University of California
Santa Barbara

Transforming Pseudorandomness and Non-malleability with Minimal Overheads

A dissertation submitted in partial satisfaction
of the requirements for the degree

Doctor of Philosophy
in
Computer Science

by

Pratik Pramod Soni

Committee in charge:

Professor Stefano Tessaro, Co-Chair
Professor Huijia (Rachel) Lin, Co-Chair
Professor Subhash Suri

September 2020

The Dissertation of Pratik Pramod Soni is approved.

Professor Subhash Suri

Professor Huijia (Rachel) Lin, Committee Co-Chair

Professor Stefano Tessaro, Committee Co-Chair

September 2020

Transforming Pseudorandomness and Non-malleability with Minimal Overheads

by

Pratik Pramod Soni

To Mumma and Papa

Acknowledgements

First and foremost, I would like to express my sincere gratitude towards my wonderful advisors Stefano Tessaro and Rachel Lin without whom this dissertation and in large part my development as a researcher would be far from complete. In particular, I am deeply grateful to Stefano for teaching me the rigor of cryptographic research and helping me kick-start my research from day one, and to Rachel for her infinite patience during the early days of our collaboration. You both have been a constant source of support, encouragement, feedback and inspiration, and I will be proud to be even a tenth of an advisor as you both have been to me.

I would like to extend my gratitude to Subhash Suri for agreeing to be on my dissertation committee. Moreover, I am very grateful to Elette Boyle and Alon Rosen for hosting me at IDC Herzliya over the summer of 2019, and also to Ron Rothblum. During my time in Israel I was able to expand my research to different areas of cryptography which was, largely, made possible by your generosity with time and ideas. I truly cherish my time in Israel and hope to be back again soon.

I am privileged to find an amazing labmate, roommate and more importantly a trustworthy friend for life in Ben Terner. From sharing a lab in the early years of our PhD at UCSB to a home when we moved to UW, I have grown as a researcher and as a person interacting with you. Multiple times when I broke down you were always there to listen and provide brotherly advice. From throwing Frisbee on Fridays or baking pizzas over weekends, I can truly say my grad school journey has been better because of your constant presence and encouragement.

I would like to acknowledge my lab mates Binyi Chen who constantly helped me navigate through initial hurdles in research; Priyanka Bose and Aishwarya Thiruvengadam for countless discussions over coffee and walks along the Pacific; Ashrujit Ghoshal, Chris-

tian Matt, Ji Luo, Xihu Zhang, Joseph Jaegar, Wei Dai, and Tianren Liu for nuturing a supportive environment over the years allowing me to learn and be constantly inspired. Thanks to Anna Kornfeld Simpson for helping us find our way at the Allen School at UW. Sincere thanks to Alex Block for being an amazing collaborator since our overlapping visits to IDC Herzliya in Summer 2019. I have found our discussions on Zero-knowledge, Polynomial Commitments and Pokemon Go very insightful. A big thanks to all the staff at the Computer Science departments of both UCSB and UW including Karen Van Gool, Elise Dorough, Nadica Kelly, Genevieve Singer, Maya Wang and many others who tirelessly work behind the scenes to make lives of us grad students much easier.

I had the pleasure of interacting with a number of talented and more importantly warm people during my visit to Israel and conferences including Ran Cohen, Sophia Koepke, Jessica Sorrell, Aarushi Goyal, Arka Rai Chaudhari, Mukul Kulkarni, Pratik Sarkar, Shruti Sekar, Romain Guy, Divya Ravi and Yilei Chen. I would like to extend my thanks to Justin Holmgren and Rafael Pass for being wonderful co-authors.

Every journey has its ups and downs, my PhD journey was no different but I was fortunate to have the constant presence of wonderful people around me who made the highs more meaningful and lows easier to move past: to Neeraj, Aditya and Nhan for opening their home to me; to Anchal for being my amazing dance partner; to Anirudha and Ekta for many evenings playing Exploding Kittens and Codenames; to Juili and Chinmay for an amazing new year's eve among other things; to Vinu for countless conversations over frozen yoghurt and to Rucha for just being there. To Nihit, Ashima, Raj, Akhil and Surabhi for wonderful memories, and to my mentors Siddharth and Milind for their constant guidance and advice

Finally, to my special one Anusha, my extended family, my little brother Chintan, and my parents Bhavana and Pramod for their perennial support, love and belief in me, you complete me!

Curriculum Vitæ
Pratik Pramod Soni

Education

2020	Ph.D. in Computer Science, University of California, Santa Barbara.
2015	M.Sc.(Hons.) in Mathematics, Birla Institute of Technology and Science, Goa, India.
2015	B.E.(Hons.) in Computer Science, Birla Institute of Technology and Science, Goa, India.

Publications[1]

1. Alexandar Block, Justin Holmgren, Alon Rosen, Ron Rothblum and Pratik Soni. **Public-Coin Zero-Knowledge Arguments with (almost) Minimal Time and Space Overheads**, In *Theory of Cryptography Conference - TCC 2020*, to appear.
2. Pratik Soni and Stefano Tessaro. **On the Query Complexity of Constructing PRFs from Non-adaptive PRFs**, In *Security and Cryptography for Networks - SCN 2020*, Sep 2020.
3. Pratik Soni and Stefano Tessaro. **Naor-Reingold Goes Public: The Complexity of Known-key Security.** In *Advances in Cryptology - EUROCRYPT 2018*, May 2018.
4. Huijia Lin, Rafael Pass, and Pratik Soni. **Two-Round and Non-Interactive Concurrent Non-Malleable Commitments from Time-Lock Puzzles.** In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, Oct 2017.
5. Pratik Soni and Stefano Tessaro. **Public-seed Pseudorandom Permutations.** In *Advances in Cryptology - EUROCRYPT 2017*, May 2017.
6. Pratik Soni, Enrico Budianto, and Prateek Saxena. **The SICILIAN Defense: Signature-based Whitelisting of Web JavaScript.** In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (CCS)*, Oct 2015.

Research Employment

- **Research Assistant**, Department of Computer Science, UC Santa Barbara (Jan'16 - Jun'20)
- **Visiting Research Student**, Paul G. Allen School of Computer Science and Engineering, University of Washington, Seattle (Jan'19 - Present)

[1] Authors arranged in alphabetical order (except 6).

- **Gradute Fellow**, FACT Center, IDC Herzliya, Israel (Jun'19 - Sep'19)
- **Junior Research Assistant**, School of Computing, National University of Singapore (NUS), Singapore (Jun'14 - May'15)
- **Research Assistant**, Center for Quantum Technologies, NUS, Singapore (May'13 - Jul'13)
- **Research Intern**, Homi Bhabha Center for Science and Education, Tata Institute for Fundamental Research, Mumbai, India (May'12 - Jul'12)

Awards and Achievements

- **Paper 4 invited to the SIAM Journal of Computing**, special issue for FOCS'17.
- **FOCS Student Travel Award 2017** (600 USD).
- **Merit Scholarship** (equivalent to Dean's list) from BITS Pilani, Goa for academic excellence (2,40,000 INR).
- Selected for Microsoft India Summer Internship, 2015 (7 students selected out of 250 applicants) – (declined).
- Selected for NUS-India Research Initiative Summer Internship, 2013.

Academic Service

- External Conference Reviewer
  - 2020: EUROCRYPT, ITC, CRYPTO, TCC
  - 2019: EUROCRYPT, CRYPTO
  - 2018: CRYPTO, TCC
  - 2017: CRYPTO, ASIACRYPT, TCC
  - 2016: SCN, TCC (2016-B)
- Organizational Czar, Crypto reading group at University of Washington, Seattle (Jan'19-May'19).
- Elected Member, UC Santa Barbara CS Senate (2016-17).
- Co-organizer, Theory-meetups and Crypto reading group at UC Santa Barbara.
- Student In-Charge, International Programmes and Collaboration Division, BITS Pilani, Goa.

Teaching Experience

- Teaching Assistant, UC Santa Barbara.
  - Automata and Formal Languages (Fall'15).
- Teaching Assistant, BITS Pilani, Goa.

– Graphs and Networks (Spring'14), Engineering Mathematics (Fall'13), Computer Programming (Spring'13 and Spring'12).

Outreach Activities

- Member, Abhigyaan, *A literacy drive which aims at 'Education for all'* at Goa, India.
- Guest Speaker at Outreach Program conducted by the cryptography group at UW.

Abstract

Transforming Pseudorandomness and Non-malleability with Minimal Overheads

by

Pratik Pramod Soni

In this thesis, we investigate the *cost* of transforming "weaker" or "less-structured" variants of a cryptographic primitive into a "stronger" or "more structured" variant of the same primitive. We conduct the study via the lens of two fundamental security properties:

Pseudrandomness is critical to almost all of cryptography, and pseudorandom functions (PRFs) and pseudorandom permutations (PRPs) are powerful primitives enabling simple solutions for fundamental problems in secret-key cryptography. Their existence from general assumptions (e.g., one-way functions) is well-studied. But here we investigate new ways of building them with the goal of efficiency and achieving stronger security.

First, we consider building PRFs from non-adaptive PRFs (naPRFs), i.e., PRFs which are secure only against distinguishers issuing all of their queries at once. Known constructions either make $\omega(1)$ calls to an underlying naPRF or incur an undesirable super-polynomial loss in security. We provide the *first* evidence for this state of affairs by showing that a large class of *one-call* constructions cannot be proved to be a secure PRF under a black-box reduction to the (polynomial-time) naPRF security of the underlying function. Second, we revisit the question of transforming PRFs to PRPs which are used to reason about the security of block-ciphers when the underlying key is kept secret. However, in practice block-ciphers are also used in settings where the key is *known* to the adversary. To address this disparity, we introduce the *first, plausible* extensions of pseudorandomness to the known-key setting and provide secure constructions of PRPs

which make *two calls* to an underlying appropriate PRF. This *matches* the complexity of PRF to PRP transformations in the secret-key setting. These results are based on joint works with Stefano Tessaro (EUROCRYPT '17, EUROCRYPT '18 and SCN '20).

Non-malleability captures security of cryptographic protocols against man-in-the-middle (MIM) attacks and non-malleable commitments (NMC) are paragon examples of non-malleable protocols. Resolving the round complexity of NMC, a fundamental measure of cost, has remained a fascinating open question and barriers to achieving two-round (and non-interactive) solutions from polynomial-time assumptions were proved in 2013. We provide the *first* constructions of two-round and non-interactive NMC under sub-exponential time well-studied assumptions, crucially exploiting the synergy between different axes of hardness to circumvent the above impossibility. At heart, our result presents a round-preserving transformation (i.e., incurring no overhead in number of rounds) from NMC on t-bit identities to $\Omega(2^t)$-bits where the length of identities is a measure of a protocol's "non-malleability". Previous such amplifications incurred additive blow-up in the round-complexity. These results are based on joint work with Huijia Lin and Rafael Pass (FOCS '17 and SICOMP '20).

Contents

Chapter 1

Introduction

Edgar Allan Poe, a cryptography enthusiast, in his 1841 essay "A Few Words on Secret Writing" remarked

> *... human ingenuity cannot concoct a cipher which human ingenuity cannot resolve.*

This view perfectly captures the state of cryptography during the 1840's where designing secure cryptosystems was a never-ending game between designers and attackers – designers found new ways to encrypt data only for the attackers to find a flaw, which designers patched until another flaw was uncovered and so on. However, the situation in the year 2020 has evolved significantly in the favor of designers who in some specific sense have found a way to outsmart the hackers. This success story is attributed to the development of framework called *provable security* which defines the intended behaviour of the cryptosystem a.k.a. *correctness*, the adversary class and what it means to for an adversary "break" the cryptosystem, and most importantly advocates that designs of cryptosystems now accompany *proofs* of security.

Cryptographic Reductions. Most proofs of security are in fact what complexity theorists refer to as *reductions*: Assuming that a primitive P exists one demonstrates the existence of another primitive P' by exhibiting a construction that implements P' relying on any implementation of P, possibly only in a black-box manner. A central thread of research in theoretical cryptography, therefore, has been to understand the minimum assumption on P that is sufficient to construct a large class of cryptographic primitives. Over the years our understanding has drastically improved. For example, a number of fascinating cryptographic primitives like *pseudorandom functions*, *signature schemes*, *private-key encryption*, *cryptographic commitments*, *zero-knowledge proofs for NP* and many more can be based on the mild assumption of the existence of *one-way functions*.

It would not be a stretch to say that this study on minimizing assumptions has put much of theoretical and practical cryptography on strong foundations. Despite this success, this line of research suffers from limitations. As the focus is on minimizing assumptions, the resulting constructions turn out to be highly inefficient and hence serve merely as evidence of feasibility. Furthermore, there are known barriers to achieving even moderately efficient constructions of pseudorandom functions, commitments, zero-knowledge proofs etc. from one-way functions. In contrast, the basic-building blocks in practice are block-ciphers and hash-functions which after years of cryptanalysis are believed to offer security stronger than one-way functions.

Finally, as our understanding of a certain cryptographic primitive develops, many variants of the primitive get introduced, e.g., ones secure against weaker/stronger adversaries or exhibiting less/more structure. Understanding the minimal assumptions sufficient for stronger variants immediately resolve the feasibility of weaker variants. However, such results provides little insights into their relative strengths. An in-depth study focussing on transforming different variants of a primitive will provide a more modular

framework to constructing cryptographic primitives.

Our goal: Transformations with minimal overheads In this thesis, we therefore focus on studying reductions – we refer to them as *transformations* – where P is either a weakly-secure variant of P' or variant with "less-structure". This is not to say that the study of minimal assumptions must be abandoned but we believe the studying of transformations, in fact, can lead to more efficient constructions even from minimal assumptions.

More specifically, we are interested in understanding the *minimal cost* of such transformations between variants of *pseudorandom primitives* and *non-malleable primitives*.

Transforming Pseudorandomness. Pseudorandomness is essential to almost all of cryptography, and pseudorandom functions are a fundamental building block which provide direct solutions to secure private-key encryption and message authentication. We here ask:

What is the minimum overhead in terms of number of calls required to transform "weak" or "less-structured" pseudorandom primitives into stronger ones?

We study the above question in two parts: (1) constructing pseudorandom functions from pseudorandom functions secure against weaker (e.g., non-adaptive) adversaries, (2) constructing the more structured *pseudorandom permutations* from pseudorandom functions. We refer the read to Section 1.1 and Section 1.2 for more details.

Transforming Non-malleability. Non-malleability was introduced to capture security of concurrently running cryptographic protocols against man-in-the-middle attacks.

While non-malleability can be formalized for non-interactive primitives like *encryption*, we here consider restrict our study to the case of interactive protocols. We ask:

What is the minimum overhead in terms of number of rounds required to transform "weak" non-malleable protocols to stronger ones?

In this thesis, we study non-malleability for cryptographic commitments and refer the read to Section 1.3 for more details.

1.1 Non-adaptive PRFs to PRFs

We study the problem of building a *pseudorandom function* (PRF) which resists *adaptive attackers* from a *non-adaptive* PRF (naPRF), i.e., a PRF which is only secure against adversaries choosing their inputs non-adaptively at once. This problem has attracted substantial amounts of interest (see e.g. [1, 2, 3, 4, 5, 6, 7, 8]) – indeed, a naPRF may initially be *easier* to devise than a full-fledged PRF.[1] However, to date, the complexity of the best possible transformation remains unknown,[2] and natural approaches such as sequential and parallel composition have been proved to fail. Motivated by this we ask the following question in this thesis:

Question 1: *Do there exist highly-efficient black-box naPRF-to-PRF transformations?*

In Chapter 2 we answer the above question in the negative: We rule out a large class of *one-call* constructions with respect to *hardness-preserving* black-box security proofs. Here, hardness preserving means that the transformation preserves security against PPT

[1] See [9] for a concrete example.

[2] Note that as naPRFs imply one-way functions and PRGs, and thus in turn also PRFs, such transformations are always possible.

adversaries. As we argue below, understanding one-call constructions is a challenging first step towards understanding the overall problem. This in particular shows that previous work by Berman and Haitner (BH) [7], giving a one-call construction relying on complexity leveraging in the security proof, is best possible. Also, it is consistent with the fact that all hardness-preserving transformations make $\omega(1)$ calls.

We also extend our result to a class of *multi*-call parallel constructions, and prove that these, too, do not transform a naPRF into a PRF. This result can be seen as a generalization of Myers [1] black-box separation for the parallel composition. We elaborate on this below, but first give some more context.

From non-adaptive to adaptive security. The problem of building PRFs from naPRFs is well-understood in the information-theoretic case, i.e., attackers are only bounded in query complexity (but not in their running time). Here, simple constructions are sufficient (e.g., sequential and parallel composition). This was first proved by Vaudenay [10], and also follows as the application of general composition theorems [2, 5].

However, negative results have shown that such simple approaches fail in the computational regime, both with respect to black-box reductions [1], as well as without any proof restriction, but assuming DDH holds [4]. Later, it was also shown [3] that public-key assumptions are necessary for counter-examples. This already suggests that the computational setting is harder, but note that these results only cover specific constructions. Here, we aim for more general impossibility, and this presents several additional challenges – in fact, already for *one-call* constructions, which are our main focus.

From naPRFs to PRFs: Prior works. The most efficient known transformations can be cast in terms of the same two steps: (1) we use a naPRF H (say, in the following, with n-bit seeds, inputs, and outputs) to build a PRF with a "small" domain, i.e., the

strings of length $\ell = \omega(\log n)$; (2) the domain of the resulting PRF is extended without extra calls by using (almost) universal hashing – this is often referred to as "Levin's trick", and is also reminiscent of universal-hashing based MACs [11, 12].[3]

There are two ways to accomplish step (1):

- CASCADING. A first, folkore approach (which is hardness-preserving) is via a variant of the cascade construction [13]. For a fixed polynomial $p = p(n)$, we first fix distinct n-bit strings $z_1, \ldots, z_p$. Now, let $\ell = d \log p$ for some $d = \omega(1)$, and think of an ℓ-bit input x as a vector $x = (x_1, \ldots, x_d) \in [p]^d$. Then, the output with seed k is y_d, where

$$y_0 = k \ , \quad y_i = \mathsf{H}(y_{i-1}, z_{x_i}) \text{ for all } i = 1, \ldots, d.$$

  This is a secure PRF as long as H is a secure PRF on the domain $\{z_1, \ldots, z_p\}$, and since p is a fixed polynomial, it is enough that H is a naPRF for p-query distinguishers that query all of $z_1, \ldots z_p$ at once.

- THE BH APPROACH. The core idea of the BH construction can be cast as the fact that every sufficiently secure naPRF secure against $(t = O(2^\ell))$-time distinguishers, where $\ell = \omega(\log n)$, is *already an adaptively secure PRF* for polynomial-time distinguishers, as long as we only query a (fixed) subset of the domain of size 2^ℓ. (This follows by a straightforward reduction which queries all of these points beforehand.) I.e., we can then obtain an adaptively secure PRF with ℓ-bit domain as $\mathsf{F}(k, x) = \mathsf{H}(k, x \| 0^{n-\ell})$. Note that it is necessary to fix a super-polynomial t a-priori, since the construction depends on t and we want security for all polynomial-time distinguishers.

[3] We stress that this approach inherently relies on an asymptotic view targeting PPT security, which we take in this paper – if we are interested in concrete security, the best we can hope for is $2^{\ell/2}$ security, and thus we may need even more calls to the underlying naPRF.

1.1.1 Our result

This still leaves open the question whether the BH construction is secure only assuming H to be secure against PPT adversaries. We show in Chapter 2 that

Theorem (Informal) *There exists no (fully) black-box reduction to show PRF security of* F *assuming* H *is a naPRF where* F *is of the form:*

$$\mathsf{F}((s,k),x) = y \ , \quad \text{where} \quad w = \mathsf{C}(s,x) \ , \quad z = \mathsf{H}(k,w) \ , \quad y = \mathsf{G}(s,x,z) \ ,$$

where C *is an arbitrary (seeded) pre-processing function from* n *bits to* n *bits.*[4]

This class in fact includes *all* possible constructions which do not manipulate the seed k of the underlying naPRF and in particular, includes the BH construction. As our main result, we show an oracle with respect to which (1) naPRFs exist, but (2) the above construction is insecure, provided C satisfies a mild combinatorial property *and* the output length of G is lower bounded by a small constant. This implies the impossibility of providing a fully-BB reduction of security for such a construction to the (polynomial-time) security of H as a naPRF.

The combinatorial condition is that for some constant $c = O(1)$, the function C satisfies a notion we refer to as c-universal, which means that for any choice of c distinct n-bit strings $x_1, \ldots, x_c$, and a random seed s, the values $\mathsf{C}(s,x_1), \ldots, \mathsf{C}(s,x_c)$ are unlikely to be all equal. While this condition appears inherent using traditional security proofs (which often requires the input to H to be "fresh"), it is not clear how to prove it is necessary for any post-processing function G.

However, we can drop this condition for some special cases. For example, when G simply outputs (part of) z, then we see that if C is not 2-universal, then we can break PRF security of the construction directly, provided the output length is $\omega(\log n)$. There are

[4]The choice of an n-bit input for C is arbitrary here, because for any domain length $\ell = \omega(\log n)$, we can modify C to make the domain n bits, either by appending $0^{\ell-n}$ to the input if $\ell > n$, or by using universal hashing as described above if $\ell < n$.

cases where however our result does not completely rule out construction – it is possible that C is not 2-universal and we can achieve security nonetheless when the naPRF has a single-bit output.

Multi-call constructions. We also extend the techniques to prove our main result on one-call constructions to a restricted class of *parallel* κ-call constructions that output, on input x,

$$y = \mathsf{G}(s_{\kappa+1}, x) \oplus \bigoplus_{i\in[\kappa]} \mathsf{H}(k_i, \mathsf{C}(s_i, x))$$

where C is a c-universal pre-processing function, whereas G can be arbitrary. This family includes e.g. the Cuckoo-Hashing based construction from [8]. This result can be seen as a generalization of the work of Myers [1], which studies the special case without any pre-processing.

Impossibility for general reductions. One may wonder whether the results claimed in this paper can be extended to rule out general reductions, e.g., from DDH, following the paradigm of Pietrzak [4]. This is unlikely to be true. In particular, Pietrzak [4] separation holds even if DDH is exponentially hard – however, under such strong hardness, one can simply use the BH construction.

A perspective. We believe the question of assessing how efficiently we can obtain a PRF from a non-adaptive object like a naPRF to be among the most fascinating ones in classical cryptography (although perhaps somewhat overlooked). Constructions are easy in retrospect, and, like in many other instances, seemingly very hard to improve, yet proving that they are indeed best possible appears to be out of reach.

This in particular justifies the perhaps limited-looking scope of our results – we hope to provide evidence that ruling out even a subclass of one-call constructions is a chal-

lenging problem *and* substantial progress. It would be of course desirable to provide impossibility for *all* constant-query constructions – a statement we conjecture to be true. However, we believe this to remain a challenging open question. Our work can be seen as one among a large body of results that provide lower bounds on the efficiency of black-box constructions, e.g. [14, 15, 16, 17, 18, 19, 20].

1.2 Public-seed PRFs to Public-seed PRPs

Next, we switch our focus to *pseudorandom permutations* (PRPs) which are a family, indexed by a key/seed, of efficiently computable/invertible permutations E on n-bit strings for which no efficient distinguisher can distinguish between access to $(\mathsf{E}_s(\cdot), \mathsf{E}_s^{-1}(\cdot))$ and $(\rho(\cdot), \rho^{-1}(\cdot)$ (for a random permutation ρ), for a randomly chosen key s. PRPs were introduced to provide provable security guarantees for block-ciphers when used within modes of operation under a *secret* key. This has motivated a large body of theoretical works on building PRPs from weaker or less structured components, e.g., through the Feistel construction and its variants [21, 22, 23].

Block ciphers are however also frequently used in settings where the key is fixed, or at least *known*. We refer to this as the *known-key setting*. For instance, it is common to rely on *permutations*[5] or (equivalently) *fixed-key* ciphers to build hash functions [24, 25], authenticated encryption [26], PRNGs [27, 28], and even more involved objects, such as garbling schemes [29].

As there is no secret key to rely upon, it is less clear what kind of security properties block ciphers should satisfy in this setting. Hence, security proofs typically assume the cipher to behave like an ideal random permutation on each key. A number of design paradigms for block ciphers (cf. e.g. [30, 31, 32, 33, 34, 35] to mention a few results)

[5] Permutations, as in the sponge construction, correspond to the extreme case where there is only one possible key to choose from.

are therefore analyzed in terms of *indifferentiability* [36], an ideal-model property which implies that for single-stage security games, the cipher inherits all properties of an ideal cipher. Still, the resulting proofs are notoriously involved, and the constructions more complex than seemingly necessary for the applications in which they are used. This is in sharp contrast with hash functions, where indifferentiability has helped shaping real-world designs. To remedy this situation, we ask:

What plausible computational assumptions can we make on block-ciphers to guarantee security in the known-key setting?

In Chapter 3, we propose the first computational assumptions to capture the security of block-ciphers in the known-key setting: More specifically, we introduce a new framework – which we call *public-seed pseudorandom permutations*, or psPRPs, for short.

Public-seed PRPs. At a high-level, public-seed PRPs are an extension of the secret-key notion of PRPs to the known-key setting but some care is required. A naive extension to the known-key setting would be to expect $\mathsf{E}_s(\cdot)$ and $\mathsf{E}_s^{-1}(\cdot)$ to be indistinguishable from ρ and ρ^{-1} (for a random permutation ρ), *even* if the seed s is known to the distinguisher. This is obviously impossible, yet an approach to get around this borrowed from the UCE framework [37] is to *split* the distinguisher into two stages. A first stage, called the *source* S, gets access to either $(\mathsf{E}_s, \mathsf{E}_s^{-1})$ or (ρ, ρ^{-1}), but does *not* know s, and then passes on some leakage L to a second-stage PPT D, the *distinguisher*, which learns s, but has no access to the oracle any more. If E is indeed secure, D will not be able to guess which one of the two oracles S had access to. This is very similar to the security definition of a UCE H which as we will see in Chapter 3 are equivalent to public-seed PRFs: the only difference is that there the source accesses either H_s or a random *function* f.

Clearly, nothing is gained if L is unrestricted, and thus restrictions on S are necessary. Two classes of sources were in particular considered in [38], *unpredictable* and *reset-secure*

sources, inspired by analogous notions for UCEs. The former demands that when the source S accesses ρ and ρ^{-1}, an (unbounded)[6] predictor P given the leakage L cannot then guess any of S's queries (and their inverses). In contrast, the latter notion demands that a computationally unbounded distinguisher R, given L cannot tell apart whether it is given oracle access to the *same* permutation ρ, or an independent one, within a polynomial number of queries. Being a psPRP for all unpredictable sources is a potentially weaker assumption than being a psPRP for reset-secure sources, since every unpredictable source is reset-secure, but not vice versa.

Applications. PsPRPs for such restricted source classes are a versatile notion. For example, a psPRP for all reset-secure sources can be used to instantiate the permutation within permutation-based hash functions admitting indifferentiability-based security proofs, such as the sponge construction [24] (which underlies SHA-3), turning them into a UCE-secure hash function sufficient for a number of applications, studied in multiple works [37, 40, 41, 42, 43]. Also, [38] shows that psPRPs for unpredictable sources are sufficient to instantiate garbling schemes obtained from fixed-key blocks ciphers [29].

