{\mathsf{pk}}+q_{\mathcal{B}}+B_{\mathsf{pk}'}-B_2''+2} = (m_2', z_2')$, etc. For any τ_ℓ with $\ell \in [L]$, we write $\mathsf{R}(\tau_\ell)$ for the right component of τ_ℓ, i.e., $\mathsf{R}(\tau_{q_{\mathcal{D}}+B_{\mathsf{pk}}+q_{\mathcal{B}}+B_{\mathsf{pk}'}-B_2''+1}) = z_1'$, etc. and $\mathsf{L}(\tau_\ell)$ is the left component of τ_ℓ, i.e. $\mathsf{L}(\tau_{q_{\mathcal{D}}+B_{\mathsf{pk}}+q_{\mathcal{B}}+B_{\mathsf{pk}'}-B_2''+1}) = m_1'$ etc.

As explained, we are interested in upper-bounding the probability $\Pr[\Sigma]$ of the event

$$\Sigma := \left[\mathsf{MD}^H(m\|\mathsf{pk}'\|m) = y' \wedge \mathsf{pk}' \neq \mathsf{pk}\right].$$

We do this by introducing a sequence of further events, Γ, Λ and Δ, with the property that $\Pr[\Sigma]$ is close to $\Pr[\Sigma \wedge \Gamma \wedge \Lambda \wedge \Delta]$, assuming $\mathbf{Adv}^{\mathsf{HnS}^H_{\mathsf{MD}_\perp}}(\bar{\mathcal{D}}, \bar{\mathcal{A}})$ is small (for suitable choices of $\bar{\mathcal{D}}$ and $\bar{\mathcal{A}}$), and such that we can upper bound the latter probability.

We start off avoiding some atypical behavior of H. Formally, we consider

[3] By "high-order" we mean the digests occurring in the computation of $\mathsf{MD}^H(m\|\mathsf{pk}'\|m)$ from $\mathsf{MD}_{\perp}^{H}(m\|\mathsf{pk}')$.

the good event $\Gamma := \Gamma_1 \wedge \Gamma_2 \wedge \Gamma_3$ with

$$\Gamma_1 := \left[\forall\, \ell, \ell' \in [L] : H(\tau_\ell) = H(\tau_{\ell'}) \;\Rightarrow\; \tau_\ell = \tau_{\ell'}\right]$$
$$\Gamma_2 := \left[\forall\, \ell, \ell' \in [L] : H(\tau_\ell) = \mathsf{R}(\tau_{\ell'}) \;\Rightarrow\; (\exists\, \ell_\circ < \ell' : \tau_\ell = \tau_{\ell_\circ})\right] \qquad \text{and}$$
$$\Gamma_3 := \left[\forall\, \ell \in [q_{\mathcal{D}}] : H(\tau_\ell) \neq IV\right]$$

Informally, Γ_1 states that there are no collisions for the points that H was queried on, Γ_2 states that a hash output does not "bump into" a previous hash input, thus retroactively connecting hash chains, and lastly, Γ_3 states that the initialization vector is never a hash output (this will be helpful later on to identify the start of a hash chain). These events are defined identically as in Section 4.7.2.

Claim 4.33. *It holds that* $\Pr[\Sigma] \leq \Pr[\Sigma \wedge \Gamma] + q_{\mathcal{D}}/|\mathcal{Y}| + 2\bar{L}^2/|\mathcal{Y}|$.

The proof of this claim is identical as that presented in Section 4.7.2.

To bound $\Sigma \wedge \neg\mathsf{GD}$, we need to adapt the subevents $\Delta, \Delta'_k, \Delta^i$ to the new setting.

First, we adapt the event Λ from Section 4.7.2 to the notation above. The goal is to show that if $\mathcal{B}$ has not queried the entire hash chain of the computation of $\mathsf{MD}(m\|\mathsf{pk}'\|m)$,

$$\Lambda := [\exists i\colon (m'_i, z'_i) \notin Q_{\mathcal{B}}]$$

that $\mathcal{B}$ has not made a hash query to one of the high-order intermediate digests z'_i, together with the corresponding message block m'_i.

Claim 4.34. *There exist hide-and-seek adversaries* $\bar{\mathcal{D}}, \bar{\mathcal{A}}$ *such that*

$$\Pr[\Sigma \wedge \Gamma \wedge \neg\Lambda] \leq q_{\mathcal{B}} \cdot 2r^2 \cdot \mathbf{Adv}^{\mathsf{HnS}^H_{\mathsf{MD}_\perp}}(\bar{\mathcal{D}}, \bar{\mathcal{A}})\,, \tag{4.25}$$

where $\bar{\mathcal{A}}$ *makes at most* $q_{\mathcal{B}}$ *and the value* x *chosen by* $\bar{\mathcal{D}}$ *preserves the entropy:*

$$\mathsf{H}_\infty(x \mid z, H) = \mathsf{H}_\infty(m \mid H, \mathsf{sk}, \mathsf{aux}(\mathsf{sk}, m))\,.$$

Moreover, aux *is polynomial-time computable and* $\mathcal{D}, \mathcal{A}$ *are PPT, then so are* $\bar{\mathcal{D}}$ *and* $\bar{\mathcal{A}}$.

Proof. We construct adversaries $\bar{\mathcal{D}}$ and $\bar{\mathcal{A}}$ against hide and seek. First, the adversary $\bar{\mathcal{D}}$ simulates the sNR game to $\mathcal{D}$ by sampling a key pair $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}$ and giving sk to $\mathcal{D}$. It forwards all queries and responses by $\mathcal{D}$ to the random oracle and back. When $\mathcal{D}$ outputs a message m, the adversary $\bar{\mathcal{D}}$ outputs $x = m\|\mathsf{pk}\|m\|\mathsf{pad}(m\|\mathsf{pk}\|m)$ and $z = \mathsf{sk}, \mathsf{aux}(\mathsf{sk}, m)$ as its output.

The adversary $\bar{\mathcal{A}}$ takes as input the hash y and $z = \mathsf{sk}, \mathsf{aux}(\mathsf{sk}, m)$. It parses z into sk and $\mathsf{aux}(\mathsf{sk}, m)$ and runs $\mathcal{B}$ on $\mathsf{sk}, y, \mathsf{aux}(\mathsf{sk}, m)$. It forwards all queries to H and their responses. Here, we observe that if Γ and Σ hold but Λ does not hold, then $\bar{\mathcal{A}}$ is able to restore the entire message m (and

thus win the corresponding hide-and-seek game against $\mathsf{MD}_\perp$ by computing $m\|\mathsf{pk}\|m\|\mathsf{pad}(m\|\mathsf{pk}\|m))$, via inspecting $\mathcal{B}$'s queries to H and its output y' as follows. Indeed, $z'_{B''_2+1} = y'$ (by Σ), and $\mathcal{B}$ has queried $(m'_{B''_2}, z'_{B''_2})$ such that $H(m'_{B''_2}, z'_{B''_2}) = z'_{B''_2+1} = y'$ (by $\neg\Lambda$), and $(m'_{B''_2}, z'_{B''_2})$ is unique with that property (by Γ_1), and so $\bar{\mathcal{A}}$ can find it. By the same argument, $\bar{\mathcal{A}}$ can then find $(m'_{B''_2-1}, z'_{B''_2-1}), (m'_{B''_2-2}, z'_{B''_2-2}), \ldots, (m'_1, z'_1)$ in the queries if it knows B''_2, which is at most $q_{\mathcal{B}}$ and can be guessed with probability $1/q_{\mathcal{B}}$. It remains to guess where the message starts within the block m'_1, which $\bar{\mathcal{A}}$ can guess with probability $\frac{1}{r}$ The adversary $\bar{\mathcal{A}}$ then identifies the padding at the end of the string. If the padding is not easily identifiable, the adversary $\bar{\mathcal{A}}$ makes a guess of the length of the padding which succeeds with probability $\frac{1}{2r}$ as the padding is at most $2r$ long. □

It remains to to bound the success probability of the adversaries in the case that Λ holds.

To define the event Δ analogously to Section 4.7.2 we define m'_i for $i = 0, -1, -2\ldots$ to be the first, second, third ... block before m'_1. Analogously we define z'_i to be the corresponding intermediate digest. We define

$$\Delta := \left[\exists i \geq -\left|\, m\|\mathsf{pk}'\,\right|_{\mathrm{bl}} + B''_1 \colon (m'_i, z'_i) \notin Q_{\mathcal{B}} \cup Q_{\mathcal{D}} \cup Q_{\mathsf{MD}(m\|\mathsf{pk}\|m)}\right]$$

and

$$\Delta'_k := \left[\exists i \in [B''_2] \colon \tau_k = (m'_i, z'_i) \notin Q_{\mathcal{B}} \cup Q_{\mathsf{MD}_\perp(m\|\mathsf{pk}\|m)} \cup \{\tau_{k'}\}_{k' \leq k}\right]$$

for $k \in \{1, \ldots, q_{\mathcal{D}}\}$. It is not too hard to see that $\Delta \vee \Delta'_1 \vee \ldots \vee \Delta'_{q_{\mathcal{D}}} \Leftarrow \Sigma \wedge \Gamma \wedge \Lambda$, and thus by basic manipulations

$$\Pr[\Sigma] \leq \Pr[\Sigma \wedge \Delta] + \sum_k \Pr[\Sigma \wedge \Gamma \wedge \Delta'_k] + \Pr[\Sigma \wedge \Gamma \wedge \neg\Lambda] + \Pr[\neg\Gamma]\,,$$

where we already have bounds for the last two terms.

First, we argue that $\Pr[\Sigma \wedge \Delta]$ is small. For that purpose, we introduce

$$\Delta^i := \left[i \leq B_{\mathsf{pk}'} - B''_1 + 1 \wedge (m'_{B''_2-i+1}, z_{B''_2-i+1}) \notin Q_{\mathcal{B}} \cup Q_{\mathcal{D}} \cup Q_{\mathsf{MD}(m\|\mathsf{pk}\|m)}\right]$$

where $\Delta^{>i} = \bigvee_{j=i+1}^{\bar{B}} \Delta^j$. We note that if $B''_1 \cdot r = |\, m\,|$ then $\Delta^{B_{\mathsf{pk}'} - B''_1 + 1}$ will always be false as the query will happen during the computation of $\mathsf{MD}(m\|\mathsf{pk}\|m)$.

$$\Pr[\Sigma \wedge \Delta] \leq \sum_{i=1}^{\bar{B}} \Pr\left[\Sigma \wedge \Delta^i \wedge \neg\Delta^{>i}\right].$$

The crucial observation now is that conditioned on Δ^i, the hash value of $z'_{B''_2-i+2} = H(m'_{B''_2-i+1}, z_{B''_2-i+1}{}')$ is uniformly random and independent of

y' (and of "everything else"). We formalize this in the claim below, and when plugging in the numbers we obtain:

$$\begin{aligned}\Pr[\Sigma \wedge \Delta] &\leq \sum_{i\in[\bar{B}]} \Pr\left[\Sigma \wedge \Delta^i \wedge \neg\Delta^{>i}\right] \\ &= 0 + \sum_{\substack{i\in[\bar{B}] \text{ s.t.} \\ \Pr[\Delta^i\wedge\neg\Delta^{>i}]>0}} \Pr\left[\Sigma \wedge \Delta^i \wedge \neg\Delta^{>i}\right] \cdot \Pr\left[\Sigma \;\middle|\; \Delta^i \wedge \neg\Delta^{>i}\right] \\ &\leq \sum_{\substack{i\in[\bar{B}] \text{ s.t.} \\ \Pr[\Delta^i\wedge\neg\Delta^{>i}]>0}} \Pr\left[\Delta^i \wedge \neg\Delta^{>i}\right] \cdot \bar{B}\bar{L}/|\mathcal{Y}| \;\leq \bar{B}\bar{L}/|\mathcal{Y}|\;,\end{aligned}$$

where $\bar{L} := q_{\mathcal{D}} + \bar{B} + q_{\mathcal{B}} + \bar{B}$, and the last inequality follows by the disjointness of $\Delta^i \wedge \neg\Delta^{>i}$ across $i \in [\bar{B}]$.

We restate the bound on Δ^i:

Claim 4.35. *It holds for every $i \in [\bar{B}]$ with $\Pr[\Delta^i \wedge \neg\Delta^{>i}] > 0$ that*

$$\Pr\left[\Sigma \;\middle|\; \Delta^i \wedge \neg\Delta^{>i}\right] \leq (i-1)\cdot\frac{\bar{L}}{|\mathcal{Y}|} + \frac{1}{|\mathcal{Y}|} \leq \frac{\bar{B}\bar{L}}{|\mathcal{Y}|}\;,$$

where $\bar{L} := q_{\mathcal{D}} + \bar{B} + q_{\mathcal{B}} + \bar{B}$.

The proof is identical to that in Section 4.7.2.

Towards controlling $\Pr[\Sigma \wedge \mathsf{GD} \wedge \Delta'_k]$, the obstacle is that $\mathcal{D}$'s output m may potentially depend on $H(m'_i, z'_i)$, since it has made a hash query to (m'_i, z'_i) and can thus make its output dependent on the hash (e.g., by choosing $m'_{i+1} := H(m'_i, z'_i)$ then). However, by our "sandwich structure" of the hash computation, this is actually not possible. Indeed, since z'_i is a point in *the second part of* the hash chain, all the points in the first part of the chain, i.e., $z_2 := H(m_1, IV)$, $z_3 := H(m_2, z_2)$ up to $z_{B'_1+1} := H(m_{B'_1}, z_{B''})$, must be determined already, and hence all of m as well, *before* $\mathcal{D}$ learns the hash of (m'_i, z'_i).
We bound the probability in the following claim:

Claim 4.36. $\Pr[\Sigma \wedge \mathsf{GD} \wedge \Delta'_k] \leq q_{\mathcal{D}} \cdot r \cdot \bar{B}\bar{L}/|\mathcal{Y}|$.

Proof. Formally, for a fixed choice of k, we consider the following procedure to (try to) extract m from the first k queries made by $\mathcal{D}$ and the replies to the first $k-1$ of these queries: Start with the kth query τ_k and look for a query within $\{\tau_1, \ldots, \tau_{k-1}\}$ that hashes into $\mathsf{R}(\tau_k)$, and then continuing iteratively with that query, until no further such query exists. By construction, this procedure finds $n \leq k$ and $j_1 < \ldots < j_n = k$ such that

$$H(\tau_{j_1}) = \mathsf{R}(\tau_{j_2})\,,\; H(\tau_{j_2}) = \mathsf{R}(\tau_{j_3}), \ldots, H(\tau_{j_{n-1}}) = \mathsf{R}(\tau_{j_n})\,.$$

The procedure then guesses a value $B^\circ \leftarrow [q_{\mathcal{D}}]$ for the number of message blocks and $\ell \leftarrow [r]$ for the exact end of the message within the last message block $m_{B_\circ}$.

The output of the procedure is then defined to be $\hat{m} := (\mathsf{L}(\tau_{j_1}), \ldots, \mathsf{L}(\tau_{j_{B^\circ}}))[1 \ldots \ell]$ where $[1 \ldots \ell]$ refers to the first ℓ bits of the block. First, we observe that $\Sigma \wedge \Gamma \wedge \Delta'_k$ imply that m is a prefix of $(\mathsf{L}(\tau_{j_1}) \| \ldots \| \mathsf{L}(\tau_{j_n}))$, and thus, as $n \leq q_{\mathcal{D}}$, it holds that $\Pr[m = \hat{m} | \Sigma \wedge \Gamma \wedge \Delta'_k] \geq \frac{1}{r \cdot q_{\mathcal{D}}}$.

Indeed, Δ'_k implies that $\tau_k = (m'_i, z'_i)$ for some i, and so

$$\begin{aligned}\mathsf{R}(\tau_k) = z'_i = \mathsf{MD}^H_\perp(m_1 \| \ldots \| m_B \| \mathsf{pk}' \| m'_1 \| \ldots \| m'_{i-1}) \\ = H(m'_{i-1}, z'_{i-1}) = H(\tau_{q_{\mathcal{D}}+q_{\mathcal{B}}+B+i+1})\,.\end{aligned}$$

Hence, by Γ_2, there exists $j_{n-1} < k$ so that $\tau_{j_{n-1}} = \tau_{q_{\mathcal{D}}+B_{\mathsf{pk}}+q_{\mathcal{B}}+i+1} = (m'_{i-1}, z'_{i-1})$, and thus $H(\tau_{j_{n-1}}) = \mathsf{R}(\tau_k)$. Furthermore,

$$\mathsf{R}(\tau_{j_{n-1}}) = z'_{i-1} = \mathsf{MD}^H_\perp(m_1 \| \ldots \| m_B \| \mathsf{pk}' \| m'_1 \| \ldots \| m'_{i-2})\,,$$

and so by repeating the argument, the procedure extracts, in this reversed order, $m'_i, m'_{i-1}, \ldots, m'_1$, some blocks of pk' and $m_{B''_1}, \ldots, m_1$, until $\tau_{j_1} = (m_1, \mathsf{IV})$, which is when the procedure stops (by Γ_3). This allows us to compute the following probability of extracting the correct message:

$$\begin{aligned}\Pr[\Sigma \wedge \mathsf{GD} \wedge \Delta'_k \wedge \hat{m} = m] &= \Pr[\Sigma \wedge \mathsf{GD} \wedge \Delta'_k] \cdot \Pr[m = \hat{m} | \Sigma \wedge \Gamma \wedge \Delta'_k] \\ &\geq \Pr[\Sigma \wedge \mathsf{GD} \wedge \Delta'_k] \cdot \frac{1}{r \cdot q_{\mathcal{D}}}\end{aligned}$$