Constructing psPRPs. Obviously, a central question is whether the assumption of being a psPRP is, by itself, attainable. Given the success story of building PRPs from PRFs where constructions making as low as two-calls to a PRF exist, it is natural to ask:

Question 2: *Can we build public-seed PRPs from public-seed PRFs, if so how many calls to public-seed PRFs are required?*

[6] Computational versions of these notions can be defined, but the resulting notions can easily be shown impossible under the assumption that IO exists [39], and are ignored in this paper.

1.2.1 Our Result

We answer the above question in two parts and in the process introduce new techniques and intermediate security notions which enhances our understanding of block-cipher constructions.

Five-call construction. First, in Chapter 3, we first show that a construction that makes only five-calls. More specifically,

Theorem (Informal) *The five-round Feistel construction, with round functions instantiated with a UCE* H *for reset-sources, is a psPRP for* X*-sources, where* $X \in \{$*reset-secure, unpredictable*$\}$*.*

In fact, this result follows from a general theorem which in fact establishes connections between UCEs and psPRPs via a weaker notion of indifferentiability – CP-sequential indifferentiability.

CP-sequential indifferentiability. The technique behind our general theorem relating UCEs and psPRPs is inspired by Bellare, Hoang, and Keelveedhi's work [44] on UCE domain extension. They show that every construction that transforms a fixed-input length random oracle into a variable-input length one in the sense of indifferentiability [36, 45] is a good domain extender for UCEs.

We extend their result along three axes. First off, we show that it applies to arbitrary pairs of ideal primitives – e.g., a fixed-input-length or variable-input length random function or a random permutation. For example, a construction using a permutation which is indifferentiable from a random oracle transforms permutations which are psPRPs for reset-sources into functions which are UCEs for reset-sources.

Second, we show that a weaker version of indifferentiability, which we call *CP-sequential indifferentiability*, suffices. Recall that indifferentiability of a construction

M transforming an ideal primitive $\mathbf{I}$ into an ideal primitive $\mathbf{J}$ means that there exists a simulator Sim such that $(\mathsf{M}^{\mathbf{I}}, \mathbf{I})$ and $(\mathbf{J}, \mathsf{Sim}^{\mathbf{J}})$ are indistinguishable. CP-sequential indifferentiability only demands this for distinguishers that make all of their *construction* queries to $\mathsf{M}^{\mathbf{I}}$ / $\mathbf{J}$ *before* they proceed to *primitive* queries to $\mathbf{I}$ / $\mathsf{Sim}^{\mathbf{J}}$. As we will see, this significantly enlarges the set of constructions this result applies to. For example, truncating the permutation output to $r < n$ bits does not achieve indifferentiability, because a simulator on an inverse query Y needs to fix $\pi^{-1}(Y)$ to some X such that, for the given random function $\rho : \{0,1\}^n \to \{0,1\}^r$, $\rho(X)$ is consistent with Y on the first r bits, which is infeasible. Yet, the same construction *is* CP-sequentially indifferentiable, intuitively because there is no way for a distinguisher to catch an inconsistent random X, as this would require an extra query to ρ. CP-sequential indifferentiability is dual to the sequential indifferentiability notion of Mandal, Patarin, and Seurin [46], which considers the opposite order of construction and primitive queries. In fact, the two notions are incomparable, as we explain below.

Finally, we will also show that under suitable restrictions on the construction M, the result extends from reset-secure sources to unpredictable ones. This will allow to lift our result for truncation to unpredictable sources.

Given our general theorem and the indifferentiability result for Feistel constructions [31, 47, 34, 35] imply already that the 8-round Feistel construction transforms a function which is $\mathsf{UCE}[\mathcal{S}^{\star\mathsf{rs}}]$-secure into a $\mathsf{psPRP}[\mathcal{S}^{\star\mathsf{rs}}]$-secure permutation.

It is important however to assess whether simpler constructions achieve this result. *Here, we show that the five-round Feistel construction suffices.* Our proof heavily exploits our connection to CP-indifferentiability. Indeed, the six-round lower bound of [47] does not apply for CP-indifferentiabiliy, as it requires the ability to ask construction queries *after* primitive queries, and we show that CP-indifferentiability is achieved at five rounds already. Our result is not merely a simplification of earlier ones, and our simulation

strategy is novel. In particular, while we still follow the chain-completion approach of previous works, due to the low number of rounds, we need to introduce new techniques to bound the complexity of the simulator. To our rescue will come the fact that no construction queries can be made after primitive queries, and hence only a limited number of chain types will need to be completed. The detailed construction and its proof is deferred to Chapter 3.

Two-call construction. The five-call construction however leaves two questions open: **(1)** Whether the number of rounds/calls to the underlying UCE function can be reduced, and **(2)** whether the same holds for unpredictable sources, too. Here, making progress on these questions the approach via CP-indifferentiability does not seem to help.

In Chapter 4, we solve both questions, and even more in fact, showing that the Naor-Reingold (NR) construction [22] solves *both* (1) and (2). In particular, let H be a family of functions from $n+1$ bits to n, and let P be a family of permutations on $2n$ bit strings. Then, the NR construction on seed $\vec{s} = (s, s^{\mathsf{in}}, s^{\mathsf{out}})$ and input $u \in \{0,1\}^{2n}$, outputs $v \in \{0,1\}^{2n}$, where

$$x_0 \,\|\, x_1 \leftarrow \mathsf{P}_{s^{\mathsf{in}}}(u)\ , \quad x_2 \leftarrow \mathsf{H}_s(0 \,\|\, x_1) \oplus x_0\ ,$$
$$x_3 \leftarrow \mathsf{H}_s(1 \,\|\, x_2) \oplus x_1\ , \quad v \leftarrow \mathsf{P}^{-1}_{s^{\mathsf{out}}}(x_3 \,\|\, x_2)\ .$$

The key point here is that P only needs to satisfy a weak non-cryptographic property, namely that for a random s and for any distinct $u \neq u'$, the right halves of $\mathsf{P}_s(u)$ and $\mathsf{P}_s(u')$ only collide with negligible probability. Therefore, only two calls to a "cryptographically hard" round function H are made. Naor and Reingold [22] showed that NR is a (strong) PRP whenever H is a pseudorandom function. Here, we show the following public-seed counterparts:

Theorems (Informal) *The NR construction, with round functions instantiated with a*

UCE H *for* X*-sources, is a psPRP for* X*-sources, where* $X \in \{$*reset-secure, unpredictable*$\}$.

Building psPRPs from simpler assumptions. Recently, some works give constructions of UCEs. Brzuska and Mittelbach [48] gave constructions from auxiliary-input point obfuscation (AIPO) and iO. In a recent paper, under the exponential DDH assumption, Zhandry [49] built a primitive (called an AI-PRG) which is equivalent to a UCE for a subset of $\mathcal{S}^{\mathsf{cup}}$ which is sufficient for instantiating point obfuscators. (The observation is not made explicit in [49], but the definitions are equivalent.) None of these results is sufficiently strong to instantiate our construction of psPRPs.

We remark here that such UCEs are of course strong, and the question of basing psPRPs on simpler assumptions is wide open. Still, we believe our results from UCEs to be very important: First off, they show relations among notions, and getting a UCE (without any injectivity structure) is possibly simpler in practice than in theory (i.e., using the compression function of SHA-256). Second, even if we instantiate H from a random oracle (which gives a good UCE [37]), the result *is* useful, as this would give us a simple instantiation of a (seeded) permutation in applications which are not even known to follow from full-fledged indifferentiability, as discussed by Mittelbach [50].

1.3 Upgrading Non-malleability for Commitments

Commitment schemes are one of the most fundamental cryptographic building blocks. Often described as the "digital" analogue of sealed envelopes, commitment schemes enable a *sender* to commit itself to a value while keeping it secret from the *receiver*. This property is called *hiding*. Furthermore, the commitment is *binding*, and thus in a later stage when the commitment is opened, it is guaranteed that the "opening" can yield only

a single value determined in the committing stage.

For many applications, however, the most basic security guarantees of commitments are not sufficient. For instance, the basic definition of commitments does not rule out an attack where an adversary, upon seeing a commitment to a specific value v, is able to commit to a related value (say, $v-1$), even though it does not know the actual value of v. To address this concern, Dolev, Dwork and Naor (DDN) introduced the concept of *non-malleable commitments* [51]. Loosely speaking, a commitment scheme is said to be non-malleable if it is infeasible for an adversary to "maul" a commitment to a value v into a commitment to a related value $\tilde{v}$. The notion of a *concurrent non-malleable commitment* [51, 52] further requires non-malleability to hold even if the adversary receives many commitments and can itself produce many commitments.

The first non-malleable commitment protocol was constructed in the original work of [51] in 1991, based on the minimal assumption of one-way functions. The first concurrently secure construction was provided by Pass and Rosen in 2005 [52]. Since then, a central question in the study of non-malleability has been to determine the exact number of communication rounds needed for achieving (concurrent) non-malleable commitments. Significant progress has been made over the years [53, 52, 54, 55, 56, 57, 58, 59, 60, 61, 62]. The current state-of-the-art is that 4-round concurrent non-malleable commitments can be constructed based on one-way functions [63], 3-round concurrent non-malleable commitments can be constructed from subexponentially-secure one-way permutations [64, 65], and very recently can be based only on the polynomial hardness of either DDH or Quadratic-residousity or N^{th}-residuosity and ZAPs [66].

On the Existence of Two-Round or Non-Interactive Non-malleable Commitments. The situation changes drastically when it comes to two-round or non-interactive (i.e., one-message) protocols: Pandey, Pass and Vaikuntanathan [57] provided a con-

struction of a non-interactive non-malleable commitment based on a new *non-falsifiable* hardness assumption, namely, the existence of an *adaptively-secure injective one-way function*—roughly speaking, a one-way function f that is hard to invert on a random point $y = f(x)$ even if you get access to an inversion oracle that inverts it on every *other* point $y' \neq y$. This assumption is not falsifiable since the inversion oracle cannot be implemented in "real-life"[7]; additionally, note that the assumption also has a strong non-malleability flavor—in particular, the assumption would clearly be false if one could "maul" $y = f(x)$ to e.g., $y' = f(x+1)$. As such, a question that remains open is whether we can obtain two-round "non-malleability" from "pure scratch" (i.e., from "hardness" alone). Indeed, a recent work by Pass [67] showed that there are some inherent limitations to reducing 2-round non-malleability to falsifiable assumptions. More precisely, Pass shows that if there exists a 2-round non-malleable commitment that can be proven secure using a polyomial-time (or even super-polynomial, but security preserving[8]) black-box reduction R to a falsifiable assumption, then the reduction R can itself be used to break the assumption. In particular, this rules out basing 2-round non-malleability (using black-box reduction) on falsifiable hardness assumptions against polynomial time adversaries.

Towards overcoming this barrier, a recent work by Goyal, Khurana and Sahai [68] presents a two-message protocol in a stronger "synchronous model" of communication (and achieving only a weaker notion of non-malleability "w.r.t. opening"). In this work, we focus on the standard communication model (and the standard notion of non-malleability) and explore whether super-polynomial-time hardness assumptions (and us-

[7]More precisely, an assumption is falsifiable if it can be modeled as a game between an efficient challenger and an adversary. The adaptive security of injective one-way functions cannot be modeled in such a way as no efficient challenger can implement the inversion oracle in the game with the adversary.

[8]Here, by security preserving, it means that the security reduction uses an adversary breaking the security of the cryptographic scheme under analysis w.r.t. one security parameter n, to break the underlying hardness assumption w.r.t. the same security parameter $n' = n$. On the other hand, if n' is different from, in particular smaller than n, the reduction is said to be non-security preserving.

ing non-security preserving reductions) can be used to overcome this barrier:

Question 3: *Can we construct non-interactive or 2-round non-malleable commitment from super-polynomial hardness assumptions?*

1.3.1 Our Result

In this thesis, in Chapter 5, we answer the above question in positive by demonstrating the existence of a two-round concurrent non-malleable commitment scheme based on sub-exponential hardness assumptions—notably, assuming the existence of the following primitives (all with subexponential security): (1) non-interactive commitments, (2) ZAPs (i.e., 2-round witness indistinguishable proofs) [69], (3) collision-resistant hash functions, and (4) a "weak" time-lock puzzle [70].

Primitives (1),(2),(3) are all very commonly used and can be based on e.g., the discrete log assumption and the RSA assumption. Primitive (4) deserves some more discussion: *Time-lock puzzles*—roughly speaking, puzzles that can be solved in "brute-force" in time 2^t, but cannot be solved "significantly faster" even using parallel computers—were proposed by Rivest, Shamir, and Wagner in 1996 [70] (following May's work on timed-release cryptography [71]), and have since been quite extensively used in the area of timed-release cryptography.

A bit more precisely, a $(T(\cdot), B(\cdot))$-*time-lock puzzle* enables a "sender" to efficiently generate a puzzle puzz with a designated "level" of hardness $t = t(n)$ along with its unique solution s, where n is the security parameter, so that: (i) the puzzle solution can be found in (uniform) time 2^t, but (ii) the puzzle solution cannot be recovered by any attacker of size at most $B(n) > 2^t$ with (parallel) running-time (i.e., circuit depth) at most $T = T(t)$ (where $T(t) << 2^t$ determines the "hardness gap" of the puzzle).[9]

[9] Time-lock puzzles as defined are falsifiable as the challenger can efficiently (in time $\mathsf{poly}(n)\, t, n = \mathsf{poly}(n)\, n$) sample a puzzle puzz together with its unique solution s.

Typical applications of time-lock puzzles only require security against polynomial-size attackers, thus it suffices to let $B(\cdot)$ be any slightly super-polynomial function; however, they require the hardness gap to be very small—namely, $T = 2^{\delta t}$ or even $T = \delta 2^t$ for some $\delta < 1$ (i.e., the problem is inherently "sequential" and the honest puzzle solver is essentially optimal, even if you have access to parallel computers). In this work, we will need security against subexponential-size attackers, but in contrast, only require the existence of a time-lock puzzle with a relatively "large" hardness gap—we only need the puzzle to be hard to break for time $T = 2^{t^\varepsilon}$ for some constant $0 < \varepsilon < 1$.

Theorem 2-rnd (Informal) . *Let T and B be two arbitrary subexponential functions. Assume the existence of non-interactive commitments, a ZAP, a family of collision-resistant hash functions, all with subexponential-security, and the existence of a (T, B)-time-lock puzzle. Then, there exists a 2-round concurrent non-malleable commitment.*

Time-lock puzzles of the above form can be instantiated naturally from the original construction due to Rivest, Shamir and Wagner. We elaborate on this shortly but first provide details on our construction.

1.3.2 Template of Our Construction

Our final construction is modular and can be broken down in two distinct tasks each requiring new techniques. We formally describe these tasks first and then provide a high level overview of our solution.

Task 1: $O(1)$-bit Non-malleable Commitments. We adopt the formulation of non-malleable commitments w.r.t. *identities* which assumes that the players have identities of certain length ℓ, and that the commitment protocol depends on the identity of the committer, which is also referred to as the *tag* of the interaction. Non-malleability ensures

that, as long as the tags of the left and right commitments are different (that is, the man-in-the-middle does not copy the identity of the left committer), no man-in-the-middle attacker can "maul" a commitment it receives *on the left* into a commitment of a related value it gives *on the right*. This is formalized by requiring that for any two values v_1, v_2, the values the man-in-the-middle commits to after receiving left commitments to v_1 or v_2 are indistinguishable.

The length ℓ of the tags can be viewed as a quantitative measure of how non-malleable a scheme is: An ℓ-bit tag non-malleable commitment gives a family of 2^ℓ commitment schemes — each with a hardwired tag — that are "mutually non-malleable" to each other. Full-fledged non-malleable commitments have tags of length equal to the security parameter $\ell = n$, and hence corresponds to a exponentially sized family. Therefore, the shorter the tags are, the easier it is to construct such a family. However, when the number of communication rounds is restricted to 2, Pass [67] showed that even the weakest non-malleable commitment for just *1-bit tags*, corresponding to a size 2 family, cannot be reduced from falsifiable assumptions, via a polynomial-time black-box reduction. This begs the following question:

Question 3.1 *Can we construct ℓ-bit non-interactive or 2-round non-malleable commitment from super-polynomial hardness assumptions for $\ell = O(1)$ or even $\ell = 1$?*

Task 2: Upgrading Non-malleability. A positive answer to the above question already requires circumventing the Pass's [67] but still falls short of the eventual goal of constructing full-fledged non-malleable commitments which supporting n-bit identities. Previous work were able to increase the length of identities from $O(1)$ to n by providing a transformation that takes in a t-bit non-malleable commitment and outputs one on 2^{t-1}-bits – we refer to this task as *upgrading non-malleability*. The works of Lin and

Pass [56], later improved by Wee [59], gave such a transformation. Then, starting from $O(1)$-bit non-malleable commitments, one could apply such a transformation $O(\log^* n)$ thereby resulting in a non-malleable commitment scheme for n-bit identities. However, the known transformations inccur an additive, constant overhead in the round-complexity whereas we would like to keep the round complexity to just two. Therefore,

Question 3.2 *For $2 < t(n) \leq n$, given a t-bit, two-round non-malleable commitment can we construct 2^{t-1}-bit, two-round non-malleable commitment from super-polynomial time assumptions?*

Our Solution to Task 1 – Synergy between Multiple Hardness Axes. In cryptography, the power, or *resource*, of attackers is usually measured by their running-time when represented as Turing machines, or equivalently by their circuit-size when represented as circuits. Time-lock puzzles, and more generally timed-release cryptography [71, 72, 73, 74, 75], on the other hand, measure the resource of attackers by their parallel running-time or equivalently by their circuit-depth. Our 2-round non-malleable commitments crucially rely on the synergy between these two types of resources. The key idea is, instead of measuring the hardness of commitment schemes in a single "axis" of resource, measure the hardness in two axes, one refers to circuit-size and the other to circuit-depth. By doing so, we can construct a pair of commitment schemes $\mathsf{Com}_1, \mathsf{Com}_2$ that are simultaneously harder than the other, in different axes. In particular, Com_2 is harder in the axis of *circuit-size*, in the sense that Com_1 admits an extractor of size S while Com_2 is secure against all circuits of size S; on the other hand, Com_1 is harder in the axis of *circuit-depth*, in the sense that Com_2 admits an extractor of depth D (and some size S') while Com_1 is hiding against all circuits with depth D (and size S'). Such a pair of commitment schemes that are mutually harder than each other already has a weak flavor of non-malleability — no adversary can "maul" a Com_2 commitment to v into

a Com_1 commitment to a related value, say $\tilde{v} = v + 1$, as otherwise one can extract $\tilde{v}$ in size S, which violates the hiding of Com_2 against S-size circuits. Similarly, no adversary can "maul" a Com_1 commitment into a Com_2 commitment, as otherwise, we can find $\tilde{v}$ in small depth D (and size S'), which violates the hiding of Com_1 against depth D circuits (of size S'). This already gives us a non-interactive (hence also 2-round) non-malleable commitment on 1-bit identities, thereby circumventing Pass's [67] lower-bound.

Our Solution to Task 2 – Round-preserving Tag Amplification. Next, we amplify this weak non-malleability to full-fledged non-malleability. More precisely, we transform the aforementioned commitment schemes, which are non-malleable w.r.t. short "tags" to that for much longer "tags", while keeping two rounds. Our transformation again crucially relies on time-lock puzzles and more specifically the ability to design mutually non-malleable cryptographic primitives. We provide an elaborate discussion on this in Chapter 5.

1.3.3 Instantiating Time-lock Puzzles and Extensions

Time-lock Puzzles. The original construction of time-lock puzzles due to Rivest, Shamir, and Wagner [70] is based on the hardness of a very natural strengthening of the factoring problem referred to as the *repeated squaring problem*: given a random RSA-modulus $N = pq$, and a random (or appropriately chosen) element g, compute

$$g^{2^{2^t}} \bmod N$$

Clearly, this can be done using 2^t repeated squarings. The RSW assumption is that this task cannot be significantly sped up, even using parallel resources, as long as the total resource of the adversary does not enable factoring N. Given the current state-of the art, the repeated squaring problem appears to be hard for *strongly exponential* parallel-

time: $T(t) = \delta 2^t$ (that is, basically, no non-trivial speed-up over repeated squaring is possible); indeed, this strong assumption is typically used in the literature on timed-release cryptography (in fact, several significantly stronger versions of this assumption, where additional leakage is given, are also typically considered—see e.g., the "generalized Blum-Blum-Shub assumption" of Boneh-Naor [75].)

Since we only need a "weakly"-secure time-lock puzzle where the hardness gap is large, it suffices for us to make a significantly weaker, *subexponential*, repeated squaring assumption, that is,

$$2^t \textit{ repeated squarings (modulo } N = pq\textit{) cannot be done in parallel-time } 2^{t^\varepsilon}$$

More formally:

Assumption 1 (Subexp. Repeated Squaring Assumption, Informal) *There exists subexponential functions T, B such that for every function $t(n) \in \omega(\log n) \cap n^{O(1)}$, the following holds: For every size $B(\cdot)$-attacker A with (parallel) running-time (i.e,. circuit depth) at most $T(t(\cdot)) < B(\cdot)$, there exists a negligible function μ such that for every $n \in \mathbb{N}$, the probability that A, given g, N where N is a randomly chosen n-bit RSA-modulus, and g is a randomly chosen (or appropriately fixed) element in Z_N^*, can compute $g^{2^{2^{t(n)}}} \bmod N$ is bounded by $\mu(n)$.*

We note that essentially the repeated squaring assumption has two security parameters, n and $t(n)$, where the former decides the size of the modulus and the maximal size $B(n)$ of the adversaries (such that factoring the modulus remains hard), and the latter decides the number $2^{t(n)}$ of repeated squaring needed to solve the puzzle by brute force, and the maximal depth $T(t(n))$ of the adversaries. The assumption says that the puzzle is hard for adversaries of depth up to $T(t(n))$ and size up to $B(n)$, even if the size of the adversary may be larger than $T(t(n))$ or even $2^{t(n)}$ (but still bounded by $B(n)$). Note

also that the subexponential repeated squaring assumption implies the subexponential hardness of factoring.[10]

We remark that comparing with other subexponential assumptions (such as e.g., the subexponential DDH assumption), the subexponential repeated squaring assumption is milder in the sense that it is a search assumption instead of a decisional assumption. It also has a strong "win-win" flavor: Repeated squaring is a problem that arises naturally in the design of algorithms (e.g., any improvement on repeated squaring would yield improved efficiency for the verification of RSA-based signatures.) On the other hand, the subexponential repeated squaring assumption has a non-standard form in that the puzzle is easy to solve in depth $2^{t(n)}$, but hard to solve in depth $2^{t(n)^\varepsilon}$ and size more than $2^{t(n)}$ and below $B(n)$.

We finally mention that the time-lock puzzle needed for our construction can also be based on the existence of a parallel-time hard language and indistinguishability obfuscation (with subexponential security) by the work of Bitansky *et al.* [76].

Towards Non-interactive Non-malleable Commitments. We also address the question of whether fully non-interactive (i.e., single-message) non-malleable commitments are possible. We show that if we replace the assumption of the existence of ZAPs (i.e., two-message witness indistinguishability) with non-interactive witness indistinguishable proofs (NIWI) [77, 78, 79], and the existence of families of collision-resistant hash functions for a *single*, collision-resistant hash function secure against *uniform* adversaries, [80, 81], then a slightly modified *non-interactive* version of our protocol satisfies concurrent non-malleability w.r.t. *uniform attackers*: Basically, the first message of our two-round protocol only contains the first message of the ZAP, and the index of the hash

[10]The state-of-the-art factoring algorithm runs in 2^{n^ε} time for some constant ε. The subexponential hardness of factoring assumes that factoring is hard for 2^{n^μ} time adversaries for some smaller constant $\mu < \varepsilon$.

function, so by relying on a NIWI and a single hash function (secure against uniform subexponential-time attackers), the first message can be skipped.

Theorem 1-rnd (Informal) *Let T and B be two arbitrary subexponential functions. Assume the existence of non-interactive commitments, a NIWI, a uniform collision-resistant function, all with subexponential-security, and the existence of a (T, B)-time-lock puzzle. Then, there exists a one-message concurrent non-malleable commitment secure w.r.t. uniform polynomial-time adversaries.*

We leave open the question of whether we can get a non-interactive non-malleable commitment w.r.t. non-uniform attackers.

Achieving Chosen Commitment Attack Security. Canetti, Lin, and Pass [82, 83] strengthened the notion of concurrent non-malleability to security against Chosen Commitment Attacks (CCA) for commitments, analogous to the extensively studied notion of security against Chosen-Ciphertext Attacks for encryption schemes. Roughly speaking, a commitment scheme is said to be CCA-secure if commitments remain hiding even against attackers with access to an inefficient oracle, called the committed-value oracle, that "breaks" each commitment sent by an attacker using brute force and returns the (unique) committed value as soon as the commitment is completed. In particular, CCA-security implies that it is infeasible for an attacker to "maul" commitments to a set of values into commitments to a set of related values, even with the help of the committed-value oracle—which implies concurrent non-malleability. It was shown in several works [82, 83, 84, 85] that CCA-secure commitments are useful for constructing multi-party computation protocols with concurrent and composable security in the plain model from polynomial-time hardness assumptions. Furthermore, in a recent work [86], 2-round CCA-secure commitments are further used for constructing round-optimal, 4-round, multi-party computation protocols secure in the stand-alone setting. We show

that our two-round, and non-interactive non-malleable commitments, in fact, satisfy the stronger notion of CCA security.

Theorem (Informal) *The two-round non-malleable commitment scheme presented in Theorem 2-rnd satisfies CCA-security, and the non-interactive non-malleable commitment scheme presented in Theorem 1-rnd satisfies CCA-security w.r.t. uniform polynomial-time adversaries.*

A Remark on "Sub-subexponential" Security. Let us finally mention that although for the simplicity of notation we rely on subexponential hardness assumption, our actual proof reveals that we only need to rely on "sub-subexponential"[11] hardness assumption for all the primitives we rely on: namely, we only require security to hold w.r.t. attackers of size (and depth) $2^{n^{1/\log\log n}}$ (and in fact, even slightly less).

1.4 Organization of Chapters

The remainder of this dissertation is organized as follows:

1. In Chapter 2, we study the problem of constructing PRFs from non-adaptive PRFs and present as our main result a proof that highly-efficient black-box transformations from naPRF to PRF, those making one-call, do not exist. Our result complements existing impossibility results (Myers, EUROCRYPT '04; Pietrzak, CRYPTO '05) ruling out natural specific approaches, such as parallel and sequential composition. Furthermore, we show that our techniques extend to rule out a natural class of constructions making parallel but arbitrary number of calls which in particular includes parallel composition and the two-call, cuckoo- hashing based construction of Berman et al. (Journal of Cryptology, '19).

[11] We refer to $2^{n^{o(1)}}$ as a sub-subexponential function.