Now we make the following "game hop", by replacing the experiment

$$\begin{aligned}(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}, \quad m \leftarrow \mathcal{D}^H(\mathsf{sk}), \\ (\mathsf{pk}', y') \leftarrow \mathcal{B}^H\big(\mathsf{sk}, \mathsf{MD}^H(m \| \mathsf{pk} \| m), \mathsf{aux}(\mathsf{sk}, m)\big),\end{aligned}$$

which defined all the above random variables and probabilities, by

$$\begin{aligned}(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}, \quad \hat{m} \leftarrow \hat{\mathcal{D}}^H(\mathsf{sk}), \\ (\mathsf{pk}', y') \leftarrow \mathcal{B}^H\big(\mathsf{sk}, \mathsf{MD}^H(\hat{m} \| \mathsf{pk} \| \hat{m}), \mathsf{aux}(\mathsf{sk}, \hat{m})\big),\end{aligned}$$

where $\hat{\mathcal{D}}$ runs $\mathcal{D}$, but then stops before sending the kth query to H and instead tries to extract m by means of the above procedure from the prior queries. Correspondingly, we denote its output by $\hat{m}$. We stress that $\hat{\mathcal{D}}$ has now query complexity $q_{\hat{\mathcal{D}}} = k - 1$. The crucial observation is that

$$\Pr[\Sigma \wedge \Gamma \wedge \Delta'_k \wedge \hat{m} = m] \leq \widehat{\Pr}[\Sigma \wedge \Delta]$$

Indeed, in case $\hat{m} = m$ there is no difference in the new experiment, except that now $\hat{\mathcal{D}}$ stops before doing the kth query, and so if $\Gamma \wedge \Delta'_k$ is satisfied in the original experiment then Δ is satisfied in the new one. Thus, we can recycle the bound from above. Using the bound $\widehat{\Pr}[\Sigma \wedge \Delta] \leq \bar{B}\bar{L}/|\mathcal{Y}|$ from Claim 4.35 we obtain using the above that

$$\begin{aligned} & \Pr[\Sigma \wedge \mathsf{GD} \wedge \Delta'_k] \cdot \frac{1}{q_{\mathcal{D}} \cdot r} \leq \Pr[\Sigma \wedge \Gamma \wedge \Delta'_k \wedge \hat{m} = m] \\ \Rightarrow \quad & \Pr[\Sigma \wedge \mathsf{GD} \wedge \Delta'_k] \leq q_{\mathcal{D}} \cdot r \cdot \bar{B}\bar{L}/|\mathcal{Y}| \end{aligned}$$

□

We wrap up the proof by adding up the probabilities

$$\begin{aligned} \Pr[\Sigma] &\leq \Pr[\Sigma \wedge \Delta] + \sum_{k=1}^{q_{\mathcal{D}}} \Pr[\Sigma \wedge \Gamma \wedge \Delta'_k] + \Pr[\Sigma \wedge \Gamma \wedge \neg\Lambda] + \Pr[\neg\Gamma] \\ &\leq \frac{\bar{B}\bar{L}}{|\mathcal{Y}|} + \frac{q_{\mathcal{D}}^2 \cdot r \cdot \bar{B}\bar{L}}{|\mathcal{Y}|} + q_{\mathcal{B}} \cdot 2r^2 \cdot \mathbf{Adv}^{\mathsf{HnS}^H}_{\mathsf{MD}_\perp}(\bar{\mathcal{D}}, \bar{\mathcal{A}}) + \frac{q_{\mathcal{D}} + 2\bar{L}^2}{|\mathcal{Y}|} \\ &= q_{\mathcal{B}} \cdot 2r^2 \cdot \mathbf{Adv}^{\mathsf{HnS}^H}_{\mathsf{MD}_\perp}(\bar{\mathcal{D}}, \bar{\mathcal{A}}) + \frac{(q_{\mathcal{D}}^2 \cdot r + 1) \cdot \bar{B}\bar{L} + q_{\mathcal{D}} + 2\bar{L}^2}{|\mathcal{Y}|} \,. \end{aligned}$$

□

Bibliography

[AAB+22] Carlos Aguilar-Melchor, Nicolas Aragon, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Edoardo Persichetti, Gilles Zémor, Jurjen Bos, Arnaud Dion, Jerome Lacan, Jean-Marc Robert, and Pascal Veron. HQC. Technical report, National Institute of Standards and Technology, 2022. available at `https://csrc.nist.gov/Projects/post-quantum-cryptography/round-4-submissions`.

[ABB+17] Erdem Alkim, Nina Bindel, Johannes A. Buchmann, Özgür Dagdelen, Edward Eaton, Gus Gutoski, Juliane Krämer, and Filip Pawlega. Revisiting TESLA in the quantum random oracle model. In Tanja Lange and Tsuyoshi Takagi, editors, *Post-Quantum Cryptography - 8th International Workshop, PQCrypto 2017*, pages 143–162. Springer, Cham, 2017.

[ABB+20] Erdem Alkim, Paulo S. L. M. Barreto, Nina Bindel, Juliane Krämer, Patrick Longa, and Jefferson E. Ricardini. The lattice-based digital signature scheme qTESLA. In Mauro Conti, Jianying Zhou, Emiliano Casalicchio, and Angelo Spognardi, editors, *ACNS 2020, Part I*, volume 12146 of *LNCS*, pages 441–460. Springer, Cham, October 2020.

[ABK25] Gorjan Alagic, Fahran Bajaj, and Aybars Kocoglu. The best of both KEMs: Securely combining KEMs in post-quantum hybrid schemes. Cryptology ePrint Archive, Paper 2025/1444, 2025.

[ABKM22] Gorjan Alagic, Chen Bai, Jonathan Katz, and Christian Majenz. Post-quantum security of the Even-Mansour cipher. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part III*, volume 13277 of *LNCS*, pages 458–487. Springer, Cham, May / June 2022.

[AHU19] Andris Ambainis, Mike Hamburg, and Dominique Unruh. Quantum security proofs using semi-classical oracles. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019,*

Part II, volume 11693 of *LNCS*, pages 269–295. Springer, Cham, August 2019.

[Ajt96] Miklós Ajtai. Generating hard instances of lattice problems (extended abstract). In *28th ACM STOC*, pages 99–108. ACM Press, May 1996.

[ANSS22] Agence Nationale de la Sécurité des Systémes d'Information (ANSSI). ANSSI views on the post-quantum cryptography transition, 2022. available at `https://cyber.gouv.fr/en/publications/anssi-views-post-quantum-cryptography-transition`.

[ARU14] Andris Ambainis, Ansis Rosmanis, and Dominique Unruh. Quantum attacks on classical proof systems: The hardness of quantum rewinding. In *55th FOCS*, pages 474–483. IEEE Computer Society Press, October 2014.

[Bar01] Boaz Barak. How to go beyond the black-box simulation barrier. In *42nd FOCS*, pages 106–115. IEEE Computer Society Press, October 2001.

[BBD+23] Manuel Barbosa, Gilles Barthe, Christian Doczkal, Jelle Don, Serge Fehr, Benjamin Grégoire, Yu-Hsuan Huang, Andreas Hülsing, Yi Lee, and Xiaodi Wu. Fixing and mechanizing the security proof of Fiat-Shamir with aborts and Dilithium. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part V*, volume 14085 of *LNCS*, pages 358–389. Springer, Cham, August 2023.

[BBD+24] Joppe W. Bos, Olivier Bronchain, Léo Ducas, Serge Fehr, Yu-Hsuan Huang, Thomas Pornin, Eamonn W. Postlethwaite, Thomas Prest, Ludo N. Pulles, and Wessel van Woerden. HAWK. Technical report, National Institute of Standards and Technology, 2024. available at `https://csrc.nist.gov/Projects/pqc-dig-sig/round-2-additional-signatures`.

[BBF+19] Nina Bindel, Jacqueline Brendel, Marc Fischlin, Brian Goncalves, and Douglas Stebila. Hybrid key encapsulation mechanisms and authenticated key exchange. In Jintai Ding and Rainer Steinwandt, editors, *Post-Quantum Cryptography - 10th International Conference, PQCrypto 2019*, pages 206–226. Springer, Cham, 2019.

[BCD+24] Manuel Barbosa, Deirdre Connolly, João Diogo Duarte, Aaron Kaiser, Peter Schwabe, Karolin Varner, and Bas Westerbaan. X-wing. *IACR Communications in Cryptology*, 1(1), 2024.

[BCFW09] Alexandra Boldyreva, David Cash, Marc Fischlin, and Bogdan Warinschi. Foundations of non-malleable hash and one-way functions. In Mitsuru Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 524–541. Springer, Berlin, Heidelberg, December 2009.

[BDF+11] Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. Random oracles in a quantum world. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 41–69. Springer, Berlin, Heidelberg, December 2011.

[BDK+22] Ward Beullens, Samuel Dobson, Shuichi Katsumata, Yi-Fu Lai, and Federico Pintore. Group signatures and more from isogenies and lattices: Generic, simple, and efficient. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 95–126. Springer, 2022.

[BDPA07] G.M. Bertoni, Joan Daemen, Michael Peeters, and Gilles Assche. Sponge functions. *ECRYPT Hash Workshop 2007*, 01 2007.

[Beu21] Ward Beullens. Mayo: Practical post-quantum signatures from oil-and-vinegar maps. In *Selected Areas in Cryptography: 28th International Conference*, 2021.

[BFS11] Paul Baecher, Marc Fischlin, and Dominique Schröder. Expedient non-malleability notions for hash functions. In Aggelos Kiayias, editor, *CT-RSA 2011*, volume 6558 of *LNCS*, pages 268–283. Springer, Berlin, Heidelberg, February 2011.

[BG81] Charles H. Bennett and John Gill. Relative to a random oracle A, $\mathbf{P}^A \neq \mathbf{NP}^A \neq \text{co-}\mathbf{NP}^A$ with probability 1. *SIAM Journal on Computing*, 10(1):96–113, 1981.

[BKP20] Ward Beullens, Shuichi Katsumata, and Federico Pintore. Calamari and Falafl: logarithmic (linkable) ring signatures from isogenies and lattices. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 464–492. Springer, 2020.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In Dorothy E. Denning, Raymond Pyle, Ravi Ganesan, Ravi S. Sandhu, and Victoria Ashby, editors, *ACM CCS 93*, pages 62–73. ACM Press, November 1993.

[BV93] Ethan Bernstein and Umesh Vazirani. Quantum complexity theory. In *Proceedings of the twenty-fifth annual ACM symposium on Theory of computing*, pages 11–20, 1993.

[CAD+24] Alessandro Chiesa, Marcel Dall Agnol, Zijing Di, Ziyi Guan, and Nicholas Spooner. Quantum rewinding for iop-based succinct arguments, 2024.

[CCD+24] Jung Hee Cheon, Hyeongmin Choe, Julien Devevey, Tim Güneysu, Dongyeon Hong, Markus Krausz, Georg Land, Marc Möller, Damien Stehlé, and MinJune Yi. Haetae: Shorter lattice-based Fiat-Shamir signatures. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2024, 2024.

[CD20] André Chailloux and Thomas Debris-Alazard. Tight and optimal reductions for signatures based on average trapdoor preimage sampleable functions and applications to code-based signatures. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020, Part II*, volume 12111 of *LNCS*, pages 453–479. Springer, Cham, May 2020.

[CDF+21] Cas Cremers, Samed Düzlü, Rune Fiedler, Marc Fischlin, and Christian Janson. BUFFing signature schemes beyond unforgeability and the case of post-quantum signatures. In *2021 IEEE Symposium on Security and Privacy*, pages 1696–1714. IEEE Computer Society Press, May 2021.

[CDF+23] Cas Cremers, Samed Düzlü, Rune Fiedler, Marc Fischlin, and Christian Janson. BUFFing signature schemes beyond unforgeability and the case of post-quantum signatures, 2023. An updated version (Version 1.4) of [CDF+21], available at `https://eprint.iacr.org/archive/2020/1525/20231020:082812`.

[CDP23] Sanjit Chatterjee, M. Prem Laxman Das, and Tapas Pandit. Revisiting the security of salted uov signature. In *Progress in Cryptology – INDOCRYPT 2022*, 2023.

[CFHL21] Kai-Min Chung, Serge Fehr, Yu-Hsuan Huang, and Tai-Ning Liao. On the compressed-oracle technique, and post-quantum security of proofs of sequential work. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part II*, volume 12697 of *LNCS*, pages 598–629. Springer, Cham, October 2021.

[CFMR+17] Antoine Casanova, Jean-Charles Faugère, Gilles Macario-Rat, Jacques Patarin, Ludovic Perret, and Jocelyn Ryckeghem. GeMSS: A Great Multivariate Short Signature. Research report, December 2017.

[CFS01] Nicolas T. Courtois, Matthieu Finiasz, and Nicolas Sendrier. How to achieve a McEliece-based digital signature scheme. In Colin Boyd, editor, *Advances in Cryptology — ASIACRYPT 2001*, 2001.

[CGH04] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited. *J. ACM*, 51(4):557–594, July 2004.

[CGLQ20] Kai-Min Chung, Siyao Guo, Qipeng Liu, and Luowen Qian. Tight quantum time-space tradeoffs for function inversion. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 673–684, 2020.

[CHH+21] Kai-Min Chung, Yao-Ching Hsieh, Mi-Ying Huang, Yu-Hsuan Huang, Tanja Lange, and Bo-Yin Yang. Isogeny-based group signatures and accountable ring signatures in QROM. Cryptology ePrint Archive, Paper 2021/1368, 2021.

[CMSZ22] Alessandro Chiesa, Fermi Ma, Nicholas Spooner, and Mark Zhandry. Post-quantum succinct arguments: Breaking the quantum rewinding barrier. In *62nd FOCS*, pages 49–58. IEEE Computer Society Press, February 2022.

[Cou06] Jean-Marc Couveignes. Hard homogeneous spaces. Cryptology ePrint Archive, Report 2006/291, 2006.

[Dal08] Léonard Dallot. Towards a concrete security proof of courtois, finiasz and sendrier signature scheme. In *Research in Cryptology*, 2008.

[Dam90] Ivan Damgård. A design principle for hash functions. In Gilles Brassard, editor, *CRYPTO'89*, volume 435 of *LNCS*, pages 416–427. Springer, New York, August 1990.

[dEK+23] Rafael del Pino, Thomas Espitau, Shuichi Katsumata, Mary Maller, Fabrice Mouhartem, Thomas Prest, Mélissa Rossi, and Markku-Juhani Saarinen. Raccoon. Technical report, National Institute of Standards and Technology, 2023. available at `https://csrc.nist.gov/Projects/pqc-dig-sig/round-1-additional-signatures`.

[Deu85] David Deutsch. Quantum theory, the church–turing principle and the universal quantum computer. *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, 400(1818):97–117, 1985.

[DFG13] Özgür Dagdelen, Marc Fischlin, and Tommaso Gagliardoni. The Fiat-Shamir transformation in a quantum world. In Kazue Sako and Palash Sarkar, editors, *ASIACRYPT 2013, Part II*, volume 8270 of *LNCS*, pages 62–81. Springer, Berlin, Heidelberg, December 2013.

[DFG19a] Luca De Feo and Steven D. Galbraith. Seasign: Compact isogeny signatures from class group actions. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2019*, 2019.

[DFG19b] Luca De Feo and Steven D Galbraith. SeaSign: compact isogeny signatures from class group actions. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 759–789. Springer, 2019.

[DFH22] Jelle Don, Serge Fehr, and Yu-Hsuan Huang. Adaptive versus static multi-oracle algorithms, and quantum security of a split-key PRF. In Eike Kiltz and Vinod Vaikuntanathan, editors, *TCC 2022, Part I*, volume 13747 of *LNCS*, pages 33–51. Springer, Cham, November 2022.

[DFH+24] Jelle Don, Serge Fehr, Yu-Hsuan Huang, Jyun-Jie Liao, and Patrick Struck. Hide-and-seek and the non-resignability of the BUFF transform. In Elette Boyle and Mohammad Mahmoody, editors, *TCC 2024, Part III*, volume 15366 of *LNCS*, pages 347–370. Springer, Cham, December 2024.

[DFHS24] Jelle Don, Serge Fehr, Yu-Hsuan Huang, and Patrick Struck. On the (in)security of the BUFF transform. In Leonid Reyzin and Douglas Stebila, editors, *CRYPTO 2024, Part I*, volume 14920 of *LNCS*, pages 246–275. Springer, Cham, August 2024.

[DFM20] Jelle Don, Serge Fehr, and Christian Majenz. The measure-and-reprogram technique 2.0: Multi-round fiat-shamir and more. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part III*, volume 12172 of *LNCS*, pages 602–631. Springer, Cham, August 2020.