2. In Chapter 3 and Chapter 4, we study the problem of constructing public-seed PRPs from public-seed PRFs. Here, first in Chapter 3 we introduce the notion of public-seed pseudorandomness of which public-seed PRPs and public-seed PRFs (equivalently Universal Computational Extractors introduced by Bellare, Hoang, and Keelveedhi (CRYPTO '13)) are specific instantiations. Then, we show that the five-round Feistel construction is a public-seed PRP when its round functions are public-seed PRFs. In Chapter 4 we improve this result by showing the two-call construction by Naor and Reingold (STOC '97) is a public-seed PRP when its round functions are public-seed PRFs. This matches the complexity (in terms of number of calls) of constructing PRPs from PRFs.

3. In Chapter 5 we study the problem of constructing non-malleable commitments and present the first construction for two-round non-malleable commitments from sub-exponential time well-studied cryptographic assumptions. Such commitments were previously shown to be impossible from polynomial-time falsifiable assumptions under black-box security reductions (Pass TCC'13). We achieve this in two steps: (a) construct non-interactive non-malleable commitments for $O(1)$-bit identities and (b) present a transformation that upgrades two-round non-malleable commitment on t-bit identities to 2^{t-1}-bit identities.

1.5 Permissions and Attributions

All results in this dissertations have either appeared in conference proceedings or journals. More specifically,

1. The contents of Chapter 2 are based on the results appearing in [87]: Soni P., Tessaro S. (2020) On the Query Complexity of Constructing PRFs from Non-

adaptive PRFs. In: Galdi C., Kolesnikov V. (eds) Security and Cryptography for Networks - SCN 2020. SCN 2020. Lecture Notes in Computer Science, vol 12238. Springer, Cham. The final authenticated version is available online at https://doi.org/10.1007/978-3-030-57990-6_27.

2. The contents of Chapter 3 are based on the results appearing in [38]: Soni P., Tessaro S. (2017) Public-Seed Pseudorandom Permutations. In: Coron JS., Nielsen J. (eds) Advances in Cryptology – EUROCRYPT 2017. EUROCRYPT 2017. Lecture Notes in Computer Science, vol 10211. Springer, Cham. © IACR 2018, https://doi.org/10.1007/978-3-319-56614-6_14.

3. The contents of Chapter 4 are based on the results appearing in [88]: Soni P., Tessaro S. (2018) Naor-Reingold Goes Public: The Complexity of Known-Key Security. In: Nielsen J., Rijmen V. (eds) Advances in Cryptology – EUROCRYPT 2018. EUROCRYPT 2018. Lecture Notes in Computer Science, vol 10822. Springer, Cham. © IACR 2017, https://doi.org/10.1007/978-3-319-78372-7_21.

4. The contents of Chapter 5 are based on [89] and its extended version [90]

   - H. Lin, R. Pass and P. Soni, "Two-Round and Non-Interactive Concurrent Non-Malleable Commitments from Time-Lock Puzzles," 2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS), Berkeley, CA, 2017, pp. 576-587. © IEEE 2017 https://doi.org/10.1109/FOCS.2017.59.

   - H. Lin, R. Pass and P. Soni, "Two-Round and Non-Interactive Concurrent Non-Malleable Commitments from Time-Lock Puzzles," SIAM Journal of Computing (SICOMP), 2020. Copyright © by SIAM. Unauthorized reproduction of this article is prohibited. https://doi.org/10.1137/17M1163177.

Chapter 2

Non-adaptive PRFs to PRFs

The main result of this chapter is a proof that highly-efficient black-box transformations from naPRF to PRF, those making one-call, do not exist. Towards showing this we first give an overview of our techniques in Section 2.1 then introduce some preliminary notation in Section 2.2. Then, we describe our main theorem in Section 2.3 and provide an overview of its proof. We dedicate Section 2.4 through Section 2.6 to give a formal proof. Finally, we reserve Section 2.7 and Section 2.8 to discuss some special cases and extensions of our main result.

2.1 Technical Overview

The study of black-box separations for cryptographic primitives was initiated by the seminal paper of Impagliazzo and Rudich [91] which provided a framework (later formalized by [92]) to provide such results. Impagliazzo and Rudich observed that fully black-box constructions relativize w.r.t. any oracle and hence to rule out fully black-box constructions it suffices to show the existence of an oracle relative to which there exists a naPRF H but $\mathsf{F}^{\mathsf{H}}[\mathsf{C}, \mathsf{G}]$ is not a PRF. Furthermore, Gertner, Malkin and Reingold [93]

Construction	Evaluation $y = \mathsf{F}((s,k),x)$	Rule out for any $c = O(1)$
$\mathsf{F}^{\mathsf{H}}[\mathsf{C},\mathsf{G}]$ Section 2.3	$y = \mathsf{G}(s,x,z)$ where $w = \mathsf{C}(s,x); z = \mathsf{H}(k,w)$	C is c-universal, any r any $m \geq \log(8ce)$
$\mathsf{F}^{\mathsf{H}}[\mathsf{C},g]$ Section 2.7.1	$y = g(x,w,z)$ where $w = \mathsf{C}(s,x); z = \mathsf{H}(k,w)$	any C, g and r any $m \geq (n+r)/c + \omega(\log n)$
$\mathsf{F}^{\mathsf{H}}[\mathsf{C}]$ Section 2.7.2	$y = \mathsf{H}(k,\mathsf{C}(s,x))[1,\ldots,m]$	any C any $m, r = \omega(\log n)$
$\mathsf{F}^{\mathsf{H}}[\kappa,\mathsf{C},\mathsf{G}]$ Section 2.8	$y = \mathsf{G}(s_{\kappa+1},x) \oplus \bigoplus_{i=1}^{\kappa} \mathsf{H}(k_i,\mathsf{C}(s_i,x))$	C is c-universal any κ, r, G

Table 2.1: We rule out fully black-box constructions of PRF F from n bits to m bits of the form described in first and second column from a naPRF H from n bits to r bits, whenever the conditions in the third column are true for some constant $c \geq 2$. C is a (keyed) function family from n bits to n bits and g is a function from $2n+r$ bits to m bits. For first row, G is a function family from $n+r$ bits to m bits and G is a family from n bits to r bits for the last row.

observed that the oracle can depend on the construction F. Our result will be of this flavor.

In the rest of this section, we give a brief overview of our main result (stated as Theorem 25 in the body). For the sake of this overview, we will only focus on a special case of the construction F – a composition of a pre-processing function C and the naPRF H, and rule out F as a fully black-box construction whenever C is an almost-universal function.

1-call pre-processing construction $\mathsf{F}^{(\cdot)}[\mathsf{C}]$. Let H be some function family from n bits to m bits with n-bit keys and let C be function family from n bits to n bits with σ-bit seeds. We consider the function family $\mathsf{F}^{\mathsf{H}}[\mathsf{C}]$ from n bits to m bits that makes oracle calls to H and takes the following form,

$$\mathsf{F}((s,k),x) = \mathsf{H}(k,\mathsf{C}(s,x)) \ .$$

Theorem (Informal). *For any almost-universal* C*, there exists an oracle* (O, R) *relative to which there exists a naPRF* H *such that* $\mathsf{F}^{\mathsf{H}}[\mathsf{C}]$ *is not a PRF.*

Although the approach of providing oracles has been the focus of many black-box separations [94, 95, 16, 17], Myers [1] was the first to apply such techniques in the context of ruling out fully black-box constructions of PRFs from naPRFs, albeit, for restricted forms of constructions. We borrow ideas from Myers [1] to design our oracle, but the general nature of our result brings in a number of challenges as we discuss below. Following [1, 94] we will allow our oracles to make random choices (and hence we give a distribution of oracles) and show that the theorem holds except with negligible probability which suffices to guarantee the existence of an oracle (in the uniform setting).

Oracle (O, R) **and naPRF** H**.** For simplicity of presentation, we will present the oracle as a pair (O, R) instead of a single oracle. The oracle O embeds a natural information-theoretically secure PRF. More formally, for every $n \in \mathbb{N}$ and for every $k \in [2^n]$, $\mathsf{O}(1^n, k, \cdot)$ is implemented by a function from $\mathsf{Funcs}(n, r)$ sampled uniformly and independently at random (with replacement). Relative to O there exists a natural naPRF H where for every $k \in \{0,1\}^n$ and $x \in \{0,1\}^n$,

$$\mathsf{H}^{\mathsf{O}}(k, x) = \mathsf{O}(k, x) \ .$$

We emphasize that H is a naPRF relative to O.

The oracle R is designed to provide a trivial way to break F. While it is easy to come up with such an oracle, we need to ensure that only adversaries making adaptive queries (to F) be able to use R to break the PRF-security of F. For this, we decompose R into $(\mathsf{R}_1, \mathsf{R}_2, \mathsf{R}_3)$ where R_1 (takes no inputs and) returns sufficiently many (say l) random challenges $x_1, \ldots, x_l$ (in the domain of F), R_2 takes $y_1, \ldots, y_l$ (in the range of F) as inputs and returns more random challenges $x_{l+1}, \ldots, x_{2l}$, and R_3 accepts $y_{l+1}, \ldots, y_{2l}$ as inputs

and returns 1 iff there exists a key (s,k) such that $y_i = \mathsf{F}((s,k),x_i)$ for all $i \in [2l]$. Note that like R_1, R_2 also provides random challenges but, additionally, forces an adversary to commit to responses $y_1,\ldots,y_l$ for challenges $x_1,\ldots,x_l$ issued by R_1. This property of R will be crucial to show the naPRF security of H relative to both O and R.

$\mathsf{F}^{\mathsf{H}}[\mathsf{C}]$ is not a PRF relative to (O,R). R provides a trivial way for an adaptive adversary to break F. An adversary A^f relative to (O,R) can provide $y_i = f(x_i)$ to R_3 for challenges x_i (issued by R_1 and R_2) by adaptively querying f. When $f = \mathsf{F}((s,k),\cdot)$ for some randomly sampled (s,k), R clearly outputs 1. In the random world (when $f \xleftarrow{\$} \mathsf{Funcs}(n,r)$), R outputs 1 if for the randomly chosen n-bit strings y_i's there exists some (s,k) for which $\mathsf{F}((s,k),x_i) = y_i$ for all i. Since there are only $2^{\sigma+n}$ such (s,k)'s in F, the probability of this happening is upper bounded by $2^{\sigma+n}/2^{2ml}$, which is negligible for sufficiently large l. Therefore, A breaks F relative to (O,R).

H remains naPRF relative to (O,R)? To conclude the theorem, we need to show that the above construction of H remains naPRF relative to (O,R).[1] Unfortunately, despite the adaptive nature of R, this is not true in general. Consider the following universal family C for which for some s^{bad} we have,

$$\mathsf{C}(s^{\mathsf{bad}},x) = \begin{cases} 0^n & \mathsf{lsb}(x) = 0\ , \\ 1^n & \mathsf{lsb}(x) = 1\ . \end{cases}$$

And for all $s \neq s^{\mathsf{bad}}$, $\mathsf{C}(s,\cdot)$ is a permutation. Now consider the following non-adaptive adversary A^{na} relative to (O,R) which breaks H and makes *only two* non-adaptive queries to the challenge oracle h. A^{na} first queries h on $Q = \{0^n,1^n\}$ and then computes $y_i = \mathsf{F}^h(s^{\mathsf{bad}},x_i)$ for any adaptive challenges x_i's provided by $\mathsf{R}_1,\mathsf{R}_2$, where the construction

[1] We emphasize that the non-adaptivity restriction on the adversary is only on the challenge oracle in the naPRF-security game. It can query the oracle (O,R) adaptively.

$\mathsf{F}^h(s,\cdot)$ replaces the calls to H in $\mathsf{F}^{\mathsf{H}}(s,\cdot)$ with calls to h. Recall that R_3 returns 1 if it finds any (s,k) such that $y_i = \mathsf{F}((s,k),x_i)$. In the real world $h = \mathsf{H}(k,\cdot)$ and therefore R_3 returns 1 as for (s^{bad},k), $\mathsf{F}((s^{\mathsf{bad}},k),x_i) = y_i$. However, the probability that R_3 returns 1 when h is a randomly sampled function can be upper bounded by the probability that there exists some $k \in \{0,1\}^n$ such that $\mathsf{H}(k,x) = h(x)$ for $x \in \{0^n,1^n\}$. Since there are at most 2^n k's and h is a random function, the above happens only with probability $2^n/2^{2n}$ which is negligible and hence R_3 returns 0, thereby breaking H. This highlights an important bug in the design of the oracle R – it allows for a non-adaptive adversary A^{na} to use it effectively in breaking the naPRF-security of H by exploiting weaknesses in the pre-processing function family C.

Oracle R Revisited. The issue with the previous oracle was that R accepts a seed s^{bad} for which a non-adaptive adversary A^{na} can make few (polynomially many) queries (let Q be this set of queries) to its oracle h and compute the entire function $\mathsf{F}^h(s^{\mathsf{bad}},\cdot)$, thereby provoking R_3 to output 1 in the real world. We ask R_3 to ignore "bad" seeds. A seed s is β-good if for every w in the range of $\mathsf{C}(s,\cdot)$ $\mathsf{Pr}_x[\mathsf{C}(s,x) = w] \leq \beta$. Let β be some negligible function (we will explain how to set this later), we modify R_3 to return 1 iff it finds some (s,k) where s is β-good. A consequence of this is that for any polynomially sized Q and any s that is β-good, it is only with negligible probability that $\mathsf{C}(s,x) \in Q$ for a random x.

Security of H Revisited. For simplicity let us assume that R_1 and R_2 just output one random challenge each (i.e., $l = 1$). Let x_1 and x_2 be those random challenges. Let us fix some s that is β-good. Let $w_i = \mathsf{C}(s,x_i)$. We are interested in A^{na} that trigger R_3 to output 1 for this fixed seed. Any $\mathcal{A}^{\mathsf{na}}$ has two choices: either (1) make its queries to h after learning x_1 but before learning x_2 or (2) make queries after learning x_2 (after

committing to y_1). Let Q be the set of queries A^{na} made to h. In case (1) since x_2 is sampled randomly and s is β-good, $\mathsf{C}(s, x_2) \notin Q$. To succeed, A^{na} now needs to correctly guess the $y_2 = h(\mathsf{C}(s, x_2))$ which happens with prob. 2^{-m} as h is random. In case (2) to succeed, A^{na} can just hope that $h(\mathsf{C}(s, x_1)) = y_1$ which also happens with prob., 2^{-m} as h is random. Therefore, it is only with prob. 2^{-m} that A^{na} can trigger R_3 to output 1 for a fixed β-good seed s. Instead of sampling one challenges each, if R_i's had sampled tuples $X_1 = (x_1, \ldots, x_l)$ and $X_2 = (x_{l+1}, \ldots, x_{2l})$ then in (1) we can show that, except with probability $(\beta q)^{l/2}$, at least $l/2$ of the $\mathsf{C}(s, X_2[i])$'s fall outside Q leading to the probability of success of A^{na} to drop to $2^{-ml/2}$, and in (2) the probability of success instead drops to 2^{-ml}. Therefore, a union bound over all β-good seeds would show A^{na} fails in triggering R_3 to output 1 (for $l = \omega(\sigma)$). In other words, by sampling tuples X_1, X_2 and considering only β-good seeds, R has now rendered itself useless to a non-adaptive adversary allowing us to reduce the naPRF-security of H relative to (O, R) to naPRF-security of H relative to *only* O. One issue that still needs to be addressed is to ensure that the new R continues to allow to break F. This is where the universality is crucial. We show that a randomly sampled s is indeed β-good for an appropriate β.

Comparison to [1]. Our oracles are similar to Myers [1] except that they are significantly more complicated. Myers rules out arbitrary number of parallel compositions of H. In its simplest form (2-call case) Myers construction can be viewed in terms of our preprocessing function $\mathsf{F}^{\mathsf{H}}[\mathsf{C}]$ where $\mathsf{C} = \mathsf{H}$ and hence C is also implemented from O. Therefore, in the non-adaptive security proof, the adversary has very little information about the structure of C. This is unlike our case where it was the structure of C, more importantly, the existence of a single "bad" seed that allowed A^{na} to break H relative to trivial attempts of designing R.

Extending to our general one-call result. Although the preprocessing case captures our core ideas, ruling it out is considerably simpler than our more general construction. An important property that the construction enjoys is that for every x, s, y the probability that $\mathsf{F}^f(s, x) = y$ for a random f is 2^{-m}. We refer to such constructions are "unbiased". When moving on to constructions with post-processing, such guarantees are not readily available making the proof difficult. In addition, working with weaker notion of c-universality for $c > 2$ brings in additional challenges. We detail our formal proof in Section 2.3.

On ruling out two- or more call constructions. The main result of this work is ruling out a large class of constructions as a PRF, that make only one call to an underlying naPRF. A natural question is to understand whether such separations can be proved for constructions making two calls or more generally $O(1)$ number of calls. We devote Section 2.8 for this. More specifically, (1) In Section 2.8.1 we show that our techniques from the 1-call case can be lifted to rule out a specific 2-call construction (even its generalization to arbitrary number of calls). We note that Berman et al. [8] studied a (non-black-box[2]) variant to construct a PRF from a naPRF. (2) The oracle (O, R) used to rule out one-call constructions admits natural extensions to constructions that make more than one call, we describe in Section 2.8.2 two explicit constructions making two-calls and four-calls respectively which also allow a non-adaptive adversary to break the underlying naPRF relative to (O, R). With these examples we hope to highlight that a general result that rules out all $O(1)$-call constructions will require new techniques or at least new oracle designs.

[2]the construction depends on the security of the underlying primitives

2.2 Preliminaries

For $n, m \in \mathbb{N}$, $\mathsf{Funcs}(n, m)$ denotes the set of all functions $\{0,1\}^n \to \{0,1\}^m$. By $[n]$ we denote the set $\{1, \ldots, n\}$. By $d \xleftarrow{\$} D$ we denote the process of sampling a random element from some finite set D and assigning it to d. For $l \in \mathbb{N}$, $(d_1, \ldots, d_l) \xleftarrow{\$} (D)^l$ and $(d_1, \ldots, d_l) \xleftarrow{\$} (D)^{[l]}$ denote the process of sampling l elements from D where each d_i is sampled independently and uniformly from D with and without replacement, respectively. For $l, p \in \mathbb{N}$ and $f \in \mathsf{Funcs}(n, m)$, $X = (x_1, \ldots, x_l)$ denotes an ordered tuple where $x_1, \ldots, x_l \in \{0,1\}^n$ and $X[i]$ denotes the i-th element in the tuple. $f(X)$ denotes the ordered tuple $(f(x_1), \ldots, f(x_l))$. For $X = (x_1, \ldots, x_l)$ and $Y = (y_1, \ldots, y_p)$, by $X||Y$ we denote $(x_1, \ldots, x_l, y_1, \ldots, y_p)$. We use capital letters to denote both tuples and sets, our usage will be clear from the context. A function $\alpha : \mathbb{N} \to \mathbb{R}_{\geq 0}$ is negligible if for every $c \in \mathbb{N}$, there exists n_0 such that $\alpha(n) \leq n^{-c}$ for all $n \geq n_0$.

Function Families. For polynomially bounded functions $m, \sigma : \mathbb{N} \to \mathbb{N}$, a *function family* $\mathsf{F} = (\mathsf{F.Kg}, \mathsf{F.Eval})$ from n bits to m bits with σ-bit keys/seeds consists of two polynomial-time algorithms – the *key (or seed) generation algorithm* $\mathsf{F.Kg}$ and the *evaluation algorithm* $\mathsf{F.Eval}$. In particular, $\mathsf{F.Kg}$ is a randomized algorithm that on input the security parameter 1^n returns a key k sampled uniformly from $\{0,1\}^{\sigma(n)}$. $\mathsf{F.Eval}$ is a deterministic algorithm that takes three inputs: 1^n, key $k \in \{0,1\}^{\sigma(n)}$ and query $x \in \{0,1\}^n$ and returns an $m(n)$-bit string $y = \mathsf{F.Eval}(1^n, k, x)$. We generally write $\mathsf{F}(1^n, k, \cdot) = \mathsf{F.Eval}(1^n, k, \cdot)$ and even drop the first input (i.e., 1^n) of both Kg and Eval for ease of notation. By $f \xleftarrow{\$} \mathsf{F}$ we denote the process of sampling $k \xleftarrow{\$} \mathsf{F.Kg}$ and assigning $f = \mathsf{F}(k, \cdot)$.

Oracle Function Families. In this work, we consider function families F where $\mathsf{F.Kg}$ and $\mathsf{F.Eval}$ can make queries to another function family modeled as an oracle O. We

refer to such families as oracle function families and denote it by $\mathsf{F}^{(\cdot)}$ and by F^{O} when the underlying oracle is O. By $\mathsf{F}^{(\cdot)}[\mathsf{C}]$ we denote function family F having access to the entire description of the function family C.

Universal Function Families. Below we define a generalization of the well-known notion of almost α-universal hash function family.

Definition 1 *For polynomially bounded functions m, σ, let C be a function family from n bits to m bits with σ-bit seeds, α be some function from $\mathbb{N}$ to $\mathbb{R}_{\geq 0}$, and $c \in \mathbb{N}$. We say that C is (α, c)-universal family if for all $n \in \mathbb{N}$, every $X \in (\{0,1\}^n)^{[c]}$,*

$$\Pr_{s \xleftarrow{\$} \mathsf{C.Kg}(1^n)} [\mathsf{C}(s, X[1]) = \mathsf{C}(s, X[2]) = \ldots = \mathsf{C}(s, X[c])] \leq \alpha(n) \ .$$

We retrieve the standard notion of almost α-universal hash function family when $c = 2$. Whenever α is a negligible function, we refer to C as a c-universal function family. We emphasize that the reader should not confuse our notion of c-universality with the notion of c-wise independent hashing.

2.2.1 (Non-) Adaptive PRFs Relative to Oracles

In this work, we consider pseudo-randomness of function families relative to an oracle which we define next.

Definition 2 *Let m be a polynomially bounded function over $\mathbb{N}$ and O be some oracle. Let $\mathsf{F}^{(\cdot)}$ be an oracle function family from n bits to m bits. For probabilistic polynomial-time (PPT) distinguisher A, let*

$$\mathsf{Adv}^{\mathsf{rel-prf}}_{A,\mathsf{F},\mathsf{O}}(n) = \left| \Pr_{f \xleftarrow{\$} \mathsf{F}} [A^{f^{\mathsf{O}},\mathsf{O}}(1^n) = 1] - \Pr_{g \xleftarrow{\$} \mathsf{Funcs}(n,m(n))} [A^{g,\mathsf{O}}(1^n) = 1] \right| \ ,$$

where probability is also taken over the random coins of A.

We say that $\mathsf{F}^{(\cdot)}$ *is a pseudo-random function (PRF) relative to oracle* O *if for all PPT distinguishers* A $\mathsf{Adv}^{\mathsf{rel-prf}}_{A,\mathsf{F},\mathsf{O}}(1^n)$ *is negligible in* n. $\mathsf{F}^{(\cdot)}$ *is a non-adaptive PRF (naPRF) relative to* O *if the above is true for all PPT distinguishers that only make non-adaptive queries to the challenge oracle* f/g.

In naPRF definition, we require that A only make non-adaptive queries to the challenge oracle f/g and can query O adaptively. In the absence of the oracle O we recover the standard notions of PRFs and naPRFs. Although as stated the oracle O is deterministic, in this work we will consider randomized oracles O and the above probabilities is taken also over the random choices made by O.

2.2.2 Black-Box Separations

The study of black-box separations for cryptographic primitives was initiated by the seminal paper of Impagliazzo and Rudich [91] which provided a framework (later formalized by [92]) to provide such results. They observed that fully black-box constructions relativize w.r.t. any oracle and hence to rule out fully black-box constructions it suffices to show the existence of an oracle relative to which there exists a naPRF H but $\mathsf{F}^{\mathsf{H}}[\mathsf{C},\mathsf{G}]$ is not a PRF. Furthermore, Gertner, Malkin and Reingold [93] observed that the oracle can depend on the construction F.

Theorem 1 ([93]) *An oracle function family* $\mathsf{F}^{(\cdot)}$ *is not a fully BB construction of a PRF from naPRF if there exists an oracle* O *and an oracle function family* $\mathsf{H}^{(\cdot)}$ *such that* H^{O} *is a naPRF relative to* O *but* F^{H} *is not a PRF relative to* O.

When restricting to uniform adversaries (which is the focus of this work) it is sufficient to exhibit a oracle that makes random choices (or a distribution of oracles) and show that Theorem 1 holds except with negligible probability. This is the approach adopted

in all previous works on black-box separations. We formally state this as the following Proposition.

Proposition 1 *An oracle function family* $\mathsf{F}^{(\cdot)}$ *is not a fully BB construction of a PRF from naPRF if there exists a randomized oracle* O *and an oracle function family* $\mathsf{H}^{(\cdot)}$ *such that* H^{O} *is a naPRF relative to* O *but* F^{H} *is not a PRF.*

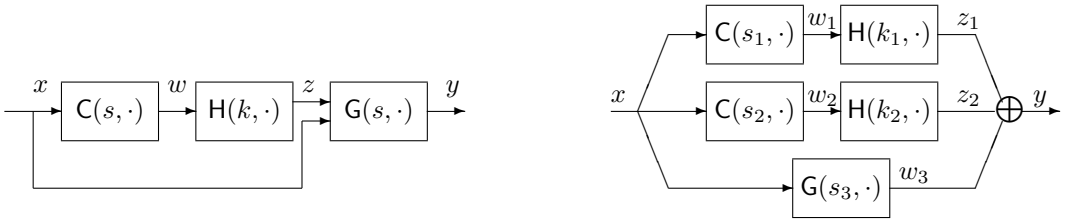
Theorem 1 for the uniform setting follows from Proposition 1 by relying on the Borel-Cantelli Lemma and on the countability of the family of uniform Turing machines. All results in this work will be of the flavor of Proposition 1. Establishing Proposition 1 w.r.t. non-uniform adversaries may not be sufficient to lift BB separations to the non-uniform model due to the uncountability of non-uniform Turing Machines. We leave it to future work to lift our results to the non-uniform setting, following ideas from [96, 16].

2.3 Main Result: Ruling out 1-call F's

In Section 2.3.1 we formally describe the class of one-call constructions to which our separation result applies. Then in Section 2.3.2 we state our main result and provide its proof's overview in Section 2.3.3.

2.3.1 General 1-call Construction

Let σ, r, m be any polynomially bounded functions. Let C be a function family from n bits to n bits with σ-bit seeds, let H be a function family on n bits to r bits with n-bit seeds and let G be a function family from $n+r$ bits to m bits with σ-bit seeds. Consider the family $\mathsf{F}^{\mathsf{H}}[\mathsf{C},\mathsf{G}]$ (depicted in Figure 2.1a) from n bits to m bits with $\sigma+n$-bit seeds such that for every $n \in \mathbb{N}$, $\mathsf{F}.\mathsf{Kg}(1^n)$ outputs (s,k) where s and k are randomly chosen



(a) 1-call construction (b) 2-call construction

Figure 2.1: (a) General 1-call construction $\mathsf{F}^{\mathsf{H}}[\mathsf{C},\mathsf{G}]$ where C (resp., G) is a family from n bits (resp., $n+r$ bits) to n bits (resp., m bits) with σ-bit keys and H is a function family from n bits to r bits with n-bit keys. Figure shows the evaluation of F on input x and key (s,k) where s is the key for both C and G and k is key for H. (b) Two-call construction $\mathsf{F}^{\mathsf{H}}[\mathsf{C},\mathsf{G}]$ where C (resp., G) is a family from n bits (resp., n bits) to n bits (resp., m bits) with σ-bit keys and H is a function family from n bits to m bits with n-bit keys. Figure shows the evaluation of F on input x and key $(\vec{s},\vec{k})$ where $\vec{s}=(s_1,s_2,s_3)$ and $\vec{k}=(k_1,k_2)$ is key for H.