[DFMS19] Jelle Don, Serge Fehr, Christian Majenz, and Christian Schaffner. Security of the Fiat-Shamir transformation in the quantum random-oracle model. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 356–383. Springer, Cham, August 2019.

[DFPS23] Julien Devevey, Pouria Fallahpour, Alain Passelègue, and Damien Stehlé. A detailed analysis of Fiat-Shamir with aborts. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part V*, volume 14085 of *LNCS*, pages 327–357. Springer, Cham, August 2023.

[DG19] Luca De Feo and Steven D. Galbraith. SeaSign: Compact isogeny signatures from class group actions. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part III*, volume 11478 of *LNCS*, pages 759–789. Springer, Cham, May 2019.

[DH76] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[DKL+18] Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. Crystals-dilithium: A lattice-based digital signature scheme. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2018.

[DS23] Marcel Dall'Agnol and Nicholas Spooner. On the necessity of collapsing for post-quantum and quantum commitments. In *18th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2023)*, volume 266 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 2:1–2:23, Dagstuhl, Germany, July 2023. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

[DST19] Thomas Debris-Alazard, Nicolas Sendrier, and Jean-Pierre Tillich. Wave: A new family of trapdoor one-way preimage sampleable functions based on codes. In Steven D. Galbraith and Shiho Moriai, editors, *Advances in Cryptology – ASIACRYPT 2019*, 2019.

[EHH+22] S Ehlen, H Hagemeier, T Hemmert, S Kousidis, M Lochter, S Reinhardt, and T Wunderer. Quantum-safe cryptography–fundamentals, current developments and recommendation. *Federal Office for Information Security (BSI), Godesberger Allee*, pages 185–189, 2022.

[ENST23] Thomas Espitau, Guilhem Niot, Chao Sun, and Mehdi Tibouchi. SQUIRRELS — Square Unstructured Integer Euclidean Lattice Signature. Technical report, National Institute of Standards and Technology, 2023. available at `https://csrc.nist.gov/Projects/pqc-dig-sig/round-1-additional-signatures`.

[FFH25] Pouria Fallahpour, Serge Fehr, and Yu-Hsuan Huang. Tighter quantum security for Fiat-Shamir-with-aborts and hash-and-sign-with-retry signatures. Cryptology ePrint Archive, Paper 2025/985, 2025.

[FH23] Serge Fehr and Yu-Hsuan Huang. On the quantum security of HAWK. In Thomas Johansson and Daniel Smith-Tone, editors, *Post-Quantum Cryptography - 14th International Workshop, PQCrypto 2023*, pages 405–416. Springer, Cham, August 2023.

[FHA23] Serge Fehr, Yu-Hsuan Huang, and Alessandro Amadori. Literature review - (quantum-safe) cryptographic combiners and hybrid security. Technical report, 2023. available at `https://hapkido.tno.nl/deliverables/literature-review-quantum-safe/`.

[FHK+22] Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Prest, Thomas Ricosset, Gregor Seiler, William Whyte, and Zhenfei Zhang. Falcon - whats next? `https://csrc.nist.gov/csrc/media/Presentations/2022/falcon-update/images-media/session-1-prest-falcon-pqc2022.pdf`, 2022.

[FHK25] Serge Fehr, Yu-Hsuan Huang, and Julia Kastner. Sandwich BUFF: Achieving non-resignability using iterative hash functions. In Benny Applebaum and Huijia (Rachel) Lin, editors, *TCC 2025, Part III*, volume 16270 of *LNCS*, pages 235–265. Springer, Cham, December 2025.

[FIKT21] Hiroki Furue, Yasuhiko Ikematsu, Yutaro Kiyomura, and Tsuyoshi Takagi. A new variant of unbalanced oil and vinegar using quotient ring: Qr-uov. In *Advances in Cryptology – ASIACRYPT 2021*, 2021.

[FO99] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In Michael J. Wiener, editor, *CRYPTO'99*, volume 1666 of *LNCS*, pages 537–554. Springer, Berlin, Heidelberg, August 1999.

[FO13] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. *Journal of Cryptology*, 26(1):80–101, January 2013.

[FS87] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *CRYPTO'86*, volume 263 of *LNCS*, pages 186–194. Springer, Berlin, Heidelberg, August 1987.

[GCF+23] Louis Goubin, Benoît Cogliati, Jean-Charles Faugère, Pierre-Alain Fouque, Robin Larrieu, Gilles Macario-Rat, Brice Minaud, and Jacques Patarin. PROV — PRovable unbalanced Oil and Vinegar. Technical report, National Institute of Standards and Technology, 2023. available at `https://csrc.nist.gov/Projects/pqc-dig-sig/round-1-additional-signatures`.

[GHHM21] Alex B. Grilo, Kathrin Hövelmanns, Andreas Hülsing, and Christian Majenz. Tight adaptive reprogramming in the QROM. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part I*, volume 13090 of *LNCS*, pages 637–667. Springer, Cham, December 2021.

[GHP18] Federico Giacon, Felix Heuer, and Bertram Poettering. KEM combiners. In Michel Abdalla and Ricardo Dahab, editors, *PKC 2018, Part I*, volume 10769 of *LNCS*, pages 190–218. Springer, Cham, March 2018.

[GJK24] Phillip Gajland, Jonas Janneck, and Eike Kiltz. A closer look at falcon. *Cryptology ePrint Archive*, 2024.

[GK03] Shafi Goldwasser and Yael Tauman Kalai. On the (in)security of the Fiat-Shamir paradigm. In *44th FOCS*, pages 102–115. IEEE Computer Society Press, October 2003.

[GL89] Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In *21st ACM STOC*, pages 25–32. ACM Press, May 1989.

[GLP12] Tim Güneysu, Vadim Lyubashevsky, and Thomas Pöppelmann. Practical lattice-based cryptography: A signature scheme for embedded systems. In *Workshop on Cryptographic Hardware and Embedded Systems*, 2012.

[GM82] Shafi Goldwasser and Silvio Micali. Probabilistic encryption and how to play mental poker keeping secret all partial information. In *14th ACM STOC*, pages 365–377. ACM Press, May 1982.

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 197–206. ACM Press, May 2008.

[HBD+20] Andreas Hülsing, Daniel J. Bernstein, Christoph Dobraunig, Maria Eichlseder, Scott Fluhrer, Stefan-Lukas Gazdag, Panos Kampanakis, Stefan Kölbl, Tanja Lange, Martin M. Lauridsen, Florian Mendel, Ruben Niederhagen, Christian Rechberger,

Joost Rijneveld, Peter Schwabe, Jean-Philippe Aumasson, Bas Westerbaan, and Ward Beullens. SPHINCS$^+$. Technical report, National Institute of Standards and Technology, 2020. available at `https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions`.

[HBD+22] Andreas Hülsing, Daniel J. Bernstein, Christoph Dobraunig, Maria Eichlseder, Scott Fluhrer, Stefan-Lukas Gazdag, Panos Kampanakis, Stefan Kölbl, Tanja Lange, Martin M. Lauridsen, Florian Mendel, Ruben Niederhagen, Christian Rechberger, Joost Rijneveld, Peter Schwabe, Jean-Philippe Aumasson, Bas Westerbaan, and Ward Beullens. SPHINCS$^+$. Technical report, National Institute of Standards and Technology, 2022. available at `https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022`.

[HC18] Yu-Hsuan Huang and Rong-Jaye Chen. Simulating quantum algorithm by using singular value decomposition. In *Cryptology and Information Security Conference*, 2018.

[HV21] Loïs Huguenin-Dumittan and Serge Vaudenay. FO-like combiners and hybrid post-quantum cryptography. In Mauro Conti, Marc Stevens, and Stephan Krenn, editors, *CANS 21*, volume 13099 of *LNCS*, pages 225–244. Springer, Cham, December 2021.

[HYC20] Yu-Hsuan Huang, Chih-Kai Yang, and Rong-Jaye Chen. Quadrangle inequality improvement for CSIDH strategy. In *Cryptology and Information Security Conference*, 2020.

[IR90] Russell Impagliazzo and Steven Rudich. Limits on the provable consequences of one-way permutations. In Shafi Goldwasser, editor, *CRYPTO'88*, volume 403 of *LNCS*, pages 8–26. Springer, New York, August 1990.

[JCCS19] Dennis Jackson, Cas Cremers, Katriel Cohn-Gordon, and Ralf Sasse. Seems legit: Automated analysis of subtle attacks on protocols that use signatures. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019*, pages 2165–2180. ACM Press, November 2019.

[JST21] Joseph Jaeger, Fang Song, and Stefano Tessaro. Quantum key-length extension. In Kobbi Nissim and Brent Waters, editors, *TCC 2021, Part I*, volume 13042 of *LNCS*, pages 209–239. Springer, Cham, November 2021.

[KBJ+14] Tiffany Hyun-Jin Kim, Cristina Basescu, Limin Jia, Soo Bum Lee, Yih-Chun Hu, and Adrian Perrig. Lightweight source authentication and path validation. In *Proceedings of the 2014 ACM Conference on SIGCOMM*, pages 271–282, 2014.

[KLS18] Eike Kiltz, Vadim Lyubashevsky, and Christian Schaffner. A concrete treatment of Fiat-Shamir signatures in the quantum random-oracle model. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part III*, volume 10822 of *LNCS*, pages 552–586. Springer, Cham, April / May 2018.

[KPG99] Aviad Kipnis, Jacques Patarin, and Louis Goubin. Unbalanced oil and vinegar signature schemes. In *Advances in Cryptology — EUROCRYPT '99*, 1999.

[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.

[KX24] Haruhisa Kosuge and Keita Xagawa. Probabilistic hash-and-sign with retry in the quantum random oracle model. In Qiang Tang and Vanessa Teague, editors, *PKC 2024, Part I*, volume 14601 of *LNCS*, pages 259–288. Springer, Cham, April 2024.

[Lam79] Leslie Lamport. Constructing digital signatures from a one-way function. Technical Report SRI-CSL-98, SRI International Computer Science Laboratory, October 1979.

[LDK+20] Vadim Lyubashevsky, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Peter Schwabe, Gregor Seiler, Damien Stehlé, and Shi Bai. CRYSTALS-DILITHIUM. Technical report, National Institute of Standards and Technology, 2020. available at `https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions`.

[LDK+22] Vadim Lyubashevsky, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Peter Schwabe, Gregor Seiler, Damien Stehlé, and Shi Bai. CRYSTALS-DILITHIUM. Technical report, National Institute of Standards and Technology, 2022. available at `https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022`.

[LMS22] Russell W. F. Lai, Giulio Malavolta, and Nicholas Spooner. Quantum rewinding for many-round protocols. In Eike Kiltz and Vinod Vaikuntanathan, editors, *TCC 2022, Part I*, volume 13747 of *LNCS*, pages 80–109. Springer, Cham, November 2022.

[LNP22] Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Maxime Plancon. Lattice-based zero-knowledge proofs and applications: Shorter, simpler, and more general. *Cryptology ePrint Archive*, 2022.

[LS19] Vadim Lyubashevsky and Peter Schwabe. Round 2 official comment: qTESLA. `https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/round-2/official-comments/qTESLA-round2-official-comment.pdf`, 2019. Accessed: 18-05-2022.

[Lyu09] Vadim Lyubashevsky. Fiat-shamir with aborts: Applications to lattice and factoring-based signatures. In Mitsuru Matsui, editor, *Advances in Cryptology – ASIACRYPT 2009*, 2009.

[Lyu12] Vadim Lyubashevsky. Lattice signatures without trapdoors. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, 2012.

[LZ19] Qipeng Liu and Mark Zhandry. Revisiting post-quantum Fiat-Shamir. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 326–355. Springer, Cham, August 2019.

[LZ23] Dongxi Liu and Raymond K. Zhao. eMLE-Sig 2.0 — Embedded Multilayer Equations with Heavy Layer Randomization. Technical report, National Institute of Standards and Technology, 2023. available at `https://csrc.nist.gov/Projects/pqc-dig-sig/round-1-additional-signatures`.

[McE78] Robert J. McEliece. A public-key cryptosystem based on algebraic coding theory. The deep space network progress report 42-44, Jet Propulsion Laboratory, California Institute of Technology, January/February 1978. `https://ipnpr.jpl.nasa.gov/progress_report2/42-44/44N.PDF`.

[Mer79] Ralph Charles Merkle. *Secrecy, authentication, and public key systems.* PhD thesis, Stanford University, Stanford, CA, USA, 1979. AAI8001972.

[Mer90] Ralph C. Merkle. One way hash functions and DES. In Gilles Brassard, editor, *CRYPTO'89*, volume 435 of *LNCS*, pages 428–446. Springer, New York, August 1990.

[MI88] Tsutomu Matsumoto and Hideki Imai. Public quadratic polynominal-tuples for efficient signature-verification and message-encryption. In C. G. Günther, editor, *EUROCRYPT'88*,

volume 330 of *LNCS*, pages 419–453. Springer, Berlin, Heidelberg, May 1988.

[NIST22] National Institute of Standards and Technology. Call for additional digital signature schemes for the post-quantum cryptography standardization process. `https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/call-for-proposals-dig-sig-sept-2022.pdf`, 2022.

[NSA24] National Security Agency (NSA). The commercial national security algorithm suite 2.0 and quantum computing FAQ, 2024. available at `https://media.defense.gov/2022/Sep/07/2003071836/-1/-1/0/CSI_CNSA_2.0_FAQ_.PDF`.

[Pat96] Jacques Patarin. Hidden fields equations (HFE) and isomorphisms of polynomials (IP): Two new families of asymmetric algorithms. In Ueli Maurer, editor, *Advances in Cryptology — EUROCRYPT '96*, 1996.

[PCF+23] Jacques Patarin, Benoît Cogliati, Jean-Charles Faugère, Pierre-Alain Fouque, Louis Goubin, Robin Larrieu, Gilles Macario-Rat, and Brice Minaud. VOX. Technical report, National Institute of Standards and Technology, 2023. available at `https://csrc.nist.gov/Projects/pqc-dig-sig/round-1-additional-signatures`.

[PFH+20] Thomas Prest, Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Ricosset, Gregor Seiler, William Whyte, and Zhenfei Zhang. FALCON. Technical report, National Institute of Standards and Technology, 2020. available at `https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions`.

[PFH+22] Thomas Prest, Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Ricosset, Gregor Seiler, William Whyte, and Zhenfei Zhang. FALCON. Technical report, National Institute of Standards and Technology, 2022. available at `https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022`.

[PS96] David Pointcheval and Jacques Stern. Security proofs for signature schemes. In Ueli M. Maurer, editor, *EUROCRYPT'96*, volume 1070 of *LNCS*, pages 387–398. Springer, Berlin, Heidelberg, May 1996.

[PS05] Thomas Pornin and Julien P. Stern. Digital signatures do not guarantee exclusive ownership. In John Ioannidis, Angelos Keromytis, and Moti Yung, editors, *ACNS 2005*, volume 3531 of *LNCS*, pages 138–150. Springer, Berlin, Heidelberg, June 2005.

[RSS11] Thomas Ristenpart, Hovav Shacham, and Thomas Shrimpton. Careful with composition: Limitations of the indifferentiability framework. In Kenneth G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 487–506. Springer, Berlin, Heidelberg, May 2011.

[SAB+22] Peter Schwabe, Roberto Avanzi, Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Gregor Seiler, Damien Stehlé, and Jintai Ding. CRYSTALS-KYBER. Technical report, National Institute of Standards and Technology, 2022. available at `https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022`.

[SFG25] Douglas Stebila, Scott Fluhrer, and Shay Gueron. Hybrid key exchange in TLS 1.3. Internet-Draft draft-ietf-tls-hybrid-design-15, Internet Engineering Task Force, September 2025. Work in Progress.

[Sha49] Claude E. Shannon. Communication theory of secrecy systems. *Bell Systems Technical Journal*, 28(4):656–715, 1949.

[Sho94] Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *35th FOCS*, pages 124–134. IEEE Computer Society Press, November 1994.

[Sim94] D.R. Simon. On the power of quantum computation. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 116–123, 1994.

[SSH11] Koichi Sakumoto, Taizo Shirai, and Harunaga Hiwatari. On provable security of UOV and HFE signature schemes against chosen-message attack. In *Post-Quantum Cryptography*, 2011.

[TTB+23] C. Tjhai, M. Tomlinson, G. Bartlett, Scott Fluhrer, Daniel Van Geest, Oscar Garcia-Morchon, and Valery Smyslov. Multiple Key Exchanges in the Internet Key Exchange Protocol Version 2 (IKEv2). RFC 9370, May 2023.

[Unr12] Dominique Unruh. Quantum proofs of knowledge. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 135–152. Springer, Berlin, Heidelberg, April 2012.

[Unr15] Dominique Unruh. Non-interactive zero-knowledge proofs in the quantum random oracle model. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 755–784. Springer, Berlin, Heidelberg, April 2015.

[Unr16] Dominique Unruh. Computationally binding quantum commitments. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 497–527. Springer, Berlin, Heidelberg, May 2016.

[Unr17] Dominique Unruh. Post-quantum security of Fiat-Shamir. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part I*, volume 10624 of *LNCS*, pages 65–95. Springer, Cham, December 2017.