$\sigma(n)$-bit seeds for both C and G, and n-bit key for H respectively. The evaluation of F on $x \in \{0,1\}^n$ proceeds as follows,

$$y = \mathsf{F}^{\mathsf{H}}((s,k),x) = \mathsf{G}(s,x,z) \;\; \text{where } z = \mathsf{H}(k,\mathsf{C}(s,x)) \; . \tag{2.1}$$

Remark 1 *Note that the function families* C *and* G *in Equation* (2.1) *share the same seed* s*. This, in fact, is a generalization of the case when* C *and* G *have independent seeds* s_1 *and* s_2 *respectively – for every such* C *and* G *we can construct families* C' *and* G' *which share the same seed* $s=(s_1,s_2)$*,*

$$\mathsf{C}'(s=(s_1,s_2),\cdot) = \mathsf{C}(s_1,\cdot) \;\; ; \;\; \mathsf{G}'(s=(s_1,s_2),\cdot,\cdot) = \mathsf{G}(s_2,\cdot,\cdot) \; .$$

Furthermore, as G *and* C *share the same seed* s*,* G *can compute* $w=\mathsf{C}(s,x)$ *from its inputs* (s,x,z) *and hence w.l.o.g. we do not feed* w *as an input to* G*.*

Remark 2 *The choice of the input length of* C *is arbitrary as any* C *mapping* $l=\omega(\log n)$*-bit strings to* n*-bit strings can be converted into* C' *which maps* n*-bit strings*

to n-bit strings by padding 0^{l-n} to the input whenever $l \geq n$ and pre-processing the input via a universal hash family from n bits to l bits whenever $l < n$. Furthermore, the resulting family C' is c-universal hash family whenever C is c-universal for any $c \geq 2$.

The construction in Equation (2.1) covers all one-call constructions which do not modify the key of the naPRF. In particular, it also covers the Berman-Haitner [7] construction – one recovers the BH construction from $\mathsf{F}[\mathsf{C}, \mathsf{G}]$ by letting C be a universal hash family and letting $\mathsf{G}(s, (x, z)) = z$.

2.3.2 Main Theorem

Below we state our main theorem which provides an oracle relative to which a naPRF H exists but the construction $\mathsf{F}^{\mathsf{H}}[\mathsf{C}, \mathsf{G}]$ is not a PRF as long as C is universal function family. This in turn implies that F cannot be a fully black-box construction of a PRF from a naPRF.

Theorem 2 (Main Theorem) *Let $c = O(1)$ and r, σ, m be any polynomially bounded functions such that $m \geq \log(8ce)$. Let C be a c-universal family from n bits to n bits. Then, for every $\mathsf{F}^{(\cdot)}[\mathsf{C}, \mathsf{G}]$ (as in Equation 2.1) from n bits to m bits there exists a randomized oracle (O, R) such that,*[3]

1. *There exists an oracle function family $\mathsf{H}^{(\cdot)}$ from n bits to r bits with n-bit keys that is a naPRF relative to (O, R).*

2. *$\mathsf{F}^{\mathsf{H}}[\mathsf{C}, \mathsf{G}]$ is not a PRF relative to (O, R).*

Removing the c-universality assumption. Theorem 25 holds for every constant c, allowing us to show black-box separations for increasingly weaker assumptions on C.

[3]For simplicity we present our oracle as a pair (O, R).

However, to completely resolve the question, one would wish to remove the assumption altogether. This is far from simple: The naive approach is to argue that likely collisions in the non-universal family C can be turned into a distinguishing attack on F. But it is not clear how to argue this generically as the post-processing family G can potentially resolve collisions in C.

Nevertheless, we can remove the c-universality assumption on C altogether for two important subclasses of $\mathsf{F}[\mathsf{C},\mathsf{G}]$: (1) A special case of $\mathsf{F}[\mathsf{C},g]$ (second row in Table 2.1), where G consists of a single function g (i.e., independent of any seed material) and (2) A special case of $\mathsf{F}[\mathsf{C}]$ (third row in Table 2.1) where G is a family that on input (s,x,z) just outputs z. At a very high level, note that for the construction $\mathsf{F}[\mathsf{C}]$, collisions in C lead to collisions in F, however such collisions occur for a random function only with negligible probability when the output length satisfies $m = \omega(\log n)$. Therefore, an adversary that knows collisions in C can trivially break the PRF security of F. For the construction $\mathsf{F}[\mathsf{C},g]$ one needs to go a step further and analyze the entropy of the output $(\mathsf{F}(x_1),\ldots,\mathsf{F}(x_c))$ for inputs x_i's for which collisions under C are likely. We can show a distinguishing attack whenever $m = \Omega(n)$. We defer the formal proofs to the full version.

Overall, we believe removing c-universality from Theorem 25 for all one-call constructions is closely related to the long-standing open problem in symmetric-key cryptography of proving security beyond the birthday barrier for the composition of a non-universal hash family and a short-output PRF. The challenge is that collisions in the hash function may still be less likely than actual output collisions when the range is small. We believe removing the c-universality assumption is unlikely to happen without making progress on this open question, and we believe that the answer depends on a more fine-grained understanding of the combinatorics of C.

2.3.3 Proof Overview of Theorem 25

We prove Theorem 25 in two parts: First, in Proposition 2 we provide an oracle for constructions $\mathsf{F}[\mathsf{C},\mathsf{G}]$ that satisfies a structural property – "unbiasedness" (define next) and provide an oracle for "biased" constructions in Proposition 3.

$(1-\delta)$-unbiased $\mathsf{F}^{(\cdot)}[\mathsf{C},\mathsf{G}]$. Before we formally define the structural property of "$(1-\delta)$-unbiasedness" of F it would be helpful to consider the following definition.

Definition 3 *For the function family G, for some $n \in \mathbb{N}$, let $x \in \{0,1\}^n$, $s \in \{0,1\}^{\sigma(n)}$ and $y \in \{0,1\}^{m(n)}$, we say that y is $1/2$-bad w.r.t. (s,x) if $\Pr_z[y = \mathsf{G}(s,x,z)] > 1/2$, otherwise y is $1/2$-good w.r.t. (s,x).*

If for some pair (s,x) there exists a $1/2$-bad y then the output of F (on input x and seed s) will be biased towards y even if H is a truly random function family. We call F as unbiased if at least $(1-\delta)$ fraction of the outputs y's will be $1/2$-good for some $\delta < 1$.

Definition 4 ($(1-\delta)$-unbiased) *For any functions r, m, σ, let C be a family from n bits to n bits and G be a family from $n+r$ bits to m bits. Then for $\delta \leq 1$ we say that $\mathsf{F}^{(\cdot)}[\mathsf{C},\mathsf{G}]$ is $(1-\delta)$-unbiased if for all polynomials $l = \omega(\sigma)/\delta$ there exists some negligible function $\nu(\cdot)$ such that*

$$\Pr_{X,s,f}[|\{i : Y[i] \text{ is } 1/2\text{-bad w.r.t. } (s, X[i])\}| \geq \delta \cdot l] \leq \nu(n) ,$$

for every $n \in \mathbb{N}$ where $Y[i] = \mathsf{F}^f(s, X[i])$, $s \xleftarrow{\$} \{0,1\}^{\sigma(n)}$, $f \xleftarrow{\$} \mathsf{Funcs}(n, r(n))$ and $X \xleftarrow{\$} (\{0,1\}^n)^{[l(n)]}$. Otherwise, we call $\mathsf{F}^{(\cdot)}[\mathsf{C},\mathsf{G}]$ as δ-biased.

We state Proposition 2 (proof in Section 2.4) which handles unbiased F's.

Proposition 2 *Let $c = O(1)$ and r, m, σ be any polynomially bounded functions. Let C be a c-universal family from n bits to n bits with σ-bit seeds and G be a family from $n+r$*

bits to m bits with σ-bit seeds such that $\mathsf{F}^{(\cdot)}[\mathsf{C},\mathsf{G}]$ is $\left(1-\frac{1}{4c}\right)$-unbiased. Then, there exists a randomized oracle (O,R) such that there exists an oracle function family $\mathsf{H}^{(\cdot)}$ from n bits to r bits with n-bit keys that is a naPRF relative to (O,R) but $\mathsf{F}^{\mathsf{H}}[\mathsf{C},\mathsf{G}]$ is not a PRF relative to (O,R).

Next, we state our Proposition 3 (proof in Section 2.6) which relies on F being biased.

Proposition 3 *Let $c = O(1)$ and let r, σ, m be polynomially bounded functions such that $m \geq \log(2ce)$. For every $\mathsf{F}^{(\cdot)}[\mathsf{C},\mathsf{G}]$ from n bits to m bits, if $\mathsf{F}^{(\cdot)}[\mathsf{C},\mathsf{G}]$ is $1/c$-biased then there exists a randomized oracle (O,R) such that there exists an oracle function family $\mathsf{H}^{(\cdot)}$ from n bits to r bits with n-bit keys that is a naPRF relative to (O,R) but $\mathsf{F}^{\mathsf{H}}[\mathsf{C},\mathsf{G}]$ is not a PRF relative to (O,R).*

Remark 3 *Note that Proposition 2 rules out F for any output length m (even $m = 1$). However, we can only prove Proposition 3 when $m \geq \log(2ce)$ for some constant c. For this reason Theorem 25 requires $m \geq \log(2ce)$. It is an important open question to extend our results for smaller m's.*

Proof of Theorem 25. Given Propositions 2 and 3, Theorem 25 follows immediately by analyzing the following two cases: (1) If $\mathsf{F}^{(\cdot)}[\mathsf{C},\mathsf{G}]$ is $\frac{1}{4c}$-biased then Theorem 25 follows from Proposition 3 with parameter $4c$ (instead of c), and (2) If $\mathsf{F}^{(\cdot)}[\mathsf{C},\mathsf{G}]$ is $\left(1-\frac{1}{4c}\right)$-unbiased then Theorem 25 follows from Proposition 2.

2.4 The Case of Unbiased F's: Proof of Proposition 2

First, in Section 2.4.1 we establish some preliminary notation necessary to describe our oracles (O,R) and the naPRF family H (which are defined in Section 2.4.2). Then, in Section 2.4.3 we argue the insecurity of F relative to (O,R) and in Section 2.4.4 we argue that H is a naPRF relative to (O,R).

2.4.1 Preliminary Notation for Defining (O, R)

First, we observe an important property of c-universal function families called (β, δ)-sparseness.

Definition 5 (s is β-sparse) *Let C be a family from n bits to n bits with σ-bit seeds. For $n \in \mathbb{N}$, $\beta \leq 1$, we say that $s \in \{0,1\}^{\sigma(n)}$ is β-sparse if $\Pr_x[\mathsf{C}(s,x) = w] \leq \beta$ for every $w \in \{0,1\}^n$.*

Definition 6 (C is (β, δ)-sparse) *Let C be a function family from n bits to n bits. For functions β and δ we say that C is (β, δ)-sparse if $\mathsf{Pr}[s$ not $\beta(n)$-sparse $] \leq \delta(n)$ for all $n \in \mathbb{N}$ over the random choice of $s \xleftarrow{\$} \mathsf{C.Kg}$.*

Lemma 1 *For any $c = O(1)$, any (α, c)-universal function family C from n bits to n bits is also (β, δ)-sparse for $\beta = \max(\alpha^{1/2c}, \frac{2c}{2^n})$, $\delta = 2^{c-1}\sqrt{\alpha}$. Furthermore, β and δ are both negligible for $c = O(1)$ and negligible α.*

The following lemma show that a universal family is sparse (proof in Section 2.9.1).

For the rest of this section let us fix some (α, c)-universal function family C from n bits to n bits with σ-bit seeds, some $n + r$ bit to m bit function family G with σ-bit seeds such that $\mathsf{F} = \mathsf{F}[\mathsf{C}, \mathsf{G}]$ is a $(1 - 1/4c)$-unbiased function family (as in the statement of Proposition 2). Furthermore, for (α, c) let β, δ be functions (as defined by Lemma 1) such that C is (β, δ)-sparse. For C and F we define two sets of "good" seeds namely Good_C and Good_F necessary to describe (O, R).

The set $\mathsf{Good}_\mathsf{C}(\beta, X)$. For some tuple $X \in (\{0,1\}^n)^l$, the set $\mathsf{Good}_\mathsf{C}(\beta, X)$ is a set of β-sparse seeds for which there are no c-way collisions among $\mathsf{C}(s, X[i])$'s. To match the usage of Good_C later in the proof, we define $\mathsf{Good}_\mathsf{C}(\beta, X)$ for $X = X_1 || X_2$ where each X_i's are l-length tuples.

Definition 7 *For $n, l \in \mathbb{N}, \beta \leq 1$, $X = X_1||X_2 \in (\{0,1\}^n)^{[2l]}$ where $X_i \in (\{0,1\}^n)^{[l]}$, let $\mathsf{Good}_{\mathsf{C}}(\beta, X)$ denote the set of all $s \in \{0,1\}^\sigma$ such that s is β-sparse and there are no c-way collisions in $\mathsf{C}(s, X)$ – for every $I \subseteq [2l]$ fof size c there exists $i, j \in I$ such that $\mathsf{C}(s, X[i]) \neq \mathsf{C}(s, X[j])$ where we are viewing $\mathsf{C}(s, X)$ as a set instead of the tuples.*

The set $\mathsf{Good}_{\mathsf{F}}(\beta, X, Y)$. Here we extend the definition of "good" seeds relative to the outputs Y. Recall that F is $(1 - 1/4c)$-unbiased and so for some sufficiently large l we expect at most $1/4c$ fraction of the $Y[i]$'s to be $1/2$-bad. $\mathsf{Good}_{\mathsf{F}}$ is the set of seeds that are in $\mathsf{Good}_{\mathsf{C}}$ for which $1/4c$ fraction of the $Y[i]$'s are $1/2$-bad.

Definition 8 *For X as in Def. 7 and $Y \in (\{0,1\}^n)^{2l}$, let $\mathsf{Good}_{\mathsf{F}}(\beta, X, Y)$ denote the set of seeds $s \in \{0,1\}^\sigma$ such that $s \in \mathsf{Good}_{\mathsf{C}}(\beta, X)$ and*
$|\{i : Y[i]$ is $1/2$-bad w.r.t. $(s, X[i])\}| \leq 2l/4c = l/2c$.[4]

2.4.2 Oracles (O, R) and H^{O}

Recall that we are designing (O, R) for constructions $\mathsf{F}^{(\cdot)}[\mathsf{C}, \mathsf{G}]$ where C is (α, c)-universal and also (β, δ)-sparse (as observed in Lemma 1), and $\mathsf{F}^{(\cdot)}[\mathsf{C}, \mathsf{G}]$ is $(1 - 1/4c)$-unbiased for some $c = O(1)$ and negligible α. Let us, furthermore, fix some sufficiently large $l = \omega(\sigma + n)$. Next, we describe our oracles (O, R) which will depend on the families $\mathsf{C}, \mathsf{G}, \mathsf{F}$ and parameters β, c, l.

Oracle O and H^{O}. Oracle O embeds an information theoretically secure PRF. That is, for every $n \in \mathbb{N}$ and every $k \in \{0,1\}^n$, $\mathsf{O}(1^n, k, \cdot)$ is implemented by a function from $\mathsf{Funcs}(n, r)$ which is sampled uniformly and independently at random with replacement. Relative to such an oracle there exists a naPRF H^{O} from n bits to r bits with n-bit keys.

[4]Note that we have $2l/4c$ because our X, Y are tuples of $2l$ length tuples.

Oracle $\mathsf{R}_1(1^n)$:

if $T_1^n = \perp$ **then** $T_1^n \xleftarrow{\$} (\{0,1\}^n)^{[l(n)]}$

return T_1^n

Oracle $\mathsf{R}_2(1^n, X, Y)$:

if $\neg\mathsf{isValid}(1^n, l, X, Y)$ **then return** 1

if $X \neq T_1^n$ **then return** $\perp$

if $T_2^n[Y] = \perp$ **then** $T_2^n[Y] \xleftarrow{\$} (\{0,1\}^n \setminus T_1^n)^{[l(n)]}$

return $T_2^n[Y]$

Oracle $\mathsf{R}_3(1^n, X = X_1||X_2, Y = Y_1||Y_2)$:

if $\neg\mathsf{isValid}(1^n, 2l, X, Y)$ **then return** $\perp$

if $X_1 \neq T_1^n \vee X_2 \neq T_2^n[Y_1]$ **then return** $\perp$

if $\exists(s,k) \in \mathsf{Good}_{\mathsf{F}}(\beta, X, Y) \times \{0,1\}^n$:

$\mathsf{F}^{\mathsf{H}^{\mathsf{O}}}[\mathsf{C},\mathsf{G}]((s,k), X) = Y$ **then return** 1

return $\perp$

Proc. $\mathsf{isValid}(1^n, t, X, Y)$

if $X \notin (\{0,1\}^n)^{[t]}$ **then return** 0

if $Y \notin (\{0,1\}^m)^t$ **then return** 0

return 1

Adversary $A^{(\mathsf{O},\mathsf{R}),f}(1^n)$:

$X_1 \leftarrow \mathsf{R}_1(1^n)$

$Y_1 \leftarrow f(X_1)$

$X_2 \leftarrow \mathsf{R}_2(1^n, X_1, Y_1)$

$Y_2 \leftarrow f(X_2)$

if $\mathsf{R}_3(1^n, X_1||X_2, Y_1||Y_2) = 1$ **then**

return 1

return 0

Figure 2.2: Description of oracle R and adaptive adversary A that breaks the security of F relative to (O, R).

$\mathsf{H.Kg}(1^n)$ returns a randomly chosen key $k \in \{0,1\}^n$ and $\mathsf{H}^{\mathsf{O}}(k,x) = \mathsf{O}(k,x)$ for every key $k \in \{0,1\}^n$ and input x.

Oracle R. We decompose R into three oracles $(\mathsf{R}_1, \mathsf{R}_2, \mathsf{R}_3)$ as described in Fig 2.2.

Oracle R_1: Oracle R_1 for every $n \in \mathbb{N}$ samples an $l(n)$ length tuple T_1^n of n-bit strings *without* replacement. It accepts as input the security parameter 1^n and outputs the corresponding T_1^n.

Oracle R_2: Oracle R_2 works identically to the oracle R_1 except that it takes as inputs the security parameter 1^n, and tuples $X \in (\{0,1\}^n)^{[l(n)]}$ and $Y \in (\{0,1\}^n)^{l(n)}$ and returns a random $l(n)$-length tuple of n-bit strings ($T_2^n[Y]$ in Figure 2.2) iff $X = T_1^n$. The tuple $T_2^n[Y]$ is sampled without replacement from $\{0,1\}^n \setminus T_1^n$. We should think of R_1 as providing the first challenge tuple $X_1 = T_1^n$ and R_2 as providing the second, "adaptive" challenge tuple $X_2 = T_2^n[Y_1]$ after receiving the response Y_1 for the first challenge X_1.

Oracle R_3: R_1 and R_2 are just fancy random string generators and provide no way to break the security of F as both these oracles are in fact independent of F. The responsibility of ensuring that one can break F is on R_3. More precisely, R_3 accepts as queries a tuple $(X = X_1||X_2, Y = Y_1||Y_2)$ outputs 1 iff it finds some key (s,k) for F which maps X to Y where $k \in \{0,1\}^n$ and $s \in \mathsf{Good}_{\mathsf{F}}(\beta, X, Y)$ and it is also required that $X_1 = \mathsf{R}_1(1^n) = T_1^n$ and $X_2 = \mathsf{R}_2(1^n, X_1, Y_1) = T_2^n[Y_1]$.

This completes the description of R. Note that R depends on the entire description of oracle O in addition to the function families C and G and the parameters l, β. For notational convenience, we will drop the superscript n from T_i^n and the input 1^n from all oracles. Next, in Section 2.4.3 we establish the insecurity of F as a PRF and in Section 2.4.4 the security of H as a naPRF relative to (O, R) which put together will conclude the proof of Proposition 2.

2.4.3 F is not a PRF relative to (O, R)

Relative to the oracle (O, R) there exists a trivial uniform adversary $A^{(\mathsf{O},\mathsf{R}),f}$ which uses adaptive access to the challenge oracle f to compute $Y_i = f(X_i)$ for X_1, X_2 provided by R. In Lemma 2 we show that A indeed breaks the PRF security of F. The proof is detailed in Section 2.9.2.

Lemma 2 (F is insecure relative to (O, R)) *For A described in Figure 2.2 there ex-*

ists a non-negligible function ε such that, $\mathsf{Adv}^{\mathsf{rel-prf}}_{A,\mathsf{F},(\mathsf{O},\mathsf{R})}(n) \geq \varepsilon(n)$.

2.4.4 H is a naPRF relative to (O,R)

In this section, we establish the non-adaptive security of H relative to (O,R) by reducing it to the non-adaptive security of H relative to *only* O. That is, for every A relative to (O,R) making only non-adaptive queries to its challenge oracle f but adaptive queries to O and R, we construct an adversary B relative to *only* O that also only makes non-adaptive queries to its challenge oracle f and is just as successful as A in the non-adaptive security game of H. The adversary $B^{\mathsf{O},f}$ internally runs $A^{(\mathsf{O},\mathsf{R}),f}$ and answers all of its queries to O and f by forwarding to its own oracles. For the queries to R, B attempts to simulate the oracle R internally for A. Recall that R is decomposed into three oracles $(\mathsf{R}_1,\mathsf{R}_2,\mathsf{R}_3)$ where R_1 and R_2 just output random l-length tuples of n-bit strings and hence are easy to simulate. The challenge is to simulate the oracle R_3, which depends on the entire description of O, with only oracle access to O. Nevertheless, we show that B can still simulate R_3 queries correctly. We emphasize that the non-adaptive query restriction on A is only w.r.t. querying f. It can query (O,R) adaptively.

Lemma 3 (H is a naPRF relative to (O,R)) *For any non-adaptive adversary A that makes at most $q \leq 2^{n/2}$ to its oracles we have for every $n \in \mathbb{N}$,* $\mathsf{Adv}^{\mathsf{rel-naprf}}_{A,\mathsf{H},(\mathsf{O},\mathsf{R})}(n) \leq 2q\cdot\varepsilon + \frac{2q}{2^n}$ *where*

$$\varepsilon = \frac{(q+1)2^{\sigma}}{2^t} + \frac{2^{\sigma+n}}{2^{t(c+1)}} + \frac{6q}{2^n} + q\ 2^{\sigma}\ \binom{l}{l/2}\ (2\beta q)^{l/2}\ ;\ t = \frac{l}{2c(c-1)}\ .$$

Note that since $l = \omega(\sigma+n)$ and β is negligible, the advantage of A for any polynomial q is negligible. This, with Lemma 2 concludes Proposition 2's proof.

Remark 4 *Although for concreteness we state A's advantage for $q \leq 2^{n/2}$, note that for the advantage to be negligible we require $q < 1/2\beta$. Therefore, we can only prove non-adaptive security of H (in an asymptotic sense) only when $q < 1/2\beta$. We note that an adversary A making $q \geq 1/\beta$ queries can, indeed, break the non-adaptive security of H. This is because, the range of function $\mathsf{C}(s,\cdot)$ for any β-sparse s has at least $\frac{1}{\beta}$ elements and hence an A can just query the challenge oracle f on the entire range of $\mathsf{C}(s,\cdot)$ and force R_3 to return 1 when $f \xleftarrow{\$} \mathsf{H}$(this is the same attack as described in Section 2.1). This is how we avoid the 1-call non-security preserving proof of Berman and Haitner [7]. More precisely, they establish PRF security of their construction assuming the naPRF is secure against $q = \beta^{-1}$-queries for some negligible β.*

Proof of Lemma 3. Fix some computationally unbounded adversary A making q queries and also some $n \in \mathbb{N}$. Let us assume w.l.o.g. that A makes q distinct queries to its oracles and is deterministic. We will proceed via a sequence of games and then appropriately describe the adversary B relative to O.

Game G_0 is identical to the real-world of the non-adaptive game for H except that G_0 maintains a set Q of all keys k for which A had issued an O-query on (k,x) for some x. The code for G_0 is shown in Figure 2.3. This is just a syntactic change, therefore $\mathsf{Pr}[\mathsf{G}_0] = \mathsf{Pr}_{\mathsf{O},\mathsf{R},f\xleftarrow{\$}\mathsf{F}}[A^{(\mathsf{O},\mathsf{R}),f} = 1]$.

Recall that any R_3 query $(X = X_1||X_2, Y = Y_1||Y_2)$ in G_0 returns 1 iff it finds a key (s,k) for F such that $\mathsf{F}((s,k),X) = Y$ where $k \in \{0,1\}^n$ and $s \in \mathsf{Good}_{\mathsf{F}}(\beta, X, Y)$. Such an R_3 seems too generous in providing help to A. This is because it also considers k's for which A has not made an $(k,\cdot)$ query to O, or equivalently $k \notin Q$, to determine its answer. Since for each k, O_k (implemented by the function π_k in Fig 2.3) behaves as a random function independent of other k''s it is unlikely that A has any information about O_k for any $k \notin Q$. Hence A's queries to R_3 should only depend on $k \in Q$. Carrying this intuition

Game $\mathsf{G}_0, \mathsf{G}_1$:

foreach $k \in \{0,1\}^n$ **do**

$\pi_k \xleftarrow{\$} \mathsf{Funcs}(n,m)$

$k^* \xleftarrow{\$} \{0,1\}^n$

$b \xleftarrow{\$} A^{(\mathsf{O},\mathsf{R}),f}$

return b

Oracle $\mathsf{R}_3(X = X_1||X_2, Y = Y_1||Y_2)$: //Game G_0

if $\neg\mathsf{isValid}(2l, X, Y)$ **then return** $\perp$

if $X_1 \neq T_1 \vee X_2 \neq T_2[Y_1]$ **then return** $\perp$

if $\exists (s,k) \in \mathsf{Good}_{\mathsf{F}}(\beta, X, Y) \times \{0,1\}^n$:

$\mathsf{F}^{\mathsf{H}}[\mathsf{C},\mathsf{G}]((s,k),X) = Y$ **then return** 1

return $\perp$

Oracle $\mathsf{R}_3(X = X_1||X_2, Y = Y_1||Y_2)$: //Game G_1

if $\neg\mathsf{isValid}(2l, X, Y)$ **then return** $\perp$

if $X_1 \neq T_1 \vee X_2 \neq T_2[Y_1]$ **then return** $\perp$

if $\exists (s,k) \in \mathsf{Good}_{\mathsf{F}}(\beta, X, Y) \times Q$:

$\mathsf{F}^{\mathsf{H}}[\mathsf{C},\mathsf{G}]((s,k),X) = Y$ **then return** 1

return $\perp$

Oracle $\mathsf{R}_1()$:

if $T_1 = \perp$ **then** $T_1 \xleftarrow{\$} (\{0,1\}^n)^{[l]}$

return T_1

Oracle $\mathsf{R}_2(X, Y)$:

if $\neg\mathsf{isValid}(1^n, l, X, Y)$ **then return** 1

if $X \neq T_1$ **then return** $\perp$

if $T_2[Y] = \perp$ **then**

$T_2[Y] \xleftarrow{\$} (\{0,1\}^n \setminus T_1)^{[l(n)]}$

return $T_2[Y]$

Oracle $\mathsf{O}(k, x)$:

$Q \leftarrow Q \cup \{k\}$

return $\pi_k(x)$

Oracle $f(x)$:

$y \leftarrow \pi_{k^*}(x)$

return y

Figure 2.3: Games G_0 and G_1 used in the proof of naPRF security of H relative to (O,R). The only difference is the implementation of the R_3 oracle – in G_0 the R_3 oracle while answering its queries considers all $k \in \{0,1\}^n$ while in G_1 it only considers $k \in Q$. The $\mathsf{isValid}$ procedure (omitted here) is as described in Fig. 2.2.

we move to the game G_1 where R_3 only considers $k \in Q$ as opposed to $k \in \{0,1\}^n$.