[Wil11] Mark M Wilde. From classical to quantum Shannon theory. *arXiv preprint arXiv:1106.1445*, 2011.

[ZBPB17] Jean Karim Zinzindohoué, Karthikeyan Bhargavan, Jonathan Protzenko, and Benjamin Beurdouche. HACL*: A verified modern cryptographic library. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017*, pages 1789–1806. ACM Press, October / November 2017.

[Zha12] Mark Zhandry. Secure identity-based encryption in the quantum random oracle model. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 758–775. Springer, Berlin, Heidelberg, August 2012.

[Zha19] Mark Zhandry. How to record quantum queries, and applications to quantum indifferentiability. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 239–268. Springer, Cham, August 2019.

Samenvatting

Een cryptografisch algoritme wordt doorgaans in meerdere stappen ontworpen: eerst wordt een eenvoudiger (maar zwakker) algoritme opgesteld, en dit wordt later aangepast tot een sterker en geavanceerder algoritme. In sommige stappen is het gebruikelijk om een generieke transformatie toe te passen die niet is afgestemd op het specifieke algoritme in kwestie. Het doel van het dit proefschrift is om de veiligheid van dergelijke transformaties rigoreuze en bewijsbaar te vaststellen, in de context van post-quantumcryptografie (PQC). De meeste van onze resultaten zijn verkregen in een geïdealiseerde setting die bekend staat als het *random oracle model* (ROM), waarin cryptografische hashfuncties (of sommige delen hiervan) worden gemodelleerd als een willekeurige functie — een random oracle — die zowel door de constructie als door de aanvallers kan worden geraadpleegd. Dit staat in contrast met het *plain model*, wat niet zo'n dergelijke idealisering kent.

In Hoofdstuk 3 bestuderen we de ontwerpprincipes van Fiat-Shamir with aborts (FSwA) en hash-and-sign with retry/aborts (HSwA). Deze transformaties zijn populair bij post-quantum digitale handtekeningalgoritmes, maar de analyse blijkt lastig te zijn. Zoals beschreven in het hoofdstuk, hebben alle eerdere veiligheidsanalyses van FSwA dezelfde subtiele maar cruciale fout, die ook terugkomt in HSwA. Deze fout maakt eerdere veiligheidsbewijzen van alle algoritmes die op FSwA zijn gebaseerd ongeldig, waaronder Dilithium, een van de standaarden die zijn geselecteerd door het Amerikaanse National Institute of Standards and Technology (NIST). We herstellen de veiligheid van FSwA en HSwA handtekeningenalgoritmes door nieuwe, gecorrigeerde veiligheidsbewijzen te leveren in een uniform kader dat beide omvat.

Onze analyses hebben betrekking op zowel klassieke als quantumaanvallen. De technische kern hiervan ligt in een hybride rijtje, waarin het random oracle wordt hergeprogrammeerd. We stellen met name dat, als we de waarden van een random oracle op een bepaalde manier herprogrammeren en vervolgens een deel van die herprogrammering ongedaan maken, een aanvaller in deze reeks wijzigingen aan het random oracle waarschijnlijk geen verschil zal merken. Voor een aanvaller die maximaal q_H quantumvragen aan het random oracle mag stellen en q_S klassieke vragen aan een onderteken orakel, gaat dit in eerste instantie gepaard met een additief veiligheidsverlies van de orde $O(q_H\sqrt{q_S\epsilon})$,

waarbij ϵ een algoritme-afhankelijke parameter is die dicht bij nul ligt. Later verbeteren we dit tot $O(q_S\sqrt{q_H\epsilon})$, via een meer geavanceerd hybride rijtje. Deze verbetering komt kwantitatief overeen met de meest geavanceerde bovengrenzen in het beter begrepen geval van de Fiat-Shamir transformatie (*zonder* aborts). Hoewel het nog onduidelijk is of onze grenzen optimaal zijn, zouden verdere verbeteringen (voor zover zij bestaan) ook doorgevoerd kunnen worden in de beter begrepen setting zonder aborts.

In Hoofdstuk 4 bestuderen we de BUFF transformatie, een transformatie voor handtekeningenalgoritmes die als doel heeft om aanvullende beveiligingseigenschappen te bieden naast de standaard onvervalsbaarheid. We tonen aan dat een van deze eigenschappen, namelijk non-resignability (NR), subtieler is dan eerder werd aangenomen. De oorspronkelijke formele definitie van NR in het plain model, voorgesteld door Cremers *et al.*, blijkt inderdaad onhaalbaar te zijn, en hetzelfde geldt voor de natuurlijke uitbreiding van NR naar het random oracle model. De onhaalbaarheid wordt gepresenteerd in de vorm van een eenvoudige concrete aanval die van toepassing is op alle "natuurlijke" algoritmes (hoewel er een kleine technische opening blijft voor de "onnatuurlijke" algoritmes). In het bijzonder omvat het alle handtekeningenalgoritmes die gebruikmaken van de BUFF transformatie, en daarmee wordt de eerder geclaimde veiligheid van BUFF ongeldig verklaard.

De bovengenoemde aanval vormt echter geen echte bedreiging voor de beoogde toepassingen van NR. In plaats daarvan laat dit zien dat de oorspronkelijke formele definitie van NR ontoereikend is. Om deze negatieve situatie te verhelpen, gaan we terug naar de tekentafel: we stellen meerdere nieuwe formele definities voor en onderzoeken de haalbaarheid van deze definities. Uiteindelijk verkrijgen we zowel positieve als negatieve resultaten. Enerzijds tonen we aan dat de BUFF transformatie inderdaad voldoet aan enkele zinvolle definities van NR, terwijl anderzijds is de haalbaarheid van NR nog steeds sterk afhankelijk van de subtiele details van de formele definitie.

In Hoofdstuk 5 bestuderen we een *KEM samensteller*. Dat is een generieke compiler die meerdere key-encapsulation mechanisms (KEMs) omzet in een gecombineerde KEM, die veilig is als minstens één van de onderliggende KEM's veilig is. Met een dergelijke samensteller zou men een gevestigde pre-quantum KEM, bijvoorbeeld op basis van RSA of Diffie-Hellman, kunnen samenstellen met een nieuw post-quantum algoritme, waardoor partijen soepeler kunnen overstappen op post-quantum cryptografische algoritmes met minder risico's.

Wat we overwegen, is een bijzonder efficiënte constructie die is voorgesteld door Giacon, Heuer en Poettering die gebaseerd is op de veiligheid van een split-key pseudorandom function (skPRF). Omdat de belangrijkste toepassing hiervan in de context van post-quantum cryptografie is, bewijzen we dat deze skPRF veilig blijft tegen quantumaanvallen en dat daardoor ook de bijbehorende KEM samensteller veilig is tegen quantumaanvallen.

Summary

A cryptographic scheme is typically designed in multiple steps: one first constructs a simpler (but weaker) scheme, and then later modifies it to a stronger, and more sophisticated one. In some of the steps, it is common to apply a generic transformation that is not tailored to the specific scheme at hand. The purpose of this thesis is to establish rigorous, provable notions of security for various such transformations that are relevant in the scope of *post-quantum cryptography* (PQC). Most of our results are obtained in an idealized setting known as the *random oracle model* (ROM), where cryptographic hash functions (or some of their components) are modelled as a random function — a *random oracle* — that can be queried by the construction as well as its attackers. This is in contrast to the *plain model* that does not involve such an idealization.

In Chapter 3, we study the Fiat-Shamir with aborts (FSwA), and hash-and-sign with retry/aborts (HSwA) design principles. These transformations are popular among post-quantum signature schemes, but their analyses turn out rather tricky. As described in the chapter, all prior security analyses of FSwA share the same subtle but crucial flaw, which also reappears in HSwA as well. This flaw invalidates prior security proofs of all schemes that rely on FSwA, including Dilithium, one of the standards selected by the US National Institute of Standards and Technology (NIST). We re-establish the security of FSwA and HSwA signature schemes, via providing new, fixed security proofs, in a unified framework that covers both.

Our analyses cover both classical and quantum attacks. The technical core here lies in a hybrid sequence that involves *random oracle reprogramming*. Specifically, we argue that, should we reprogram the values of a random oracle in a certain way, and then later undo some of those reprogramming, then in this sequence of modifications to the random oracle, an attacker is unlikely to notice any difference. For an attacker that is given at most q_H quantum queries to the random oracle, and q_S classical queries to the signing oracle, this initially comes with an additive security loss of order $O(q_H\sqrt{q_S\epsilon})$, where ϵ is a scheme-dependent parameter that is close to zero. We then later improve this to $O(q_S\sqrt{q_H\epsilon})$, via a more sophisticated hybrid sequence. This improvement has quantitatively matched state-of-the-art bounds in the better understood case of the Fiat-Shamir transformation (*without* aborts). Although it remains

open whether our bounds are optimal, any further improvements would also need to carry over to the better-understood setting without aborts.

In Chapter 4, we study the BUFF transformation, which is a transformation for signature schemes that aims to provide additional security properties beyond the standard unforgeability. We show that one of these properties — non-resignability (NR) — is more subtle than previously believed. Indeed, the original formal definition of NR in the plain model, put forward by Cremers *et al.*, turns out unachievable, and the same applies to the natural extension of NR in the random oracle model. The unachievability is presented in the form of a simple concrete attack that applies to all "natural" schemes on the table (though leaving a small technical gap for the "un-natural" ones). In particular, it covers all signature schemes that use the BUFF transformation, and so it invalidates prior claimed security of BUFF.

The aforementioned attack, however, does not really threaten the intended applications of NR. Instead, it demonstrates that the original formal definition of NR is inadequate. To recover from this negative state of affairs, we thus go back to the drawing board: proposing a series of new formal definitions, and investigate achievability of these definitions. In the end, we obtain both positive and negative results. On one hand, we show that the BUFF transformation indeed satisfies some meaningful definitions of NR, while on the other hand, whether or not NR is achieved, or achievable at all, still heavily depends on the subtle details of the formal definition.

In Chapter 5, we study a *KEM combiner*. That is, a generic compiler that transforms multiple key-encapsulation mechanisms (KEMs) into a combined KEM that is secure, as long as at least one of the underlying KEMs is secure. Such a combiner would allow one to combine a well-established pre-quantum KEM, say, based on RSA or Diffie-Hellman, to a new post-quantum scheme, and thereby providing a smoother transition for parties to deploy post-quantum cryptographic schemes with less risk.

What we consider, is a particularly efficient construction proposed by Giacon, Heuer, and Poettering that relies on the security of a split-key pseudorandom function (skPRF). Considering its main application in the context of post-quantum cryptography, we prove that the considered skPRF remains secure against quantum attacks, and consequently the corresponding KEM combiner is also secure against quantum attacks.

Acknowledgement

Being able to do scientific research for the last four years has been a life-long dream come true. Like many others, my PhD journey has consisted of both ebbs and flows, and I owe my sincerest gratitude to the many people who have supported me along the way.

First and foremost, I would like to thank my first advisor, Serge Fehr, who has provided me with guidance on a day-to-day basis. Serge and I shared a common interest in a research style that is theoretical and foundational, which made our collaboration very natural and rewarding. Working with Serge has been a pleasant and unique experience. I often notice his persistence in stripping down complicated arguments and ideas to their simplest forms. At first, this pursuit of simplicity almost felt too restrictive. Over time, however, it has led to better scientific papers and understandings. Even today, I can still vividly recall his voice, urging me to simplify my manuscripts further. Beyond research itself, I also learned from him how to interact with others, kindly and respectfully. My thesis would not have been the same without these lessons, which will surely continue to shape how I do research in the future.

Ronald Cramer, my second advisor, is also the head of Cryptology Group. As a scholar who has witnessed and shaped much of the history of cryptology himself, Ronald often shared with us stories of modern cryptographic methods from the perspective of their historical origins. Located at a hub of cryptology for more than two decades, the group has nurtured so many leading figures in the field. Having been a member for the last four years, I thank him for founding such a wonderful group, and for his down-to-earth friendliness that has bonded the group members more firmly together.

Julia Kastner, my office mate, collaborator, and paranymph, I will never forget your enthusiasm for squirrels, nor all the German language support you provided during the last year of my PhD. Thank you for always having my back — not only in research, when I stumbled over security proofs, but also in the bouldering gym and in everyday life.

Jelle Don, the days that we spent countless hours in front of a whiteboard, trying to tackle a single problem, are truly unforgettable. I also enjoyed hearing your life stories and anecdotes outside of research, whether about an advanture deep in nature, a special choir event, or your fundraising experiences. Thank

you for all the fruitful conversations and for being such an inspiration.

This thesis would literally not exist without its funding source — the HAPKIDO project. My sincere thanks go to everyone who contributed to this project. Additionally, I would also like to thank the reading committee — Prof.dr. Shewta Agrawal, Dr. Christian Majenz, and Prof.dr. Serge Vaudenay — for carefully reading through this thesis, and providing their useful feedbacks.

CWI has been an incredible place to be, largely because of the people at the institute. Special thanks to Simona Etinski, Michael Yonli, Junqiao (Randy) Lin, Shane Gibbons, and the activity committee for putting together engaging activities every once in a while; to Eamonn Postlethwaite, André Schrottenloher, long-term visitors Patrick Struck, Pouria Fallahpour, and Dominik Hartmann for many fruitful research discussions; to Pedro Capitão for co-organizing weekly seminars with me for an entire year; and to Minnie Middelberg, Susanne van Dam, Emil Gorter, and all other supporting staff for their extensive assistance with non-research matters.

The four-year study at CWI is not an easy journey, but even getting to the starting line would not have been possible without Kai-Min Chung. The summer after finishing my undergraduate study, I visited Kai-Min as a summer intern. There, I had my first chance to work with him on the topic of the quantum random oracle model (QROM), an experience without which I might have pursued a PhD elsewhere, or might not have pursued a PhD at all. For a similar reason, I also thank Rong-Jaye Chen, the advisor of my Bachelor's and Master's research, who introduced me into the fascinating world of cryptography.

My fellow ICPC nerds — Jarik Karsten, Wouter Koolen-Wijkstra, Ludo Pulles (who is also my paranymph), and Michelle Sweering — thank you for all the enjoyable moments that we spent on solving a wide range of puzzles, often unrelated to research, whether in front of a computer or around a dinner table. These fast-paced problem-solving sessions, like games of blitz chess, have always refreshed my mind alongside the slower pace of long-term research.

My Taiwanese friends, including but not limited to Yi Lee, Yao-Ting Lin, Miryam Huang, Er-Cheng Tang, Po-Yao (Cosmos) Wang, Alfon Hwu, Po-Kai Yang, David Lin, and Yong-Xuan Wang, thank you for the check-ins (even just occasionally). Knowing everyone is doing well has given me a sense of reassurance, when living on the opposite side of the globe from my hometown.

Finally, my deepest gratitude goes to my family, for their unwavering support, even when it is difficult to understand what I am doing. I was lucky to be raised in a family that values education, without drowning my curiosity in too much coursework. Having my younger brother by my side in childhood, our friendly rivalry has also pushed us to learn and grow together. Today, as lives have taken us to different corners of the world, the bonds we have built will continue to keep our hearts inseparable.

Curriculum Vitae

Yu-Hsuan Huang was born in Kaohsiung, Taiwan, on April 21, 1996. He graduated from Kaohsiung Senior High School in 2014. After spending one more year preparing for the annual Taiwanese Advanced Subject Test, he continued to study at National Chiao-Tung University, and obtained his Bachelor's and Master's degree in Computer Science, in 2019 and 2020 respectively.

During the undergraduate study, Yu-Hsuan was an active member of Programming Challenging Contest Association, where he participated in International Collegiate Programming Contest (ICPC) on behalf of the university. In the spring of 2019, he visited University of Illinois Urbana-Champaign (UIUC) as an exchange student for one semester.

Yu-Hsuan grew his interest in cryptology already since he was an undergraduate student, where he was supervised by Prof. Rong-Jaye Chen for both his Bachelor's and Master's research, the focus of which eventually shifted to elliptic-curve isogenies in the context of post-quantum cryptography. In 2019, he also worked as a summer intern at Academia Sinica in the research group of Dr. Kai-Min Chung, where he first had the chance to work on a research topic about quantum random oracle model (QROM).

In 2021, Yu-Hsuan started working at Centrum Wiskunde & Informatica (CWI) as a PhD student, on the topic of post-quantum cryptography, under the supervision of Prof.dr. Serge Fehr. His PhD research is mainly focused on provable security, with QROM playing a significant role. During his PhD study, Yu-Hsuan also contributed[4] to the proposal of HAWK, a lattice-based signature scheme that is also a candidate in the NIST PQC competition.

[4]More specifically, he contributed to the QROM security proof of HAWK, which is also included in the proposal.

Post-Quantum Security of Cryptographic Transformations in the Random Oracle Model

In cryptography, generic transformations are often used to strengthen simpler but weaker schemes, into more sophisticated and stronger ones.

This thesis aims to establish rigorous, provable notions of security for various such transformations that are relevant in the scope of post-quantum cryptography.

We achieve this via formal mathematical proofs, and in some cases, via proposing new security definitions, when existing ones are inadequate. Most analyses are treated in an idealized setting known as the random oracle model.