Games G_0 and G_1 are close: To give an intuition of why G_0 and G_1 are close, let us assume that A only makes one R_3 query and furthermore is its last query. It is

easy to see that the games remain identical until the R_3 query. Let the query be on some $(X = X_1||X_2, Y = Y_1||Y_2)$. Furthermore, let us assume that $X_1 = \mathsf{R}_1(1^n)$ and $X_2 = \mathsf{R}_2(X_1, Y_1)$ as otherwise R_3 outputs $\perp$ in both G_0 and G_1, hence identical responses. Now, the output of the R_3 query in G_0 differs from that in G_1 if there exists some $k \notin Q$ and some $s \in \mathsf{Good}_{\mathsf{F}}(\beta, X, Y)$ such that $\mathsf{F}((s,k), X) = Y$. Fix one such $k \notin Q$ and some $s \in \mathsf{Good}_{\mathsf{F}}(\beta, X, Y)$. The probability that R_3 in G_1 errs by ignoring this (s,k) can be upper bounded by the probability that over the choice of O_k (a random function) that for each $i \in [l]$, we have $Y_1[i] = \mathsf{F}^{\mathsf{O}_k}(s, X_1[i])$. Let $W_1[i] = \mathsf{C}(s, X_1[i])$ for all $i \in [l]$. Since, $s \in \mathsf{Good}_{\mathsf{F}}(\beta, X, Y)$ (Definition 8) we know that there exists at least $l/(c-1)$ distinct $W_1[i]$'s as $|\mathsf{C}(s, X_1)| > l/(c-1)$. Furthermore, we know that for $Y = Y_1||Y_2$ at most $l/2c$ of the $Y[i]$'s are 1/2-bad w.r.t. $(s, X[i])$. Therefore, we can safely conclude that there exists a subset $I_s \subseteq [l]$ of size at least $l/(c-1) - l/2c$ such that for every $i \neq j \in I_s$, we have

1. $W_1[i] \neq W_1[j]$, where recall that $W_1 = \mathsf{C}(s, X_1)$
2. $Y_1[i]$ is 1/2-good w.r.t. $(s, X_1[i])$.[5]

Furthermore, none of the $W_1[i]$ have been queried before and hence $Z_1[i] = \mathsf{O}(k, W_1[i])$ are random independent strings. Therefore, the probability that

$$\Pr_{\mathsf{O}_k}[\forall i \in I_s : \mathsf{F}^{\mathsf{O}_k}(s, X_1[i]) = Y_1[i]] \leq \Pr_{\mathsf{O}_k}[\forall i \in I_s : \mathsf{G}(s, X_1[i], Z_1[i]) = Y_1[i]] \leq \frac{1}{2^t} ,$$

where $t = |I_s| \geq \frac{(c+1)\cdot l}{2c(c-1)} = \Omega(l)$ as $c = O(1)$. Taking the union bound over all σ-bit s's and n-bit k's we can show that the probability that G_1 errs on the first R_3 query is negligible. In other words, G_1 can safely ignore $k \notin Q$ and this is because if for some X and Y and some s if A has not already determined that $\mathsf{F}((s,k), X) = Y$ then the

[5]Recall that y is 1/2-good w.r.t. (s, x) if $\Pr_z[\mathsf{G}(s, x, z) = y] \leq 1/2$.

probability that it is indeed the case is small. We will use this fact a number of times in the proof. Let us make this formal.

Definition 9 *We say that a set $Q \subseteq \{0,1\}^n \times \{0,1\}^m$ is a (n,m)-query-set if for every w there exists at most one y such that $(w,y) \in Q$. Furthermore, let $\mathsf{Query}(Q)$ define the set of queries, that is, $\mathsf{Query}(Q) = \{w : \exists y \text{ s.t. } (w,y) \in Q\}$.*

Lemma 4 *For $n, l, t \in \mathbb{N}$, consider $X \in (\{0,1\}^n)^{[l]}$, $Y \in (\{0,1\}^m)^l$ and let $Q \subseteq \{0,1\}^n \times \{0,1\}^m$ be an (n,m)-query set. Let s be such that there exists $I_s \subseteq [l]$ of size t such that $\forall i \neq j \in I_s$ the following holds: (1) $\mathsf{C}(s, X[i]) \neq \mathsf{C}(s, X[j])$, (2) $\mathsf{C}(s, X[i]) \notin \mathsf{Query}(Q)$, and (3) $Y[i]$ are $1/2$-good w.r.t. $(s, X[i])$. Then,*

$$\Pr_{g \xleftarrow{\$} \mathsf{Funcs}(n,m)|Q}[\mathsf{F}^g(s,X) = Y] \leq 2^{-t} ,$$

where $g \xleftarrow{\$} \mathsf{Funcs}(n,m)|Q$ is the process of sampling a function uniformly at random from $\mathsf{Funcs}(n,m)$ such that for every $(w,y) \in Q$ we have $g(w) = y$.

But is bounding the above probability for $k \notin Q$ enough to show that G_0 and G_1 close? Recall that A has access to the oracle f which internally calls O_{k^*} (where k^* is the random key sampled by the games to implement f. That is, $f = \mathsf{O}_{k^*}$) . It could very well be that $k^* \notin Q$ but that hardly ensures that A has made no queries to O on $(k^*, \cdot)$. In fact if A manages to find some s such that $\mathsf{F}^f(s,X) = Y$ then R_3 queries answered in G_1 are necessarily incorrect. For this to happen, A needs to find some s such that $\mathsf{F}^f(s,X_1) = Y_1$ and $\mathsf{F}^f(s,X_2) = Y_2$. This is where the iterative nature of $\mathsf{R}_1, \mathsf{R}_2$ is supremely crucial which ensures that A learns X_2 after committing to Y_1 (i.e., after querying R_2 on (X_1, Y_1)). Since A only makes non-adaptive queries to f it is either in one of the following cases: (1) Issues all f queries after committing to Y_1 or (2) Issues all f queries before learning X_2. In (1) A succeeds only if the challenge oracle f agrees with

Y_1 on all of X_1 (for some $s \in \mathsf{Good}_{\mathsf{C}}(\beta, X)$) which is unlikely by a discussion we made in the context of handling $k \notin Q$ and in (2) A succeeds only if $\mathsf{C}(s, X_2)$ falls inside the set of f queries it had issued. Fortunately, such an event is also unlikely for s that is β-sparse and randomly sampled X_2. In both cases, for every $s \in \mathsf{Good}_{\mathsf{C}}(\beta, X)$ the conditions of Lemma 4 are satisfied for some $t = \Theta(l)$. The full proof is in Section 2.5.

Lemma 5 *For $t = l/(2c(c-1))$,*

$$|\mathsf{Pr}[\mathsf{G}_0] - \mathsf{Pr}[\mathsf{G}_1]| < q \cdot \left(\frac{(q+1)2^{\sigma}}{2^t} + \frac{2^{\sigma+n}}{2^{t(c+1)}} + \frac{6q}{2^n} + q\ 2^{\sigma} \binom{l}{l/2} (2\beta q)^{l/2} \right) .$$

Next, we consider a similar transition from the game H_0 (identical to the random world of the naPRF security game of H) to a game H_1 where R_3 queries are answered only by considering $k \in Q$ as done in G_1. By a similar analysis(more details in Section 2.5.3),

Lemma 6 *For $t = l/(2c(c-1))$,*

$$|\mathsf{Pr}[\mathsf{H}_0] - \mathsf{Pr}[\mathsf{H}_1]| < q \cdot \left(\frac{(q+1)2^{\sigma}}{2^t} + \frac{2^{\sigma+n}}{2^{t(c+1)}} + \frac{6q}{2^n} + q\ 2^{\sigma} \binom{l}{l/2} (2\beta q)^{l/2} \right) .$$

Now, we are set to describe our adversary B relative to O. Note that in both G_1 and H_1 the R_3 queries only depend on $k \in Q$. Consider the following adversary B which is relative to O and has non-adaptive access to the challenge oracle f. It internally runs A and answers its queries to O and f by forwarding them to its own oracles. It internally simulates R_1 and R_2 and to simulate R_3 we allow B to learn the entire description of O_k whenever the first query to O_k is made by A. Such a B can then perfectly simulate the game G_1 (resp., H_1) for A. Therefore, we have argued that,

$$\mathsf{Pr}[\mathsf{G}_1] = \underset{\mathsf{O}, k^* \xleftarrow{\$} \{0,1\}^n}{\mathsf{Pr}} [B^{\mathsf{O}, f=\mathsf{H}^{\mathsf{O}}_{k^*}} = 1] \ ; \mathsf{Pr}[\mathsf{H}_1] = \underset{\mathsf{O}, f \xleftarrow{\$} \mathsf{Funcs}(n,m)}{\mathsf{Pr}} [B^{\mathsf{O}, f} = 1] \ . \tag{2.2}$$

The final step is to invoke the security of H relative to O. For this, we consider an extended version of the security game of H relative to O where for every (k, x) query to O instead of just getting $\mathsf{O}(k, x)$ the adversary B gets the entire description of O_k – we refer to such queries as "advanced" queries. Note that B makes exactly q "advanced" queries and also only makes non-adaptive queries to f. Then, we claim the following whose proof follows from standard techniques,

$$\left| \Pr_{\mathsf{O},k^* \xleftarrow{\$} \{0,1\}^n} [B^{\mathsf{O},f=\mathsf{H}^{\mathsf{O}}_{k^*}} = 1] - \Pr_{\mathsf{O},f \xleftarrow{\$} \mathsf{Funcs}(n,r)} [B^{\mathsf{O},f} = 1] \right| \leq \frac{2q}{2^n} \ . \tag{2.3}$$

Combining Lemmas 5, 6 with Equations 2.2,2.3 concludes the proof of Lemma 3. Next, we discuss the proofs of Lemma 5 and Lemma 6 in Section 2.5.

2.5 Proofs of Lemma 5 and Lemma 6

We will first focus on showing Lemma 5. The proof of Lemma 6 is similar and we discuss it in Section 2.5.3.

Proof of Lemma 5. To show the indistinguishability of G_0 and G_1 we consider for every $i \in \{0, \ldots, q\}$ an intermediate game G^i where any queries to R_3 within the first i queries are answered as in G_1 (i.e., by just considering $k \in Q$), while the rest of the R_3 queries are answered as done in G_0 (which consider all $k \in \{0,1\}^n$). Note that G^0 is identical to G_0 and G^q to G_1. Therefore,

$$|\Pr[\mathsf{G}_0] - \Pr[\mathsf{G}_1]| \leq \sum_{i \in \{0,\ldots,q-1\}} |\Pr[\mathsf{G}^i] - \Pr[\mathsf{G}^{i+1}]| \ . \tag{2.4}$$

Let us fix some $i \in \{0, \ldots, q-1\}$ and consider G^i and G^{i+1}. The first point of difference between G^i and G^{i+1} is the $i+1$-st query. Furthermore, if the $i+1$-st query is to any oracle other than R_3 then both games remain identical as queries to oracles other

than R_3 are handled identically throughout both games. Therefore, w.l.o.g. we assume that the $i+1$-st query in both games is to R_3. Given this, we introduce two games $\hat{\mathsf{G}}^i$ and $\hat{\mathsf{G}}^{i+1}$ for G^i and G^{i+1} respectively (Figure 2.4).

Description of Game $\hat{\mathsf{G}}^i$: Game $\hat{\mathsf{G}}^i$ is identical to the game G^i except that the oracles O and f are implemented via lazy sampling until the $i+1$-st query.

More precisely, for the first i queries: (1) For any query to O on (k, x) which is the first query to O_k (i.e., $k \notin Q$), a random function is sampled from $\mathsf{Funcs}(n, m)$ and assigned to π_k. The game also inserts k in the set Q. The response for this query and any future query x on O_k is replied with $\pi_k(x)$. (2) For any query to f on x, if $k^* \notin Q$ the response is a uniformly random value $y \xleftarrow{\$} \{0,1\}^m$ otherwise the response is $y = \pi_{k^*}(x)$.

The oracles f and O_{k^*} are correlated and hence the function π_{k^*} in (1) is sampled to be consistent with the set Q_f. We denote this by $\pi_{k^*} \xleftarrow{\$} \mathsf{Funcs}(n, m)|Q_f$ in Figure 2.4. Furthermore, the game maintains the queries/responses to O_{k^*} in the set Q_{k^*} and queries/responses to f in a different set Q_f.

By the assumption on A's behavior, we know that the $i+1$-st query is to R_3. Since this query to R_3 (in G^i) depends on all $k \in \{0,1\}^n$ (even the ones not in the set Q) the game at the beginning of this call to R_3 completes the description of the oracles O and f. That is, it first samples functions π_k for all $k \notin Q \cup \{k^*\}$ inside the subroutine $\mathsf{CompleteO}$ and completes the description of f (equivalently, O_{k^*}) inside $\mathsf{Completef}$. Now, the response for this $i+1$-st query is 1 if there exists some $k \in \{0,1\}^n$ and some $s \in \mathsf{Good}_{\mathsf{F}}(\beta, X, Y)$ such that $\mathsf{F}((s,k), X) = Y$. Otherwise, R_3 returns $\perp$. It is clear that this R_3 query is computed as in G^i. In the process, $\hat{\mathsf{G}}^i$ sets two bad flags bad_1 and bad_2 where bad_1 is set if there exists some $k \notin Q$ for which $\mathsf{F}((s,k), X) = Y$, and bad_2 is set if the same is true for $k = k^*$. The game $\hat{\mathsf{G}}^i$ is only syntactically different from G^i, therefore

$$\mathsf{Pr}[\mathsf{G}^i] = \mathsf{Pr}[\hat{\mathsf{G}}^i] \ . \tag{2.5}$$

Game $\boxed{\hat{\mathsf{G}}^i}, \hat{\mathsf{G}}^{i+1}$:

$\mathsf{bad}_1, \mathsf{bad}_2, \mathsf{done} \leftarrow \mathsf{false}; c \leftarrow 0$

$k^* \xleftarrow{\$} \{0,1\}^n$

$b \xleftarrow{\$} A^{\mathsf{O},\mathsf{R},f}$

return b

Proc. $\mathsf{R}_2(X, Y)$:

$c \leftarrow c + 1$

if $\neg\mathsf{isValid}(1^n, l, X, Y)$ **then return** 1

if $X \neq T_1$ **then return** $\perp$

if $T_2[Y] = \perp$ **then**

  $T_2[Y] \xleftarrow{\$} (\{0,1\}^n \setminus T_1)^{[l(n)]}$

return $T_2[Y]$

Proc. $\mathsf{O}(k, x)$:

$c \leftarrow c + 1$

if $\neg\mathsf{done} \wedge k \notin Q$ **then**

  if $k = k^*$ **then** $\pi_{k^*} \xleftarrow{\$} \mathsf{Funcs}(n, m)|Q_f$

  else $\pi_k \xleftarrow{\$} \mathsf{Funcs}(n, m)$

$Q \leftarrow Q \cup \{k\}$

return $\pi_k(x)$

Proc. $\mathsf{R}_1(1)$:

$c \leftarrow c + 1$

if $T_1 = \perp$ **then** $T_1 \xleftarrow{\$} (\{0,1\}^n)^{[l]}$

return T_1

Proc. $\mathsf{CompleteO}()$:

foreach $k \notin Q \cup \{k^*\}$ **do**

  $\pi_k \xleftarrow{\$} \mathsf{Funcs}(n, m)$

return 1

Proc. $\mathsf{Completef}()$:

if $k^* \notin Q$ **then** $\pi_{k^*} \xleftarrow{\$} \mathsf{Funcs}(n, m)|Q_f$

return 1

Proc. $f(x)$:

$c \leftarrow c + 1$

if $\neg\mathsf{done} \wedge k^* \notin Q$ **then** $y \xleftarrow{\$} \{0,1\}^m$

else $y \leftarrow \pi_{k^*}(x)$

$Q_f \leftarrow Q_f \cup \{(x, y)\}$

return y

Figure 2.4: Intermediate Games used in the proof of non-adaptive security of H relative to (O, R).

Proc. $\mathsf{R}_3(X = X_1||X_2, Y = Y_1||Y_2)$:

$c \leftarrow c + 1; b \leftarrow 0$

if $c \leq i$ **then**

  if $\neg\mathsf{isValid}(2l, X, Y) \vee X_1 \neq T_1 \vee X_2 \neq T_2[Y_1]$ **then return** $\perp$

  if $\exists (s,k) \in \mathsf{Good}_{\mathsf{F}}(\beta, X, Y) \times Q : \mathsf{F}((s,k), X) = Y$ **then return** 1

elseif $c = i + 1$ **then**

  if $\neg\mathsf{isValid}(2l, X, Y) \vee X_1 \neq T_1 \vee X_2 \neq T_2[Y_1]$ **then**

    $\mathsf{CompleteO}(); \mathsf{Completef}()$

    $\mathsf{done} \leftarrow \mathsf{true}$

    return $\perp$

  if $\exists (s,k) \in \mathsf{Good}_{\mathsf{F}}(\beta, X, Y) \times Q : \mathsf{F}((s,k), X) = Y$ **then** $b \leftarrow 1$

  $\mathsf{CompleteO}()$

  if $\exists (s,k) \in \mathsf{Good}_{\mathsf{F}}(\beta, X, Y) \times (Q^{\mathsf{c}} \setminus \{k^*\}) : \mathsf{F}((s,k), X) = Y$ **then**

    $\mathsf{bad}_1 \leftarrow \mathsf{true}; \boxed{b \leftarrow 1}$

  $\mathsf{Completef}()$

  if $\exists s \in \mathsf{Good}_{\mathsf{F}}(\beta, X, Y) : \mathsf{F}^f(s, X) = Y$ **then**

    $\mathsf{bad}_2 \leftarrow \mathsf{true}; \boxed{b \leftarrow 1}$

  $\mathsf{done} \leftarrow \mathsf{true}$

  if $b = 1$ **then return** 1

else

  if $\neg\mathsf{isValid}(2l, X, Y)$ **then return** $\perp$

  if $X_1 \neq T_1 \vee X_2 \neq T_2[Y_1]$ **then return** $\perp$

  if $\exists (s,k) \in \mathsf{Good}_{\mathsf{F}}(\beta, X, Y) \times \{0,1\}^n :$

    $\mathsf{F}((s,k), X) = Y$ **then return** 1

return $\perp$

Figure 2.4: (continued) Intermediate Games used in the proof of non-adaptive security of H relative to (O, R).

Description of Game $\hat{\mathsf{G}}^{i+1}$: Game $\hat{\mathsf{G}}^{i+1}$ is identical to that of $\hat{\mathsf{G}}^{i}$ except that in the $i+1$-st query, R_3 responds with 1 iff there exists some $k \in Q$ such that $\mathsf{F}((s,k),X) = Y$. This is identical to how this query to R_3 is handled in G^{i+1}. The game $\hat{\mathsf{G}}^{i+1}$ is also a syntactic variant of G^{i+1}, therefore

$$\mathsf{Pr}[\mathsf{G}^{i+1}] = \mathsf{Pr}[\hat{\mathsf{G}}^{i+1}] \ , \tag{2.6}$$

Furthermore, the games $\hat{\mathsf{G}}^{i}$ and $\hat{\mathsf{G}}^{i+1}$ are identical until either of the bad flags are set in $\hat{\mathsf{G}}^{i}$. By the fundamental lemma of game playing we have,

$$|\mathsf{Pr}[\hat{\mathsf{G}}^{i}] - \mathsf{Pr}[\hat{\mathsf{G}}^{i+1}]| \leq \mathsf{Pr}[\mathsf{bad} \text{ in } \hat{\mathsf{G}}^{i}] \ . \tag{2.7}$$

Next, we bound the probability of bad being set in $\hat{\mathsf{G}}^{i}$ in Lemma 7.

Lemma 7 *For every $i \in \{0,\ldots,q\}$,*

$$\mathsf{Pr}[\mathsf{bad} \textit{ is set in } \hat{\mathsf{G}}^{i}] \leq \frac{(q+1)2^{\sigma}}{2^{t}} + \frac{2^{\sigma+n}}{2^{t(c+1)}} + \frac{6q}{2^{n}} + q\ 2^{\sigma} \binom{l}{l/2} (2\beta q)^{l/2} \ .$$

Before we prove Lemma 7, we note that Lemma 5 follows directly by combining Lemma 7 and Equation (2.5), Equation (2.6) and Equation (2.7).

The rest of this section is devoted to proving Lemma 7: Let us fix some $i \in \{0,\ldots,q\}$. Note that bad is set in $\hat{\mathsf{G}}^{i}$ if either of bad_1 or bad_2 is set. While the analysis of bad_1 is straightforward, some care needs to be taken while bounding bad_2. In Section 2.5.1 we define some bad events on which we will condition on to bound the probability of setting $\mathsf{bad} = \mathsf{bad}_1 \vee \mathsf{bad}_2$ in Section 2.5.2.

2.5.1 Bad Events in $\hat{\mathsf{G}}^{i}$

In this section we define bad events and bound the probability of these events occurring in $\hat{\mathsf{G}}^{i}$.

The first event is BadO which captures the event that A makes a direct query to $\mathsf{O}(k^*, \cdot)$ within its first i queries. Conditioned on $\neg\mathsf{BadO}$ the queries/responses to f and O are independent.

Definition 10 *The event* BadO *occurs in* $\hat{\mathsf{G}}^i$ *if within the first* i *queries, there exists an* $\mathsf{O}(k, \cdot)$ *query such that* $f = \mathsf{O}_k$*. In other words,* BadO *occurs if there exists a* $\mathsf{O}(k^*, \cdot)$ *query within the first* i *queries.*

The second event is BadR which captures the event that after all parallel queries to f have been made, a future R_2 query results in an X_2 for which more than $l/2$ of the $\mathsf{C}(s, X_2[i])$'s fall inside the set queried Q_f, enabling A to compute the $\mathsf{F}^f(s, X_2[i])$.

Definition 11 *The event* BadR *occurs in* $\hat{\mathsf{G}}^i$ *if within the first* i *queries, immediately after an assignment* $T_2[Y_1] \xleftarrow{\$} (\{0,1\}^n \setminus T_1)^{[l]}$ *there exists some* s *such that the following holds for* $Q = Q_{k^*} \cup Q_f$*,*

1. s *is* β*-sparse.*
2. *there exists some* $I_s \subseteq [l]$ *of size at least* $l/2$ *such that for every* $i \in I_s$*,*

$$\mathsf{C}(s, T_2[Y_1][i]) \in \mathsf{Query}(Q) \ ,$$

where by $\mathsf{Query}(Q) = \{w : (w, y) \in Q\}$*.*

Informally, Badf captures the event that after all parallel queries to f have been made, for a prior R_2 query on (X_1, Y_1) there exists some seed s (e.g., for which no c-way collisions occur) for which more than $l/2$ of the $\mathsf{C}(s, X_1[i])$'s fall inside the query set Q_f and furthermore for all such i's we have $\mathsf{F}^f(s, X_1[i]) = Y_1[i]$. A direct consequence of showing that Badf doesn't occur allows us to focus on seeds s for which at least $l/2$ of the $\mathsf{C}(s, X_1[i])$'s do not belong to Q_f. This will be helpful in later bounding the probability of setting bad_2.

Definition 12 *The event* Badf *occurs in* $\hat{\mathsf{G}}^i$ *within the first* i *queries, if immediately after all parallel queries to* f *there exist some* Y_1 *such that* $T_2[Y_1] \neq \perp$ *and* s *such that the following hold for* $Q = Q_f \cup Q_{k^*}$,

1. *there are no* c*-way collisions in* $\mathsf{C}(s, T_1)$.[6]
2. *for* $I_s = \{i \in [l] : \mathsf{C}(s, X_1[i]) \in \mathsf{Query}(Q)\}$, *we have* $|I_s| \geq l/2$.
3. $|\{i : Y_1[i]$ *is* $1/2$*-bad w.r.t.* $(s, T_1[i])\}| \leq l/2c$. [7]
4. *for every* $i \in I_s$,

$$\mathsf{G}(s, T_1[i], z) = Y_1[i] \ , \tag{2.8}$$

where $(\mathsf{C}(s, T_1[i]), z) \in Q$.

Definition 13 *The event* Bad *happens in* $\hat{\mathsf{G}}^i$ *if the event* $\mathsf{BadO} \vee \mathsf{BadR} \vee \mathsf{Badf}$ *happens.*

Next, we bound the probability of the event Bad happening.

Claim 1

$$\mathsf{Pr}[\mathsf{Bad}] \leq \frac{6q}{2^n} + q \cdot \frac{2^\sigma}{2^t} + q \ 2^\sigma \binom{l}{l/2} (2\beta \cdot q)^{l/2} \quad \textit{where } t = \frac{l}{2c(c-1)} \ .$$

At a high level the proof of Claim 1 proceeds in two steps – (1) bounding the probability of each of the three bad events BadO, Badf and BadR happening, and (2) doing a simple union bound. The formal proof can be found in Section 2.5.4.

[6] Recall that R_3 ignores seeds s for which there are c-way collisions in $\mathsf{C}(s, T_1)$.
[7] Recall that R_3 ignores pairs $(s, Y = Y_1||\cdot)$ for which there are more than $l/2c$ bad $Y_1[i]$'s.

2.5.2 Bounding bad in $\hat{\mathsf{G}}^i$

In this section, we bound the probability of flags bad_1 and bad_2 getting set in the game $\hat{\mathsf{G}}^i$ (see Lemma 7). We will be conditioning on $\neg\mathsf{Bad}$ event (Definition 13). Recall that each of bad_1 and bad_2 are set only when the $i+1$-st query is made, furthermore this query is to R_3. Let us assume that this $i+1$-st query is on $(X = X_1||X_2, Y = Y_1||Y_2)$. If $\mathsf{isValid}(X_1, Y_1, X_2, Y_2)$ is false then conditioned on this the probability of setting bad is zero. So, w.l.o.g. assume that $\mathsf{isValid}(X_1, Y_1, X_2, Y_2)$ is indeed true. This means that $X_1 = T_1$ and $T_2[Y_1] \neq \perp$. In fact, $T_2[Y_1] = X_2$. Since, $T_2[Y_1] \neq \perp$, this means that prior to this query to R_3 there was a query to R_2 on (X_1, Y_1) which is when the value $T_2[Y_1]$ was defined. Let us assume that this was some j-th query where $j < i$. Using the above notation we next bound the probability of bad_1 being set.

Claim 2 $\mathsf{Pr}[\mathsf{bad}_1] \leq \frac{2^{\sigma+n}}{2^{t\cdot(c+1)}}$, *where* $t = \frac{l}{2c(c-1)}$.

Proof: By the description of Game $\hat{\mathsf{G}}^i$, we note that bad_1 is set only if for the query $(X = X_1||X_2, Y = Y_1||Y_2)$ to R_3, the set Q and k^*, there exists some $s \in \mathsf{Good}_{\mathsf{F}}(\beta, X, Y)$ and $k \in Q^{\mathsf{c}} \setminus \{k^*\}$, such that $\mathsf{F}((s,k), X) = Y$ where Q^{c} is the set complement of Q. That is,

$$\begin{aligned}\mathsf{Pr}[\mathsf{bad}_1] &\leq \mathsf{Pr}[\exists s \in \mathsf{Good}_{\mathsf{F}}(\beta, X, Y), k \in Q^{\mathsf{c}} \setminus \{k^*\} : \mathsf{F}((s,k), X) = Y] \\ &\leq \mathsf{Pr}[\exists s \in \mathsf{Good}_{\mathsf{F}}(\beta, X, Y), k \in Q^{\mathsf{c}} \setminus \{k^*\} : \mathsf{F}((s,k), X_1) = Y_1] \ ,\end{aligned} \tag{2.9}$$

where we use the fact that probability that $\mathsf{F}((s,k), X_1) = Y_1$ upperbounds the probability of $\mathsf{F}((s,k), X) = Y$.

Let us fix some such $k \in Q^{\mathsf{c}} \setminus \{k^*\}$ and some $s \in \mathsf{Good}_{\mathsf{F}}(\beta, X, Y)$. Since for such k, the $\mathsf{CompleteO}$ subroutine sampled a uniform function π_k,

$$\underset{\pi_k}{\mathsf{Pr}}[\mathsf{F}^{\pi_k}(s, X_1) = Y_1] \leq \underset{\pi_k}{\mathsf{Pr}}[\forall i \in [l] : \mathsf{F}^{\pi_k}(s, X_1[i]) = Y_1[i]] \ , \tag{2.10}$$

Since $s \in \mathsf{Good}_{\mathsf{F}}(\beta, X, Y)$ we know that there exists some subset $I_s \subseteq [l]$ of size $l/(c-1) - l/2c$ where for every $i \neq j \in I_s$, we have

1. $\mathsf{C}(s, X_1[i]) \neq \mathsf{C}(s, X_1[j])$
2. $Y_1[i]$ is 1/2-good w.r.t. $(s, X_1[i])$

Furthermore, no queries to O_k have been made, therefore none of the $\mathsf{C}(s, X_1[i])$ have been queried yet. Therefore, we can bound the probability in Equation 2.10 by a simple application of Lemma 4 by $\frac{1}{2^t}$ where $t = \frac{l}{2c(c-1)} \cdot (c+1)$. The final bound follows by a union bound over all such 2^n k's and 2^σ s's. ■

Next, we bound the probability of bad_2 being set conditioned $\neg\mathsf{Bad}$.

Claim 3 $\Pr[\mathsf{bad}_2 | \neg\mathsf{Bad}] \leq \frac{2^\sigma}{2^t}$, *where* $t = \frac{l}{2c(c-1)}$.

Proof: By the description of Game $\hat{\mathsf{G}}^i$, we note that bad_2 is set only if for the query $(X = X_1||X_2, Y = Y_1||Y_2)$ there exists some $s \in \mathsf{Good}_{\mathsf{F}}(\beta, X, Y)$ and such that $\mathsf{F}^f((s,k), X) = Y$. Here, we will condition on $\neg\mathsf{Bad}$. Since BadO doesn't happen, the responses to the oracle f and O are independent.

Recall that the R_2 query corresponding to this R_3 query was for some $j \leq i$. Let us assume that A makes exactly p distinct queries to f. Since A only has non-adaptive access to f, it makes all its queries to f at once. Let us assume the p distinct queries are the t-th, $t+1$-th, ..., $(t+p-1)$-th queries. Now there are two cases to consider here depending on the ordering of R_2 and f queries. The first case is when $j < t$, that is, the R_2 query was made earlier than the first f query and the second case is that $j > t+p-1$.

- Case (a) $j < t$: We condition on $\neg\mathsf{Badf}$. Since the event Badf doesn't happen we only need to focus on $s \in \mathsf{Good}_{\mathsf{F}}(\beta, X, Y)$ for which at most $l/2$ of all the l many $\mathsf{C}(s, X_1[i])$ were queried to f. Fix one such s. We make three observations for this

s (1) at least $l/2$ of the l many $\mathsf{C}(s, X_1[i])$ are not yet queried to f, (2) since there are no c way collisions in $\mathsf{C}(s, X_1)$, at least $l/2(c-1)$ of the l many $\mathsf{C}(s, X_1[i])$'s are distinct, and (3) there are at most $l/2c$ $Y_1[i]$'s that are 1/2-bad w.r.t. $(s, X_1[i])$ (by definition of s). Combining (1), (2) and (3), we have that there exists $I_s \subseteq [l]$ of size at least $l/2(c-1) - l/2c$ such that for all $i \neq j \in I_s$,

1. $\mathsf{C}(s, X_1[i]) \neq \mathsf{C}(s, X_1[j])$
2. $\mathsf{C}(s, X_1[i]) \notin \mathsf{Query}(Q_f)$
3. $Y_1[i]$'s are 1/2-good w.r.t. $(s, X_1[i])$.

Then the bound follows from a simple application of Lemma 4 with parameter $t = \frac{l}{2c(c-1)}$. Taking union bound over all such s, we have that,

$$\mathsf{Pr}[\mathsf{bad}_2 \wedge j < t | \neg \mathsf{Bad}] \leq \frac{2^\sigma}{2^t} \ , \tag{2.11}$$

where $t = \frac{l}{2c(c-1)}$.

- Case (b) $j > t + p - 1$: We now condition on $\neg\mathsf{BadR}$. Let us fix some $s \in \mathsf{Good}_\mathsf{F}(\beta, X, Y)$. Since the event BadR didn't happen we know that for every β-sparse s (and hence every $s \in \mathsf{Good}_\mathsf{F}(\beta, X, Y)$) at least $l/2$ of the $\mathsf{C}(s, X_2[i])$'s are not in the set $\mathsf{Query}(Q_f)$. Then by the same argument as done in Case (a) when $j < t$, we note that there exists a set I_s of size $l/2(c-1) - l/2c$ such that for all $i \neq j \in I_s$,

  1. $\mathsf{C}(s, X_2[i]) \neq \mathsf{C}(s, X_2[j])$
  2. $\mathsf{C}(s, X_2[i]) \notin \mathsf{Query}(Q_f)$
  3. $Y_2[i]$'s are 1/2-good w.r.t. $(s, X_2[i])$.

  Then the bound follows from a simple application of Lemma 4 with parameter $t = \frac{l}{2c(c-1)}$.

Taking union bound over all such s, we have that,

$$\mathsf{Pr}[\mathsf{bad}_2 \wedge j > t + p - 1 | \neg \mathsf{Bad}] \leq \frac{2^\sigma}{2^t} \ , \tag{2.12}$$

where $t = \frac{l}{2c(c-1)}$.

Since Case (a) and Case (b) are exclusive, from Equation (2.11) and Equation (2.12) we have,

$$\mathsf{Pr}[\mathsf{bad}_2 | \neg \mathsf{Bad}] \leq \frac{2^\sigma}{2^t} \ , \tag{2.13}$$

where $t = \frac{l}{2c(c-1)}$. ■

Finally, we arrive at the probability of bad being set by combining Claim 1,Claim 2 and Claim 3,

$$\begin{aligned}\mathsf{Pr}[\mathsf{bad}] &\leq \mathsf{Pr}[\mathsf{bad}_1] + \mathsf{Pr}[\mathsf{bad}_2] \leq \mathsf{Pr}[\mathsf{bad}_1] + \mathsf{Pr}[\mathsf{bad}_2 | \neg \mathsf{Bad}] + \mathsf{Pr}[\mathsf{Bad}] \\ &\leq \frac{2^{\sigma+n}}{2^{t(c+1)}} + \frac{2^\sigma}{2^t} + \frac{6q}{2^n} + q \cdot \frac{2^\sigma}{2^t} + q \binom{l}{l/2} (2\beta \cdot q)^{l/2} \ ,\end{aligned}$$

where $t = \frac{l}{2c(c-1)}$. This concludes the proof of Lemma 7.

2.5.3 Proof of Lemma 6

The proof is identical to the proof of Lemma 5. More precisely we consider a sequence of hybrids H^i except that there is no dependence between the oracles f and O as in case of G^i. The indistinguishability of neighboring games H^i and H^{i+1} by bounding the probability of bad flag being set in H^i computed identically to the probability of bad being set in G^i (discussed in Lemma 7). The only difference is that in the case of H^i the event BadO happens with probability zero as there is no k^* such that $f = \mathsf{O}(k^*, \cdot)$. The rest of the analysis is identical to Lemma 7. We skip the formal proof.

2.5.4 Proof of Claim 1

We bound the probability of the event Bad by first individually bounding the probabilities of each of the three bad events BadO, Badf and BadR defined in Section 2.5.1, and then applying a simple union bound.

Claim 4 $\Pr[\mathsf{BadO}] \leq \frac{2q}{2^n}$.

Proof: First note that R_1 and R_2 are independent of oracles f and O, and responses by R_3 within the first i queries only depend on $k \in Q$. Therefore querying the oracle $\mathsf{R} = (\mathsf{R}_1, \mathsf{R}_2, \mathsf{R}_3)$ gives no more information to A about k^* than it already has from its queries to (f, O). Therefore, the probability of querying the oracle O on key k^* in the $j+1$-st query is no better than $\frac{1}{2^n - j}$. Using this, the probability of the event BadO happening can be upper bounded by $\sum_{j=0}^{i-1} \frac{1}{2^n - j}$ (i.e., union bound over all q queries). Finally note that as $i \leq q << 2^{n/2}$, the later can be upper bounded by $2i/2^n$. ■

Next, we proceed to bound the probabilities of events BadR and Badf. Here we will be conditioning on $\neg\mathsf{BadO}$ which ensures that no queries to O on key k^* are made, that is, $Q_{k^*} = \phi$. This, furthermore ensures that (a) responses by oracles f and O are independent of each other, and (2) allows us to view $Q = Q_f$ in the definitions of BadR (Definition 11) and Badf (Definition 12).

Claim 5 $\Pr[\mathsf{BadR} | \neg\mathsf{BadO}] \leq q \; 2^{\sigma} \; \binom{l}{l/2} \; (2\beta \cdot q)^{l/2}$.

Proof: Recall that the event BadR occurs in $\hat{\mathsf{G}}^i$ if within the first i queries, immediately after an assignment $T_2[Y_1] \xleftarrow{\$} (\{0,1\}^n \setminus T_1)^{[l]}$ (happens during an $\mathsf{R}_2(T_1, Y_1)$ query), there exists some s such that the following holds for $Q = Q_{k^*} \cup Q_f$,

1. s is β-sparse.

2. there exists some $I_s \subseteq [l]$ of size at least $l/2$ such that for every $i \in I_s$,

$$\mathsf{C}(s, T_2[Y_1][i]) \in \mathsf{Query}(Q) \ ,$$

where by $\mathsf{Query}(Q) = \{w : (w, y) \in Q\}$.

To bound BadR, we will condition on $\neg\mathsf{BadO}$ which ensures that $Q_{k^*} = \phi$ and hence $Q = Q_f$ in the definition of BadR. If $Q_f = \phi$ within the first i queries then the set Q is also empty, conditioned on which the probability of BadR occurring is 0. So, w.l.o.g. let us assume that $Q = Q_f \neq \phi$. This means some queries to f were made within the first i queries. In fact, since A has only non-adaptive access and the $i+1$-st query is to R_3, A must then have made all its queries to f within the first i queries. Let j-th query be the first query to f for some $0 < j \leq i$. Let us assume that there are $p \leq q$ distinct queries to f. Then $j + p - 1$-th query is the last query to f. We denote the set of queries made to f with $\mathsf{Query}(Q_f)$.

Now, let us consider some R_2 query made after these f queries. Let the query be on some (X_1, Y_1). Let us furthermore assume that X_2 is the response of this R_2 query, that is, $X_2 = \mathsf{R}_2(X_1, Y_1)$. If $X_1 \neq T_1$ or if $\mathsf{isValid}(X_1, Y_1) \neq 1$ then $X_2 = \perp$ and conditioned on the probability of BadR occurring is zero. Let us, w.l.o.g. assume that X_1 and Y_1 are indeed as necessary then $X_2 = T_2[Y_1]$ is a l-length tuple of n-bit strings. Since A only makes distinct queries, it must be that $X_2 = T_2[Y_1]$ was freshly sampled at random. The event BadR occurs for such an R_2 query if for the freshly chosen, random tuple X_2 there exists some seed s that is β-sparse for which there exists $I_s \subseteq [l]$ of size $l/2$ such that for all $i \in I_s$, we have

$$\mathsf{C}(s, X_2[i]) \in \mathsf{Query}(Q) \ .$$

Let us fix one such s that is β-sparse. We are interested in the following,

$$\Pr_{X_2}\left[\substack{\exists I_s \subseteq [l] \\ |I_s| = l/2} : \forall i \in I_s, \mathsf{C}_s(X_2[i]) \in \mathsf{Query}(Q)\right] \ .$$

Let us furthermore fix some arbitrary subset $I_s = [l]$ of size $l/2$.

By definition of β-sparseness, we know that for every $w \in \mathsf{Query}(Q_f)$ the number of pre-images under s is at most $\beta \cdot 2^n$. Therefore, the number of pre-images of $\mathsf{Query}(Q_f)$ under s is at most $\beta \cdot p \cdot 2^n$. Therefore, for the above fixed I_s,

$$\Pr_{X_2}[\forall i \in I_s : \mathsf{C}_s(X_2[i]) \in \mathsf{Query}(Q)] \leq \prod_{i=0}^{l/2-1} \frac{\beta \cdot p2^n - i}{2^n - (3l/2 + i)} ,$$

where recall that X_2 is sampled with replacement from $\{0,1\}^n \setminus T_1$. Since $2l < 2^{n/2}$,

$$\prod_{i=0}^{l/2-1} \frac{\beta \cdot p2^n - i}{2^n - (3l/2 + i)} \leq \left(\frac{\beta \cdot p2^n}{2^n - (2l-1)}\right)^{l/2} \leq (2\beta p)^{l/2} .$$

Therefore,

$$\Pr_{X_2}[\forall i \in I_s : \mathsf{C}_s(X_2[i]) \in \mathsf{Query}(Q)] \leq (2\beta p)^{l/2} .$$

The final bound follows by taking a union bound over all $\binom{l}{l/2}$ number of subsets of $[l]$ of size $l/2$, over all 2^σ s's and over all q queries to R_2. ■

Claim 6 $\mathsf{Pr}[\mathsf{Badf}|\neg\mathsf{BadO}] \leq \frac{q \cdot 2^\sigma}{2^t}$, *where* $t = \frac{l}{2c(c-1)}$.

The idea of the proof is pretty similar to the Proof of Lemma 4 presented in the discussion.

Proof: Throughout the proof we will condition on BadO not happening which implies $Q_{k^*} = \phi$. We divide the proof into two cases depending on when queries f were made.

- Case A: No queries to f were made within the first i queries (i.e., $Q_f = \phi$). Then, the probability of Badf is 0 (by Definition 12).

- Case B: Let us now consider the case f was queried within the first i queries. Since A has only non-adaptive access to f it must make all its queries to f before this $i+1$-th query to R_3. Let us assume that A makes $p \leq q$ distinct queries to f and denote this set of queries by $\mathsf{Query}(Q_f)$.

Let us fix some query to R_2 on Y_1 that occurs before the first query to f and fix some s that satisfies the following conditions:

1. $T_2[Y_1] \neq \perp$.
2. there are no c-way collisions in $\mathsf{C}(s, T_1)$.
3. for $I_s = \{i \in [l] : \mathsf{C}(s, X_1[i]) \in \mathsf{Query}(Q_f)\}$, we have $|I_s| \geq l/2$.
4. $|\{i : Y_1[i] \text{ is } 1/2\text{-bad w.r.t. } (s, T_1[i])\}| \leq l/2c$.

Badf occurs for the pair (Y_1, s) iff for all $i \in I_s$ the responses $Z_1[i] = f(W_1[i])$ were such that $\mathsf{G}(s, T_1[i], Z_1[i]) = Y_1[i]$. We are interested in computing the probability of this happening. Recall that, since there are no c-way collisions in $\mathsf{C}(s, T_1[i])$ it must be that are at least $l/(c-1)$ distinct elements in $\mathsf{C}(s, T_1[i])$. Furthermore, it must be that there are at least $l/2(c-1)$ distinct elements among W_1's restricted to the set I_s. That is,

$$|\{W_1[i] : \forall i \in I_s\}| > l/2(c-1) \ .$$

Next, there are at most $l/2c$ i's in $[l]$ for which $Y_1[i]$ is $1/2$-bad (by assumption on Y_1 and s). Therefore, we can safely conclude that there exists a subset $J_s \subseteq [l]$ of size at least $l/2(c-1) - l/2c$ such that for every $i \neq j \in J_s$, we have

1. $W_1[i] \neq W_1[j]$.
2. $Y_1[i]$ is $1/2$-good w.r.t. $(s, T_1[i])$.

Since we are conditioning on $\neg\mathsf{BadO}$, we know that no queries to f (or O_k^*) have be made earlier. Therefore, $Z_1[i] = f(\mathsf{C}(s, T_1[i]))$ for all $i \in J_s$ are sampled at random. So the probability of Badf happening for the fixed (s, Y_1) can be upper bounded by the probability that over the random choice of $Z_1[i]$'s, we have $\mathsf{F}^f(s, X_1[i]) = Y_1[i]$. This can be upper bounded by $\frac{1}{2^t}$ where $t = |J_s| \geq l/2(c-1) - l/2c = \frac{l}{2c(c-1)}$.

Now, for every such Y_1 there exists some R_2 query on $(\cdot, Y_1)$. Therefore, there are at most q such Y_1's. Final bound follows by taking union bound over all 2^σ s's' and q such Y_1's. ■

Proof of Claim 1. Finally, we are ready to bound the probability of Bad. Note that,

$$\begin{aligned}\mathsf{Pr}[\mathsf{Bad}] &\leq \mathsf{Pr}[\mathsf{BadO} \vee \mathsf{BadR} \vee \mathsf{Badf}] \ , \\ &\leq 3 \cdot \mathsf{Pr}[\mathsf{BadO}] + \mathsf{Pr}[\mathsf{Badf}|\neg\mathsf{BadO}] + \mathsf{Pr}[\mathsf{BadR}|\neg\mathsf{BadO}] \ , \\ &\leq \frac{6q}{2^n} + q \cdot \frac{2^\sigma}{2^t} + q\ 2^\sigma \binom{l}{l/2} (2\beta \cdot q)^{l/2} \ , \text{ where } t = \frac{l}{2c(c-1)} \ ,\end{aligned}$$

where inequalities follow from Claim 4,Claim 5,Claim 6.

2.6 The Case of Biased F's: Proof of Proposition 3

Let us fix some functions r, m, σ and some $1/c$-biased function family $\mathsf{F}[\mathsf{C}, \mathsf{G}]$ (as in the statement of Proposition 3). Since, F is $1/c$-biased, we know by Definition 4 that there exists some sufficiently large polynomial $l = \omega(\sigma)$ and some non-negligible function ε_l such that for all $n \in \mathbb{N}$,

$$\underset{X,s,f}{\mathsf{Pr}} [|\{i : Y[i] \text{ is } 1/2\text{-bad w.r.t. } (s, X[i])\}\}| \geq l/c] \geq \varepsilon_l(n) \ . \tag{2.14}$$

Let us fix one such l and ε_l. In the rest of this section we prove Proposition 3 as follows: First in Section 2.6.1 we provide our oracles (O, R) (that depends on $\mathsf{F}[\mathsf{C}, \mathsf{G}]$ and parameters l, c) and function family H^{O}. Then in Section 2.6.2 we show that F is not a PRF relative to (O, R) and finally in Section 2.6.3 we discuss the non-adaptive PRF security of H relative to (O, R).

2.6.1 Oracle (O, R) and H^{O}

Oracle O and the function family H^{O} are identical to the ones defined in Section 2.4.2.

Oracle $\mathsf{R}_1()$:	Oracle $\mathsf{R}_2(X, Y)$:
$X \xleftarrow{\$} (\{0,1\}^n)^{[l]}$	**if** $\exists s$ s.t. $\lvert\{i : Y[i] \text{ is } 1/2\text{-bad w.r.t. } (s, X[i])\}\rvert \geq l/c$ **then**
return X	**return** 1
	return $\perp$

Figure 2.5: The oracle R for Section 2.6.

The oracle R is decomposed into $(\mathsf{R}_1, \mathsf{R}_2)$, where R_1 returns $X \xleftarrow{\$} (\{0,1\}^n)^{[l]}$. The oracle R_2 accepts an l-length tuple $(X[i], Y[i])$ where $X[i]$'s are distinct n-bit strings and $Y[i]$'s are m-bit strings (possibly not distinct). R_2 returns 1 iff it finds some seed s and set $I_s \subseteq [l]$ of size l/c such that $Y[i]$'s are 1/2-bad w.r.t. $(s, X[i])$ for all $i \in I_s$. We give the formal description of R in Fig. 2.5. This completes the description of our oracles.

Remark 5 *On first look it might seem that the oracle R_1 is not necessary. Indeed, an adversary that can receive the parameter l as non-uniform advice should be able to simulate R_1 perfectly. However, we are in the regime of uniform security where adversaries cannot receive non-uniform advice. We solve this issue by designing an oracle R_1 that provides l to a uniform adversary upon query.*

2.6.2 F is not a PRF relative to (O, R)

First let us prove the following useful claim that captures the probability that R's outputs 1 when $Y[i]$'s are outputs of a random function.

Claim 7 *For any $n, l \in \mathbb{N}$, any $X \in (\{0,1\}^n)^{[l]}$,*

$$\Pr_Y[\exists s : \lvert\{i : Y[i] \text{ is } 1/2\text{-bad w.r.t. } (s, X[i])\}\rvert > l/c] \leq \binom{l}{l/c} \cdot \frac{2^\sigma}{2^{ml/c}} ,$$

where $Y \xleftarrow{\$} (\{0,1\}^m)^l$.

Proof: Let us fix some s and some $I_s \subseteq [l]$ of size l/c. We are interested in the probability that,

$$\Pr_{Y[1],\ldots,Y[l]}[\forall i \in I_s : \ Y[i] \text{ is } 1/2\text{-bad w.r.t. } (s, X[i]) \ ] \ ,$$

For any $(s, X[i])$ there can be at most one $y_{s,X[i]}$ which is 1/2-bad. That is, there is at most one $y_{s,X[i]}$ for which,

$$\Pr_{z}[\mathsf{G}(s, X[i], z) = y_{s,X[i]}] > 1/2 \ .$$

Now the probability that a randomly sampled $Y[i]$ equals $y_{s,X[i]}$ is at most $1/2^m$. Therefore,

$$\Pr_{Y[1],\ldots,Y[l]}[\forall i \in I_s : \ Y[i] \text{ is } 1/2\text{-bad w.r.t. } (s, X[i]) \ ] \leq \left(\frac{1}{2^m}\right)^{l/c} \ .$$

The proof follows from a union bound over all 2^σ seeds s and $\binom{l}{l/c}$ subsets I_s. ■

Next, we show that relative to (O, R) the construction $\mathsf{F}^{\mathsf{H}}[\mathsf{C}, \mathsf{G}]$ is not secure.

Lemma 8 (F is insecure relative to (O, R)) *There exists a PPT adversary A relative to (O, R) and a non-negligible function ε such that for every $n \in \mathbb{N}$,*

$$\mathsf{Adv}^{\mathsf{rel-prf}}_{A,\mathsf{F},(\mathsf{O},\mathsf{R})}(n) \geq \varepsilon(n) \ .$$

Proof: First, we define the adversary A and then argue that it achieves a non-negligible advantage. A relative to (O, R) proceeds by querying R_1 to receive $X \xleftarrow{\$} (\{0,1\}^n)^{[l]}$ and then queries its challenge oracle f to compute $Y = f(X)$. A then queries $\mathsf{R}_2(X, Y)$ and outputs whatever R_2 outputs.

Now, let us first consider the case when A is interacting with $f = \mathsf{F}^{\mathsf{H}}((s,k), \cdot)$ for some randomly chosen s and k. Recall that $\mathsf{H}_k(\cdot) = \mathsf{O}(k, \cdot)$ is sampled randomly from

$\mathsf{Funcs}(n,r)$ and hence has the same distribution as the $f \xleftarrow{\$} \mathsf{Funcs}(n,r)$. Since, F is $1/c$-biased there exists some s with probability ε_l (this was defined at the begining of Section 2.6) for which at least l/c of the $Y[i]$'s are 1/2-bad w.r.t. $(s, X[i])$. Therefore, R_2 returns 1 with probability at least ε_l in this case.

Now consider the case when A is interacting with $f \xleftarrow{\$} \mathsf{Funcs}(n,m)$. Since $X[i]$'s are distinct, the corresponding $Y[1] = f(X[1]), \ldots, Y[l] = f(X[l])$ are l randomly chosen n-bit strings. Then by Claim 7 the probability that R_2 returns 1 is at most $\binom{l}{l/c} \cdot \frac{2^\sigma}{2^{ml/c}}$. Since, $\binom{n}{c} \leq (en/c)^c$, we can upperbound $\binom{l}{l/c}$ by $ce^{l/c}$. Therefore, probability that R_2 outputs 1 in this case can be upper bounded by $2^\sigma \cdot (ce/2^m)^{l/c}$. Since $2^m \geq 2ce$ and $l = \omega(\sigma)$ and $c = O(1)$, R_2 returns 1 in this case only with negligible probability.

Therefore, combining both cases we conclude that A achieves a non-negligible advantage in breaking F relative to (O, R). ■

2.6.3 H is naPRF relative to (O, R).

Next we show that relative to (O, R), H remains a naPRF. Note that the oracle R is completely independent of the oracle O, and hence also independent of H. This allows us to reduce the non-adaptive security of H relative to (O, R) to the non-adaptive security of H relative to O. That is, for every non-adaptive PPT adversary A relative to (O, R) participating in the naPRF-security of H, we can construct a non-adaptive computationally unbounded adversary B relative to only O. B, with l hardwired, internally runs A and answers A's queries to oracles O and h by forwarding to its own oracle (O, h) while perfectly simulating the oracle R for A. This is possible because R can be computed by an unbounded adversary. Next, note that if A makes q queries to its oracles then B also makes only q queries to its oracles. We claim (without proof) that adversary B making at most q queries to its oracle achieves advantage at most $2q/2^n$ in the naPRF

security game of H relative to only O, which clearly upperbounds the advantage of A in the naPRF security of H relative to (O, R).

Lemma 9 *For any adversary A making q polynomially bounded queries to its oracles where it only makes non-adaptive queries to h, for every $n \in \mathbb{N}$,*

$$\mathsf{Adv}^{\mathsf{rel-naprf}}_{A,\mathsf{H},(\mathsf{O},\mathsf{R})} \leq \frac{2q(n)}{2^n} \ .$$

Combining Lemma 9 and Lemma 8 concludes the proof of Proposition 3.

2.7 Removing c-universality Assumption on C

In this section we study two special cases of the construction $\mathsf{F}^{(\cdot)}[\mathsf{C}, \mathsf{G}]$ and rule out $\mathsf{F}^{(\cdot)}$ as a fully black-box construction of a PRF from a naPRF for any C.

2.7.1 1-call Constructions with Fixed Post-processing Function

In this section, we study the first special case of the construction $\mathsf{F}^{(\cdot)}[\mathsf{C}, \mathsf{G}]$. We begin by describing the construction formally and then provide the black-box separation.

Construction $\mathsf{F}^{\mathsf{H}}[\mathsf{C}, g]$. Let σ, r, m be any polynomially bounded functions. Let C be a function family from n bits to n bits with σ-bit seeds, let H be a function family on n bits to r bits with n-bit seeds and let g be a function from $2n + r$ bits to m bits with σ-bit seeds. Consider the family $\mathsf{F}^{\mathsf{H}}[\mathsf{C}, g]$ from n bits to m bits with $\sigma + n$-bit seeds such that $\mathsf{F.Kg}$ outputs (s, k) where s is a random σ-bit seed for C. And the evaluation for F on x proceeds as follows,

$$y = \mathsf{F}^{\mathsf{H}}((s,k),x) = g(x,w,z) \quad \text{where } w = \mathsf{C}(s,x) \ ; \ z = \mathsf{H}(k,w) \ . \tag{2.15}$$

Note that $\mathsf{F}^{\mathsf{H}}[\mathsf{C}, g]$ is a special case of the construction $\mathsf{F}^{\mathsf{H}}[\mathsf{C}, \mathsf{G}]$ from Section 2.3 as for every g there exists a family G where $\mathsf{G}(s, x, z)$ can simulate $g(x, w, z)$ as G can compute $w = \mathsf{C}(s, x)$ with its seed.

Theorem 3 rules out $\mathsf{F}[\mathsf{C}, g]$ as a fully black-box construction of a PRF from a naPRF for every C and g.

Theorem 3 *Let r, σ, m be any polynomially bounded functions such that $m = \Omega(n + r)$. Then for every oracle function family $\mathsf{F}^{(\cdot)}[\mathsf{C}, g]$ (defined in Equation 2.15) there exists an oracle (O, R) relative to which naPRF H exists but $\mathsf{F}^{\mathsf{H}}[\mathsf{C}, g]$ is not a PRF.*

Proof: Towards proving Theorem 3, let $c \geq 2$ be the smallest constant such that $m \geq (n + r)/c + \omega(\log n)$. Note that such a c exists as $m = \Omega(n + r)$. Theorem 3 follows from the following two cases,

- Case (a) C is c-universal: Here, Theorem 25 provides us with the relevant oracles (note that $m \geq \log(8ce)$ as $m = \Omega(n + r)$) implying Theorem 3 for c-universal C's.
- Case (b) C is not c-universal: In Lemma 10 we present the necessary oracles to handle this case. This concludes the proof of Theorem 3.

■

Lemma 10 *Let $c = O(1)$ and r, σ, m be any polynomially bounded functions such that $m \geq (n + r)/c + \omega(\log n)$. Then for every function family $\mathsf{F}^{(\cdot)}[\mathsf{C}, g]$ (defined in Equation 2.15) where C is not c-universal there exists an oracle (O, R) relative to which naPRF H exists but $\mathsf{F}^{\mathsf{H}}[\mathsf{C}, g]$ is not a PRF.*

Proof: Since C is not c-universal, there exists some non-negligible function ε such that for every $n \in \mathbb{N}$ there exists distinct $x_1, \ldots, x_c \in \{0,1\}^n$,

$$\Pr_s[\mathsf{C}(s, x_1) = \ldots = \mathsf{C}(s, x_c)] \geq \varepsilon(n) \ .$$

Let us fix some n. Let $x_1, \ldots, x_c$ be the corresponding distinct inputs for which collisions are likely. Now, under a random s not only do they share the same $w = \mathsf{C}(s, x_i)$ but for any function H and any k, they also share the same $z = \mathsf{H}(k, w) = \mathsf{H}(k, \mathsf{C}(s, x_i))$. Now, we consider the set $\mathcal{Y}_g$ of all possible outputs $(y_1, \ldots, y_l)$

$$\mathcal{Y}_g = \{(y_1, \ldots, y_k) : \exists (w, z) \text{ s.t. } g(x_i, w, z) = y_i \forall i \in [3]\} \ .$$

Clearly, $|\mathcal{Y}_g| \leq 2^{n+r}$. However, the $(y_1, \ldots, y_k)$ where $y_i = f(x_i)$ when f is a random function from n-bits to m-bits come from a set of size 2^{mc}. We exploit this fact to build an oracle (O, R) relative to which a secure naPRF H exists but $\mathsf{F}^{\mathsf{H}}[\mathsf{C}, g]$ is not a PRF.

Oracle R. The oracle R is decomposed into two oracles $(\mathsf{R}_1, \mathsf{R}_2)$ such that R_1 on input 1^n provides strings $(x_1, \ldots, x_c)$ which collide under C with non-negligible probability ε. The oracle R_2 just accepts a tuple of c inputs $(x_1, \ldots, x_c)$ and c outputs $(y_1, \ldots, y_c)$ and returns 1 iff there exist some (w, z) such that $g(x_i, w, z) = y_i$ for all $i \in [c]$.

Oracle O and H^{O}. The oracle O and H^{O} are identical to the ones described in Section 2.4.2.

F is not a PRF relative to (O, R). Consider a uniform adversary A that first queries R_1 to receive likely collisions $x_1, \ldots, x_c$ of C. Then, it computes $y_i = f(x_i)$ by making queries to its challenge oracle f and outputs whatever R_2 for $((x_1, \ldots, x_c), (y_1, \ldots, y_c))$. When A is interacting with $f \xleftarrow{\$} \mathsf{F}$, oracle R_2 outputs 1 with probability ε. But, as discussed above, R_2 outputs 1 with probability at most $2^{n+r}/2^{mc}$ when $f \xleftarrow{\$} \mathsf{Funcs}(n, m)$. Since, $m \geq (n + r)/c + \omega(\log n)$, we know R_2 outputs 1 when f is random only with negligible probability. Hence, A achieves non-negligible advantage in breaking F.

H is naPRF relative to (O, R). Recall that R is independent of O and furthermore can be computed by a computationally unbounded adversary. Therefore, as done in the

Proof of Proposition 3 we can reduce the naPRF security of H relative to (O, R) to naPRF security of H relative to O. This concludes the proof of Lemma 10. ■

2.7.2 1-call Constructions with Only Pre-processing

In this section, we study the most basic one-call construction which is a composition of preprocessing function C with the naPRF H. We begin by describing the construction formally and then provide the black-box separation.

Construction $\mathsf{F}^{\mathsf{H}}[\mathsf{C}]$. Let σ, r, m be any polynomially bounded functions. Let C be a function family from n bits to n bits with σ-bit seeds, let H be a function family on n bits to m bits with n-bit keys. Consider the family $\mathsf{F}^{\mathsf{H}}[\mathsf{C}]$ from n bits to m bits with $\sigma + n$-bit seeds such that $\mathsf{F.Kg}$ outputs (s, k) where s is a random σ-bit seed for C. And the evaluation for F on x proceeds as follows,

$$y = \mathsf{F}((s, k), x) = \mathsf{H}(k, \mathsf{C}(s, x))[1, \ldots, m] \ . \tag{2.16}$$

Theorem 4 rules out $\mathsf{F}[\mathsf{C}, g]$ as a fully black-box construction of a PRF from a naPRF for every C and g.

Theorem 4 *Let σ, m be any polynomially bounded functions such that $m = \omega(\log n)$. Then for every oracle function family $\mathsf{F}^{(\cdot)}[\mathsf{C}, g]$ (defined in Equation 2.15) there exists an oracle (O, R) relative to which naPRF H exists but $\mathsf{F}^{\mathsf{H}}[\mathsf{C}, g]$ is not a PRF.*

Proof: Theorem 4 follows from the following two cases –

- Case (a) - C is 2-universal: Here, Theorem 25 provides us with the relevant oracles (note that $m \geq \log(16e)$ as $m = \omega(\log n)$) implying Theorem 4 for C's which are 2-universal.

- Case (b) - C is not 2-universal: Here, it is clear that collisions in C lead to collisions in $\mathsf{F}[\mathsf{C}]$. That is, if for inputs $x_1 \neq x_2$ and some non-negligible function ε, $\mathsf{C}(s, x_1) = \mathsf{C}(s, x_2)$ with probability ε then $\mathsf{F}((s,k), x_1) = \mathsf{F}((s,k), x_2)$ also with probability ε over the choice of (s,k). However, such collisions happen only with negligible probability for a random function whenever $m = \omega(\log n)$. Therefore, an adversary that receives collisions in C (as non-uniform advice) can break the PRF security of F. But, to show the separation in our case (uniform security), one needs to exhibit a uniform adversary that breaks F. We make the adversary uniform by designing an oracle that provides collisions in C. We handle this case explicitly in Lemma 11.

This concludes the proof of Theorem 4. ■

Lemma 11 *Let σ, m be polynomially bounded functions such that $m = \omega(\log n)$. Then for every oracle function family $\mathsf{F}^{(\cdot)}[\mathsf{C}]$ (defined in Equation 2.16) where C is not 2-universal there exists an oracle (O, R) relative to which naPRF H exists but $\mathsf{F}^{\mathsf{H}}[\mathsf{C}, g]$ is not a PRF.*

Proof: Since C is not 2-universal, there exists some non-negligible function ε such that for every $n \in \mathbb{N}$ there exists $x_1 \neq x_2 \in \{0,1\}^n$,

$$\Pr_s[\mathsf{C}(s, x_1) = \mathsf{C}(s, x_2)] \geq \varepsilon(n) \ .$$

We exploit this fact to build an oracle (O, R) relative to which naPRF H exists while $\mathsf{F}[\mathsf{C}]$ is not a PRF.

Oracle R. The oracle R on input 1^n provides strings (x_1, x_2) which collide under C with non-negligible probability ε.

Oracle O and H^{O}. The oracle O and H^O are identical to the ones described in Section 2.4.2.

F is not a PRF relative to (O, R). Consider a uniform adversary A that first queries R to receive likely collisions x_1, x_2 of C. Then, it computes $y_i = f(x_i)$ by making queries to its challenge oracle f and outputs 1 iff $y_1 = y_2$. When $f \xleftarrow{\$} \mathsf{F}$, A outputs 1 with probability ε but A outputs 1 only with negligible probability when $f \xleftarrow{\$} \mathsf{Funcs}(n, m)$, thereby achieving a non-negligible advantage in breaking the PRF-security of F.

H is naPRF relative to (O, R). Recall that R is independent of O and furthermore can be computed by a computationally unbounded adversary. Therefore, as done in the Proof of Proposition 3 we can reduce the naPRF security of H relative to (O, R) to naPRF security of H relative to O. This concludes the proof of Lemma 11. ■

2.8 Multiple-call Constructions

We devote this section to study multiple call constructions. First, in Section 2.8.1, we lift our techniques from Section 2.3 to a specific 2-call construction (and its generalization to arbitrary calls). This result is a generalization of [1] which only rules our arbitrary parallel composition of naPRFs as a PRF. In Section 2.8.2 we provide examples of constructions making two- and four-calls which cannot be ruled out using our techniques. While it is unlikely that these constructions will admit a security proof, we want to highlight that ruling them out will require new techniques (or at least new designs of separation oracles).

2.8.1 Ruling out Two-call Cuckoo-hashing based Construction

Let σ, m be polynomially bounded functions, C (resp., G) be a function family from n bits to n bits (resp., m bits) with σ-bit seeds, and let H be a function family on n bits to m bits with n-bit seeds. Consider the family $\mathsf{F}^{\mathsf{H}}[\mathsf{C}, \mathsf{G}]$ (Figure 2.1b) from n bits

to m bits with $(3\sigma + 2n)$-bit seeds such that for every $n \in \mathbb{N}$, $\mathsf{F.Kg}(1^n)$ outputs $(\vec{s}, \vec{k})$ where $\vec{s} \xleftarrow{\$} \left(\{0,1\}^{\sigma(n)}\right)^3$ and $\vec{k} \xleftarrow{\$} (\{0,1\}^n)^2$. The evaluation of F on $x \in \{0,1\}^n$ with $\vec{s} = (s_1, s_2, s_3)$ and $\vec{k} = (k_1, k_2)$ is,

$$y = \mathsf{F}^{\mathsf{H}}((\vec{s}, \vec{k}), x) = \mathsf{H}(k_1, \mathsf{C}(s_1, x)) \oplus \mathsf{H}(k_2, \mathsf{C}(s_2, x)) \oplus \mathsf{G}(s_3, x) \ . \tag{2.17}$$

We note that the construction in Equation (2.17) covers the cuckoo-hashing based naPRF to PRF construction [8] – one recovers their construction by letting C and G be hash function family with sufficient independence. Informally, they showed that for every polynomial time computable function t, if C and G are $O(\log t(n))$-wise independent and H is a naPRF secure for adversaries making at most t queries, then F is a PRF for adversaries making at most $t/4$ queries.[8]

Below we state our result which provides an oracle relative to which there exists an naPRF H such that F (in Equation (2.17)) is not a PRF as long as C is a 2-universal hash function family. This in turn implies that F cannot be a fully black-box construction of a PRF from a naPRF.

Theorem 5 *Let σ, m be polynomially bounded functions, C be a 2-universal family from n bits to n bits and G be a function family from n bits to m bits. Then, for $\mathsf{F}^{(\cdot)}[\mathsf{C}, \mathsf{G}]$ (Equation (2.17)) from n bits to m bits there exists a randomized oracle (O, R) and an oracle function family $\mathsf{H}^{(\cdot)}$ from n bits to m bits with n-bit keys such that $\mathsf{H}^{(\cdot)}$ is a naPRF relative to (O, R) but $\mathsf{F}^{\mathsf{H}}[\mathsf{C}, \mathsf{G}]$ is not a PRF.*

Before we move on to proving Theorem 5, some remarks are in order. First, we emphasize that our result rules out any output length m (even $m = 1$). Secondly, the proof of security from [8] requires C to be a $O(\log n)$-wise independent function family, the later implies our notion of 2-universality. However, we here rule out F as a construction of

[8] they require the range of C to be restricted to the first $4t(n)$ elements of $\{0,1\}^n$

a PRF which, at first, seems to be contradictory to [8]. Here, we emphasize that our focus is on fully-black-box constructions (which are meant to work for any secure naPRF H) whereas the construction [8] depends on the purported security of the underlying naPRF H and hence is not fully-black-box. Thirdly, we emphasize that Theorem 5 readily extends to the case when C is assumed to only be c-universal for some constant $c > 2$ - making much weaker assumptions on C and hence resulting in a stronger negative result. We here choose to focus on the case of $c = 2$ for simplicity.

Finally, the construction in Section 2.8.1 is a specific case of the following $\kappa(n)$-call function family $\mathsf{F}_\kappa^{(\cdot)}[\mathsf{C},\mathsf{G}]$ which takes as seed $(\vec{s},\vec{k})$ where $\vec{s} \in (\{0,1\}^\sigma)^{[\kappa+1]}$ and $\vec{k} \in (\{0,1\}^n)^{[\kappa]}$ and on input $x \in \{0,1\}^n$ evaluates to $y \in \{0,1\}^m$ where

$$y = \mathsf{F}_\kappa^{\mathsf{H}}((\vec{s},\vec{k}),x) = \mathsf{G}(s_{\kappa+1},x) \oplus \bigoplus_{i\in[\kappa(n)]} \mathsf{H}(k_i,\mathsf{C}(s_i,x)) \ . \tag{2.18}$$

We note that Theorem 5 also extends to rule out $\mathsf{F}_\kappa^{(\cdot)}$ for every polynomially bounded, polynomial-time computable function κ. We state the theorem but only prove the case for $\kappa = 2$ for simplicitly or equivalently Theorem 5.

Theorem 6 *Let σ, m, κ be polynomially bounded functions and $c \geq 2$. Let C be a c-universal family from n bits to n bits with σ-bit seeds, and G be any function family from n bits to m bits with σ-bit seeds. Then, for $\mathsf{F}^{(\cdot)}[\mathsf{C},\mathsf{G}]$ (as in Equation 2.18) from n bits to m bits there exists a randomized oracle (O,R) and an oracle function family $\mathsf{H}^{(\cdot)}$ from n bits to m bits with n-bit keys such that $\mathsf{H}^{(\cdot)}$ is a naPRF relative to (O,R) but $\mathsf{F}^{\mathsf{H}}[\mathsf{C},\mathsf{G}]$ is not a PRF relative to (O,R).*

On proofs of Theorem 5 and Theorem 6. The proof of Theorem 5 closely follows that of Proposition 2. At a high level the main challenge in proving Theorem 5 is to show that O remains a secure naPRF relative to (O,R). Recall that a non-adaptive adversary A can break naPRF challenge oracle f relative (O,R) if it can predict the output of F

on challenges $X = X_1||X_2$ issued by R under some $(\vec{s}, \vec{k})$ where either (a) $\vec{k} = (\cdot, f)$ or (b) $\vec{k} = (f, \cdot)$. Because of non-adaptive access to f and iterative nature of R, A is forced to make all its queries to f either after committing to Y_1 (potential outputs for X_1) or before learning X_2. Irrespective of which is the case, A (to trigger case (a)) needs to hope that for sufficiently large set $I \subseteq [|X|]$: $\forall i \in I$ where $X[i]$ is a fresh query to f we have

$$f(\mathsf{C}(s_1, X[i])) = \mathsf{G}(s_3, X[i]) \oplus \mathsf{H}(k, \mathsf{C}(s_2, X[i])) \oplus Y[i] \ .$$

Restricting R to only consider "good" seeds s_1 ensures that enough of $\mathsf{C}(s_1, X[i])$'s are distinct and since f is randomly sampled for fresh queries, the above happens with exponentially small probability. Case (b) is symmetrical. The proof of Theorem 5 is detailed in **??**.

The general case of κ-calls (Theorem 6) is a syntactic generalization of the proof of Theorem 5: the oracle (O, R) is a straightforward extension of the ones used to show Theorem 5. Here as well it is crucial to show that it is hard for an adversary A having only non-adaptive access to the challenge oracle f in the naPRF game to find some $(\vec{s}, \vec{k})$ such that $\mathsf{F}((\vec{s}, \vec{k}), X) = Y$ where for $\vec{k} = (k_1, \ldots, k_\kappa)$ there exists some $i \in [\kappa]$ for which $k_i = f$ for. This requires us to union bound over κ many cases instead of just two cases (case (a) and (b)) for the two-call case. Setting $l = |X|$ large enough suffices to cover the union bound.

2.8.2 Challenges to Ruling out General $O(1)$-call Constructions

In this section, we describe two constructions which make more than one call to the underlying naPRF, for which natural extensions of the oracle from Section 2.4.2 allow to break both PRF- and naPRF-security. While it is unlikely that these constructions will admit a security proof, we want to highlight that ruling them out will require new

techniques (or at least new designs of separation oracles). This also provides some justification to the limited-looking scope of our results. We provide a two-call and a four-call construction in this section.

Two-call Combiner Construction

Let H be a function family on n-bit seeds mapping n bits to n bits. Consider the following two-call construction $\mathsf{F}_2[\mathsf{H}]$ on $2n$-bit seeds mapping n-bits to n-bits: For every $(k_1, k_2) \in \{0,1\}^n \times \{0,1\}^n$ and every $x \in \{0,1\}^n$,

$$\mathsf{F}_2((k_1, k_2), x) = \mathsf{H}(k_1, x) \oplus \mathsf{H}(k_2, b^n) \ ,$$

where $b = x[0]$. This construction can be seen as the parallel (i.e., xor) composition of the one-call, pre-processing-only construction (Section 2.1) with itself, albeit with different pre-processing functions for each call. A peculiar property of the construction F_2 is that the second call to H (on key k_2) is either on input 0^n or 1^n even when the inputs x for F_2 are n-bit strings. We show how a non-adaptive adversary can exploit this property to break the naPRF-security of H relative to a natural extension of (O, R) from Section 2.4.2.

Oracle (O, R). Recall the oracle (O, R) from Section 2.4.2 where O embeds an information-theoretically secure naPRF, and R_1 and R_2 provide adaptive challenges (inputs in the domain of F). The idea behind having R_1 and R_2 was to disallow non-adaptive adversaries from being able to successfully use R_3 to break pseudo-randomness. The oracle R_3 on inputs tuple of challenges $X = X_1||X_2$ (issued by R_1 and R_2) and tuple $Y = Y_1||Y_2$ returns 1 iff there exist some key for the construction that maps X to Y. We extend the oracle (O, R) in the most natural way to the above two-call construction F_2 (formal code in Figure 2.6). We claim that relative to (O, R) there exists a natural adaptive adversary that breaks the PRF-security of F_2. We just focus on showing that naPRF-security of H breaks down relative to (O, R).

Oracle $\mathsf{R}_1(1^n)$:

if $T_1^n = \perp$ **then** $T_1^n \xleftarrow{\$} (\{0,1\}^n)^{[l(n)]}$

return T_1^n

Oracle $\mathsf{R}_3(1^n, X = X_1||X_2, Y = Y_1||Y_2)$:

if $\neg\mathsf{isValid}(1^n, 2l, X, Y)$ **then return** $\perp$

if $X_1 \neq T_1^n \vee X_2 \neq T_2^n[Y_1]$ **then return** $\perp$

if $\exists (k_1, k_2) \in \{0,1\}^n \times \{0,1\}^n$:

$\mathsf{F}_2^{\mathsf{H}^{\mathsf{O}}}((k_1, k_2), X) = Y$ **then return** 1

return $\perp$

Oracle $\mathsf{R}_2(1^n, X, Y)$:

if $\neg\mathsf{isValid}(1^n, l, X, Y)$ **then return** 1

if $X \neq T_1^n$ **then return** $\perp$

if $T_2^n[Y] = \perp$ **then**

$T_2^n[Y] \xleftarrow{\$} (\{0,1\}^n \setminus T_1^n)^{[l(n)]}$

return $T_2^n[Y]$

Proc. $\mathsf{isValid}(1^n, t, X, Y)$

if $X \notin (\{0,1\}^n)^{[t]}$ **then return** 0

if $Y \notin (\{0,1\}^n)^t$ **then return** 0

return 1

Figure 2.6: Description of oracle R for the two-call construction F_2 in Section 2.8.2. Oracles R_1 and R_2 samples random n-bit tuples without replacement from $\{0,1\}^n$, the subroutine $\mathsf{isValid}$ performs sanity checks on the lengths of its inputs X and Y and the oracle R_3 returns 1 if it finds a pair (k_1, k_2) that maps X to Y under F. T_1 and T_2 are internal data-structures maintained by R for book-keeping.

naPRF-security of H relative to (O, R). Recall that in the naPRF-security-game for H relative to (O, R), an adversary $A^{(\mathsf{O},\mathsf{R},h)}$ can make adaptive queries to (O, R) but only non-adaptive queries to the challenge oracle h where h is sampled uniformly from either $\mathsf{Funcs}(n, n)$ or H^{O}. Consider the case when $h \xleftarrow{\$} \mathsf{H}$, that is, there exists some $k^* \in \{0,1\}^n$ such that $h(\cdot) = \mathsf{H}(k^*, \cdot)$. To be able to use R successfully, A would need to compute the outputs $Y = Y_1||Y_2$ (under F_2) for adaptive challenges $X = X_1||X_2$ (issued by R) by mimicking one of the two calls in F_2 using h, while only having non-adaptive access to h. This, in fact, is possible due to the peculiar property of the construction F_2 described above, let us explain. A can obtain $h(0^n)$ and $h(1^n)$ (non-adaptive queries are sufficient). Then for some k_1 of its choice, A can compute the outputs $Y = \mathsf{F}_2((k_1, k^*), X)$ for adaptively chosen challenges X. For this it requires adaptive access to $\mathsf{O}(k_1, \cdot)$ but

only non-adaptive access to h. The oracle R_3 always returns 1 for such Y computed by A. However, note that when $h \xleftarrow{\$} \mathsf{Funcs}(n,n)$ the above strategy of computing Y would result in R_3 outputting 1 only when $h(b^n) = \mathsf{O}(k, b^n)$ for both $b \in \{0,1\}$ and some $k \in \{0,1\}^n$. For a particular k, this happens with probability $1/2^{2n}$ (over the randomness of sampling O_k), by a union bound over 2^n possible k's we can show that R_3 returns 1 with only $1/2^n$ probability, ensuring A can break the non-adaptive security of H.

We note that the non-adaptive attack described above is very similar to the one mentioned in Section 2.1 for the one-call pre-processing-only construction $\mathsf{F}[\mathsf{C},\mathsf{H}] = \mathsf{H}(k, \mathsf{C}(s,x))$. The major difference is that for pre-processing-only construction, the attack was possible because of the existence of negligible fraction of "bad seeds" in an actually universal hash family C. But, here the second call can be viewed as being pre-processed by a non-universal family $\mathsf{C}^*(s,x) = x[0]$, for which all seeds are "bad". We note that the one-call construction with C^* can trivially be ruled out but for the two-call case such trivial approach does not seem to help.

Four-call Construction

In this section, we present a general family of constructions for which our natural extensions of (O,R) from Section 2.4.2 would fail to provide separations. At a high level, take any $(q \geq 2)$-call construction F where all the q calls to H are on independent and uniform chosen keys $(k_1, \ldots, k_q)$, sampled as part of the key for the construction. Now, construct a family $\tilde{\mathsf{F}}$ that makes $2q$-calls where the additional q calls are used to first generate the q keys $(k_1, \ldots, k_q)$ from a short key k on some a-priori fixed inputs, e.g., $k_i = \mathsf{H}(k,i)$. We show that this allows a non-adaptive adversary to break the non-adaptive security of H relative to a natural extension of the oracle from Section 2.4.2 to the $2q$-call construction. For simplicitly we only describe the construction for $q = 2$, i.e., a four-call construction.

Let H be a function family on n-bit seeds mapping n bits to n bits. Let $\mathsf{F}[\mathsf{H}]$ be any two-call construction on $\sigma + 2n$-bit seeds mapping n bits to n bits, where the two calls to H are on independent keys (k_1, k_2) sampled during key generation.

Consider the following four-call construction $\tilde{\mathsf{F}}[\mathsf{H}]$ on $\sigma + n$-bit seeds mapping n bits to n bits such that for all $(s, k) \in \{0,1\}^\sigma \times \{0,1\}^n$ and $x \in \{0,1\}^n$,

$$\tilde{\mathsf{F}}((s,k),x) = \mathsf{F}((s,k_1,k_2),x) \ ,$$

where $k_i = \mathsf{H}(k,i)$.

As in the two-call case, we show that a certain property of the construction $\tilde{\mathsf{F}}$ allows a non-adaptive adversary to break H relative to natural generalizations of the oracle (O,R) from Section 2.4.2. At a high level, a non-adaptive adversary $A^{(\mathsf{O},\mathsf{R},h)}$ can use its challenge oracle h to compute $k_1 = h(1)$ and $k_2 = h(2)$ using only non-adaptive queries. Then for some randomly chosen s, A can compute $Y = \mathsf{F}((s,k_1,k_2),X)$ for any adaptive challenges X using adaptive access to $O(k_1,\cdot)$ and $O(k_2,\cdot)$. When $h \xleftarrow{\$} \mathsf{H}$, A outputs 1 with probability 1 as there exists some k^* such that $\mathsf{H}(k^*,\cdot) = h$ for which Y computed by A equals $\tilde{\mathsf{F}}(s,k^*)(X)$. However, when $h \xleftarrow{\$} \mathsf{Funcs}(n,n)$ the probability that A outputs 1 can be upper bounded by the probability that there exists some k for which $h(i) = \mathsf{O}(k,i)$ for $i \in [2]$. As seen in the two-call case this probability can be upper bounded by $1/2^n$ which implies that A breaks the naPRF-security of H. The main ideas are similar to the two-call case and we skip describing the oracles and the non-adaptive adversary more formally here.

In conclusion, the constructions described in this section highlight that the natural class of oracles R that break the adaptive security of the construction (e.g., F_2 or $\tilde{\mathsf{F}}$) by finding keys consistent to some tuple (X,Y) of input and output pairs, in some cases allow even non-adaptive adversaries to, additionally, break the underlying naPRF H.

This provides some insight into why ruling out constructions that make more than one call construction is significantly more challenging. We believe tackling the general case of ruling out all $O(1)$ calls constructions will, at the minimum, require a new approach for designing separations oracles which perhaps will need to be significantly different from natural approaches known in the literature.

2.9 Proofs from Section 2.4

2.9.1 Proof of Lemma 1

Clearly, β and δ are negligible whenever α is negligible and $c = O(1)$. Now suppose for contradiction that for the above defined β and δ,

$$\Pr_s[s \text{ is not } \beta\text{-sparse}] > \delta \ .$$

That is, with probability at least δ for the randomly sampled s there exists some w_s with more than $\beta \cdot 2^n$ pre-images under C_s. We sample c inputs x_i at random without replacement and compute the probability that all map to $\mathsf{C}(s, x_i) = w_s$.

With probability at least β the first input x_1 will be mapped to w_s, with probability at least $\frac{\beta \cdot 2^n - 1}{2^n - 1}$ the second input x_2 will be mapped to w_s and so on. That is,

$$\Pr_{x_1,\ldots,x_c,s}[\forall i \in [c] : \mathsf{C}(s, x_i) = w_s] > \delta \cdot \prod_{i=1}^{c} \frac{\beta \cdot 2^n - (i-1)}{2^n - (i-1)}$$

Since, $\beta = \max(\alpha^{1/2c}, 2c/2^n)$, each of the numerator $\beta 2^n - (i-1)$ is lower bounded by $\beta 2^n/2$. The denominators $2^n - (i-1)$ are upper bounded by 2^n. Therefore,

$$\Pr_{x_1,\ldots,x_c,s}[\forall i \in [c] : \mathsf{C}(s, x_i) = w_s] > \delta \cdot \left(\frac{\beta \cdot 2^n}{2 \cdot 2^n}\right)^c \geq \delta \cdot \frac{\beta^c}{2^{c-1}} \ ,$$

We know that $\delta \frac{\beta^c}{2^{c-1}} \geq \alpha$. Therefore what we have shown is that,

$$\Pr_{x_1,\ldots,x_c,s}[\mathsf{C}(s, x_1), \ldots, \mathsf{C}(s, x_c)] > \alpha \ ,$$

then by averaging we know that there must exist some distinct $x_1, \ldots, x_c$ for which,

$$\Pr_s[\mathsf{C}(s, x_1), \ldots, \mathsf{C}(s, x_c)] > \alpha \ ,$$

which contradicts the (α, c)-universality of C.

2.9.2 Proof of Lemma 2

Before we describe the proof of Lemma 2, we write down the following simple but useful claim. Informally, it states that for every tuple X, most seeds s of C are "good" in the sense of Definition 7.

Claim 8 *For $n, l \in \mathbb{N}$ and any $X = X_1||X_2 \in (\{0,1\}^n)^{[2l]}$ where each X_i are of l-length tuples,*

$$\Pr_s[s \notin \mathsf{Good}_C(\beta, X)] \leq \delta + \alpha \cdot \binom{2l}{c} \ ,$$

where β, δ are as defined in Lemma 1.

Remark 6 *We remark that the right hand side in Claim 8 is negligible only when $c = O(1)$ and this is why our results are restricted to $c = O(1)$.*

Proof sketch of Claim 8. From Definition 7, we know that $s \notin \mathsf{Good}_C(\beta, X)$ if either s is not β-sparse or s leads to c-way collisions among X. The probability of former was analyzed in Lemma 1 and the later follows directly from union bound and (α, c)-universality of C.

Now we are set to prove Lemma 2.

Proof of Lemma 2. Let us fix some $n \in \mathbb{N}$. Let the inputs received by A from R_1 and R_2 be X_1 and X_2 respectively and let $Y_1 = f(X_1)$ and $Y_2 = f(X_2)$ where f is the challenge oracle given to A.

First, consider the real world where A is interacting with $f \xleftarrow{\$} \mathsf{F}$ (or implicitly $f = \mathsf{F}((s^*, k^*), \cdot)$ for randomly chosen (s^*, k^*)). Here, A will output 1 whenever R_3 outputs 1. For any $X = X_1||X_2$ and $Y = Y_1||Y_2$, R_3 outputs 1 if there exists (s, k) such that (1) $\mathsf{F}((s, k), X) = Y$ and (2) $s \in \mathsf{Good}_{\mathsf{F}}(\beta, X, Y)$. Clearly, (s^*, k^*) satisfies the condition (1). Now, let us upper bound the probability that $s^* \notin \mathsf{Good}_{\mathsf{F}}(\beta, X, Y)$. From Definition 8 we know that $s^* \notin \mathsf{Good}_{\mathsf{F}}(\beta, X, Y)$ if either (a) $s^* \notin \mathsf{Good}_{\mathsf{C}}(\beta, X)$ or if (b) the number of $Y[i]$'s that are $1/2$-bad w.r.t. $(s, X[i])$ are more than $l/2c$. We first analyze (a). We can clearly bound the probability that $s^* \notin \mathsf{Good}_{\mathsf{C}}(\beta, X)$ using Claim 8. However, note that X in Claim 8 needs to be fixed before sampling s^* but in our case $X = X_1||X_2$ are adaptively sampled. This is not an issue because our X_i's are random and sampled independently which allows us to alternatively view the experiment as first sampling X_1, X_2 and then sampling s^* independently of them. Next, we analyze part (b). Recall that F is $(1 - 1/4c)$-unbiased. Therefore for l (which is set to be $\omega(\sigma + n)$) we know that there exist some negligible function α' such that the probability that for $Y = \mathsf{F}((s^*, k), X)$ there are more than $l/2c$ $Y[i]$'s that are $1/2$-bad w.r.t. $(s, X[i])$ is at most α' (note that it is $l/2c$ because $|X| = 2l$). Therefore, combining cases (a) and (b) (and Claim 8),

$$\mathsf{Pr}[s^* \notin \mathsf{Good}_{\mathsf{F}}(\beta, X, Y)] \leq 2^{c-1}\sqrt{\alpha(n)} + \alpha \cdot \binom{2l(n)}{c} + \alpha' \ . \tag{2.19}$$

Now, we consider the case that A is interacting with $f \xleftarrow{\$} \mathsf{Funcs}(n, m)$. The probability that R_3 outputs 1 on $(X = X_1||X_2, Y = Y_1||Y_2)$ is clearly upper bounded by the probability that there exists some (s, k) such that $\mathsf{F}((s, k), X_1) = Y_1$. For X_1, consider the set $\mathcal{Y} = \{\mathsf{F}((s, k), X_1) : (s, k) \in \{0, 1\}^{\sigma+\kappa}\}$. Clearly, size of $\mathcal{Y}$ is upper bounded by $2^{\sigma+n}$. Since X_1 is sampled without replacement (hence all $X_1[i]$'s are distinct) and hence for $Y_1 = f(X_1)$, we can view each of the $Y_1[i]$'s to be sampled independently at random.

Therefore, we can bound the required probability as the probability as

$$\Pr_{Y \xleftarrow{\$} (\{0,1\}^n)^l}[Y \in \mathcal{Y}] < \frac{2^{\sigma+n}}{2^{ml}} \ . \tag{2.20}$$

Combining Equation 2.19 and 2.20, we have

$$\mathsf{Adv}^{\mathsf{rel-prf}}_{A,\mathsf{F},(\mathsf{O},\mathsf{R})}(n) \geq 1 - 2^{c-1}\sqrt{\alpha(n)} - \alpha \cdot \binom{2l(n)}{c} - \alpha' - \frac{2^{\sigma+n}}{2^{ml}} \ .$$

Since, $c = O(1)$, $l = \omega(\sigma + n)$ is a polynomial, and α and α' are negligible functions, we can conclude that $\mathsf{Adv}^{\mathsf{rel-prf}}_{A,\mathsf{F},(\mathsf{O},\mathsf{R})}(n)$ as stated above is non-negligible.

Chapter 3

Public-seed PRFs to Public-seed PRPs via Feistel

The main result of this chapter builds a public-seed pseudorandom permutation (psPRP) from a public-seed pseudorandom function (psPRF) (also known as UCE in the literature). Towards this Section 3.1 first proposes a general framework for public-seed pseudorandom notions, and Section 3.2 puts this to use to provide general reduction theorems between pairs of such primitives, and defines in particular CP-sequential indifferentiability. Section 3.3 describes the main result on the public-seed pseudorandomness of 5-round Feistel followed by the proof of its CP-sequential indifferentiability in Section 3.4 upon which the main result is based on.

Notational preliminaries. Throughout this chapter, we denote by $\mathsf{Funcs}(X, Y)$ the set of functions $X \to Y$, and in particular use the shorthand $\mathsf{Funcs}(m, n)$ whenever $X = \{0,1\}^m$ and $Y = \{0,1\}^n$. We also denote by $\mathsf{Perms}(X)$ the set of permutations on the set X, and analogously, $\mathsf{Perms}(n)$ denotes the special case where $X = \{0,1\}^n$. We say that a function $f : \mathbb{N} \to \mathbb{R}_{\geq 0}$ is *negligible* if for all $c \in \mathbb{N}$, there exists a λ_0 such that

$f(\lambda) \leq \lambda^{-c}$ for all $\lambda \geq \lambda_0$.

Our security definitions and proofs will often use games, as formalized by Bellare and Rogaway [97]. Typically, our games will have boolean outputs – that is, either true or false – and we use the shorthand $\mathsf{Pr}[\mathsf{G}]$ to denote the probability that a certain game outputs the value true, or occasionally 1 (when the output is binary, rather than boolean).

3.1 Public-seed Pseudorandomness

We present a generalization of the UCE notion [37], which we term *public-seed pseudorandomness*. We apply this notion to define psPRPs as a special case, but the general treatment will be useful to capture transformations between UCEs and psPRPs in Section 3.2 via one single set of theorems.

3.1.1 Ideal Primitives and their Implementations

We begin by formally defining *ideal primitives* using notation inspired by [98, 99].

Ideal primitives. An *ideal primitive* is a pair $\mathbf{I} = (\Sigma, \mathcal{T})$, where $\Sigma = \{\Sigma_\lambda\}_{\lambda\in\mathbb{N}}$ is a family of sets of functions (such that all functions in Σ_λ have the same domain and range), and $\mathcal{T} = \{\mathcal{T}_\lambda\}_{\lambda\in\mathbb{N}}$ is a family of probability distributions, where $\mathcal{T}_\lambda$'s range is a subset of Σ_λ for all $\lambda \in \mathbb{N}$. The ideal primitive $\mathbf{I}$, once the security parameter λ is fixed, should be thought of as an oracle that initially samples a function I as its initial state according to $\mathcal{T}_\lambda$ from Σ_λ. We denote this sampling as $I \xleftarrow{\$} \mathbf{I}_\lambda$. Then, $\mathbf{I}$ provides access to I via queries, that is, on input $\mathbf{x}$ it returns $I(\mathbf{x})$.[1]

[1]The reader may wonder whether defining Σ is necessary, but this will allow us to enforce a specific format on valid implementations below.

Examples. We give a few examples of ideal primitives using the above notation. In particular, let $\kappa, m, n : \mathbb{N} \to \mathbb{N}$ be functions.

Example 1 *The random function* $\mathbf{R}_{m,n} = (\Sigma^{\mathbf{R}}, \mathcal{T}^{\mathbf{R}})$ *is such that for* $\lambda \in \mathbb{N}$, $\Sigma^{\mathbf{R}}_\lambda = \mathsf{Funcs}(m(\lambda), n(\lambda))$, *and* $\mathcal{T}^{\mathbf{R}}_\lambda$ *is the uniform distribution on* $\Sigma^{\mathbf{R}}_\lambda$. *We also define* $\mathbf{R}_{*,n}$ *to be the same for* $\mathsf{Funcs}(*, n(\lambda))$, *that is, when the domain is extended to arbitrary length input strings.*[2]

Example 2 *The random permutation* $\mathbf{P}_n = (\Sigma^{\mathbf{P}}, \mathcal{T}^{\mathbf{P}})$ *is such that for all* $\lambda \in \mathbb{N}$,

$$\Sigma^{\mathbf{P}}_\lambda = \Big\{ P : \{+,-\} \times \{0,1\}^{n(\lambda)} \to \{0,1\}^{n(\lambda)} \mid \\ \exists \pi \in \mathsf{Perms}(n(\lambda)) : \; P(+,x) = \pi(x), P(-,x) = \pi^{-1}(x) \Big\} ,$$

and moreover, $\mathcal{T}^{\mathbf{P}}_\lambda$ *is the uniform distribution on* $\Sigma^{\mathbf{P}}_\lambda$.

Example 3 *The ideal cipher* $\mathbf{IC}_{\kappa,n} = (\Sigma^{\mathbf{IC}}, \mathcal{T}^{\mathbf{IC}})$ *is such that*

$$\Sigma^{\mathbf{IC}}_\lambda = \Big\{ E : \{0,1\}^{\kappa(\lambda)} \times \{+,-\} \times \{0,1\}^{n(\lambda)} \to \{0,1\}^{n(\lambda)} \mid \\ \forall k \in \{0,1\}^{\kappa(\lambda)} \exists \pi_k \in \mathsf{Perms}(n(\lambda)) : \; E(k,+,x) = \pi_k(x), E(k,-,x) = \pi_k^{-1}(x) \Big\} ,$$

and $\mathcal{T}^{\mathbf{IC}}_\lambda$ *is the uniform distribution on* $\Sigma^{\mathbf{IC}}_\lambda$.

Efficiency considerations. Usually, for an ideal primitive $\mathbf{I} = (\Sigma, \mathcal{T})$, the bit-size of the elements of Σ_λ grows exponentially in λ, and thus one would not implement a primitive $\mathbf{I}$ by sampling I from Σ_λ, but rather using techniques such as lazy sampling. An implementation of a primitive $\mathbf{I}$ is a *stateful* randomized PPT algorithm A such that $A(1^\lambda, \cdot)$ behaves as $I \xleftarrow{\$} \mathbf{I}_\lambda$ for all $\lambda \in \mathbb{N}$. We say that $\mathbf{I}$ is *efficiently implementable* if such an A exists. All the above examples – $\mathbf{R}_{m,n}, \mathbf{R}_{*,n}, \mathbf{P}_n$, and $\mathbf{IC}_{\kappa,n}$ – are efficiently implementable as long as m, n, κ are polynomially bounded functions.

[2]Note that this requires some care, because Σ_λ is now uncountable, and thus sampling from it requires a precise definition. We will not go into formal details, similar to many other papers, but it is clear that this can easily be done.

MAIN $\mathsf{psPR}^{S,D}_{\mathsf{F},\mathbf{I}}(\lambda)$:	ORACLE $\mathcal{O}(i,\mathbf{x})$:
$(1^n, t) \xleftarrow{\$} S(1^\lambda, \varepsilon)$	**if** $b = 1$ **then**
$b \xleftarrow{\$} \{0,1\}$	**return** $\mathsf{F.Eval}(1^\lambda, k_i, \mathbf{x})$
$k_1, \ldots, k_n \xleftarrow{\$} \mathsf{F.Kg}(1^\lambda)$	**else**
$f_1, \ldots, f_n \xleftarrow{\$} \mathbf{I}_\lambda$	**return** $f_i(\mathbf{x})$
$L \xleftarrow{\$} S^{\mathcal{O}}(1^\lambda, t)$	
$b' \xleftarrow{\$} D(1^\lambda, k_1, \ldots, k_n, L)$	
return $b' = b$	

Figure 3.1: Game psPR used to define pspr-security for a primitive F that is Σ-compatible with $\mathbf{I}$. Here, S is the source and D is the distinguisher. Recall that the notation $f \xleftarrow{\$} \mathbf{I}_\lambda$ indicates picking a function from Σ_λ using $\mathcal{T}_\lambda$.

Σ-compatible function families. A *function family* $\mathsf{F} = (\mathsf{Kg}, \mathsf{Eval})$ consists of a *key (or seed) generation algorithm* F.Kg and an *evaluation algorithm* F.Eval. In particular, F.Kg is a randomized algorithm that on input the unary representation of the security parameter λ returns a *key* k, and we let $[\mathsf{F.Kg}(1^\lambda)]$ denote the set of all possible outputs of $\mathsf{F.Kg}(1^\lambda)$. Moreover, F.Eval is a deterministic algorithm that takes three inputs; the security parameter in unary form 1^λ, a key $k \in [\mathsf{F.Kg}(1^\lambda)]$ and a query $\mathbf{x}$ such that $\mathsf{F.Eval}(1^\lambda, k, \cdot)$ implements a function that maps queries $\mathbf{x}$ to $\mathsf{F.Eval}(1^\lambda, k, \mathbf{x})$. We say that F is *efficient* if both Kg and Eval are polynomial-time algorithms.

Definition 1 (Σ-compatibility) *A function family* F *is* Σ*-compatible with* $\mathbf{I} = (\Sigma, \mathcal{T})$ *if* $\mathsf{F.Eval}(1^\lambda, k, \cdot) \in \Sigma_\lambda$ *for all* $\lambda \in \mathbb{N}$ *and* $k \in [\mathsf{F.Kg}(1^\lambda)]$.

3.1.2 Public-seed Pseudorandomness, psPRPs, and Sources

We now define a general notion of public-seed pseudorandom implementations of ideal primitives.

The general definition. Let $\mathsf{F} = (\mathsf{Kg}, \mathsf{Eval})$ be a function family that is Σ-compatible with an ideal primitive $\mathbf{I} = (\Sigma, \mathcal{T})$. Let S be an adversary called the *source* and D an adversary called the *distinguisher*. We associate to them, F and $\mathbf{I}$, the game $\mathsf{psPR}^{S,D}_{\mathsf{F},\mathbf{I}}(\lambda)$ depicted in Fig. 3.1. The source initially chooses the number of keys n. Then, in the second stage, it is given access to an oracle $\mathcal{O}$ and we require any query $(i, \mathbf{x})$ made to this oracle be valid, that is, $\mathbf{x}$ is a valid query for any $f_i \in \Sigma_\lambda$ and $i \in [n]$, for n output by the first stage of the source. When the challenge bit $b = 1$ ("real") the oracle responds via $\mathsf{F.Eval}$ under the key k_i ($\mathsf{F.Eval}(1^\lambda, k_i, \cdot)$) that is chosen by the game and *not* given to the source. When $b = 0$ ("ideal") it responds via f_i where $f_i \xleftarrow{\$} \mathbf{I}_\lambda$. After its interaction with the oracle $\mathcal{O}$, the source S communicates the *leakage* $L \in \{0,1\}^*$ to D. The distinguisher is given access to the keys $k_1, \ldots, k_n$ and must now guess $b' \in \{0,1\}$ for b. The game returns true iff $b' = b$ and we describe the pspr-advantage of (S, D) for $\lambda \in \mathbb{N}$ as

$$\mathsf{Adv}^{\mathsf{pspr}[\mathbf{I}]}_{\mathsf{F},S,D}(\lambda) = 2\Pr\left[\mathsf{psPR}^{S,D}_{\mathsf{F},\mathbf{I}}(\lambda)\right] - 1\ . \tag{3.1}$$

In the following, we are going to use the shorthands $\mathsf{UCE}[m,n]$ for $\mathsf{pspr}[\mathbf{R}_{m,n}]$, $\mathsf{UCE}[n]$ for $\mathsf{pspr}[\mathbf{R}_{*,n}]$, and $\mathsf{psPRP}[n]$ for $\mathsf{pspr}[\mathbf{P}_n]$.

Note that our security game captures the multi-key version of the security notions, also considered in past works on UCE, as it is not known to be implied by the single-key version, which is recovered by having the source initially output $n = 1$.

Restricting sources. One would want to define F as secure if $\mathsf{Adv}^{\mathsf{pspr}[\mathbf{I}]}_{\mathsf{F},S,D}(\lambda)$ is negligible in λ for all polynomial time sources S and distinguishers D. However, as shown already in the special case of UCEs [37], this is impossible, as one can always easily construct (at least for non-trivial $\mathbf{I}$'s) a simple source S which leaks the evaluation of $\mathcal{O}$ on a given point, and D can check consistency given k.

Therefore to obtain meaningful and non-empty security definitions we restrict the

MAIN $\mathsf{Pred}^{P}_{\mathbf{I},S}(\lambda)$:	MAIN $\mathsf{Reset}^{R}_{\mathbf{I},S}(\lambda)$:
$\mathsf{done} \leftarrow \mathsf{false}; Q \leftarrow \emptyset$; $(1^n, t) \xleftarrow{\$} S(1^\lambda, \varepsilon)$	$\mathsf{done} \leftarrow \mathsf{false}$; $(1^n, t) \xleftarrow{\$} S(1^\lambda, \varepsilon)$
$f_1, \ldots, f_n \xleftarrow{\$} \mathbf{I}_\lambda$	$f_1^0, f_1^1, \ldots, f_n^0, f_n^1 \xleftarrow{\$} \mathbf{I}_\lambda$
$L \xleftarrow{\$} S^{\mathcal{O}}(1^\lambda, t); \mathsf{done} \leftarrow \mathsf{true}$	$L \xleftarrow{\$} S^{\mathcal{O}}(1^\lambda, t); \mathsf{done} \leftarrow \mathsf{true}$
$Q' \xleftarrow{\$} P^{\mathcal{O}}(1^\lambda, 1^n, L)$	$b \xleftarrow{\$} \{0,1\}; b' \xleftarrow{\$} R^{\mathcal{O}}(1^\lambda, 1^n, L)$
return $(Q \cap Q' \neq \emptyset)$	**return** $b' = b$
ORACLE $\mathcal{O}(i, \mathbf{x})$:	ORACLE $\mathcal{O}(i, \mathbf{x})$:
if $\neg\mathsf{done}$ **then** $Q \leftarrow Q \cup \{\mathbf{x}\}$	**if** $\neg\mathsf{done}$ **then return** $f_i^0(\mathbf{x})$
return $f_i(\mathbf{x})$	**else return** $f_i^b(\mathbf{x})$

Figure 3.2: Games Pred and Reset are used to define the unpredictability and reset-security of the source S respectively against the ideal primitive $\mathbf{I}$. Here, S is the source, P is the predictor and R is the reset adversary.

considered sources to some class $\mathcal{S}$, without restricting the distinguisher class. Consequently, we denote by $\mathsf{psPR}[\mathbf{I}, \mathcal{S}]$ the security notion that asks $\mathsf{Adv}^{\mathsf{pspr}[\mathbf{I}]}_{\mathsf{F},S,D}(\lambda)$ to be negligible for all polynomial time distinguishers D and all sources $S \in \mathcal{S}$. Following [37], we also use $\mathsf{psPR}[\mathbf{I}, \mathcal{S}]$ to denote the set of F's which are $\mathsf{psPR}[\mathbf{I}, \mathcal{S}]$-secure. Note that obviously, if $\mathcal{S}_1 \subseteq \mathcal{S}_2$, then $\mathsf{psPR}[\mathbf{I}, \mathcal{S}_2] \subseteq \mathsf{psPR}[\mathbf{I}, \mathcal{S}_1]$ where $\mathcal{S}_1$ and $\mathcal{S}_2$ are source classes for the ideal primitive $\mathbf{I}$. We will use the shorthands $\mathsf{psPRP}[n, \mathcal{S}]$ for $\mathsf{psPR}[\mathbf{P}_n, \mathcal{S}]$ and $\mathsf{UCE}[m, n, \mathcal{S}]$ for $\mathsf{psPR}[\mathbf{R}_{m,n}, \mathcal{S}]$, where $m = *$ if the domain is unbounded.

Below, we discuss two important classes of restrictions, which are fundamental for the remainder of this paper – unpredictable and reset-secure sources.

Unpredictable sources. Let S be a source. Consider the game $\mathsf{Pred}^{P}_{\mathbf{I},S}(\lambda)$ of Fig. 3.2 associated to S and an adversary P called the predictor. Given the leakage, the latter

outputs a set Q'. It wins if this set contains any $\mathcal{O}$-query of the source. For $\lambda \in \mathbb{N}$ we let

$$\mathsf{Adv}^{\mathsf{pred}[\mathbf{I}]}_{S,P}(\lambda) = \Pr\left[\mathsf{Pred}^{P}_{\mathbf{I},S}(\lambda)\right] \ . \tag{3.2}$$

We say that P is a *computational predictor* if it is polynomial time, and it is a *statistical* predictor if there exists polynomials q, q' such that for all $\lambda \in \mathbb{N}$, predictor P makes at most $q(\lambda)$ oracle queries and outputs a set Q' of size at most $q'(\lambda)$ in game $\mathsf{Pred}^{P}_{\mathbf{I},S}(\lambda)$. We stress that in this case the predictor need not be polynomial time, even though it makes a polynomial number of queries. We say S is *computationally unpredictable* if $\mathsf{Adv}^{\mathsf{pred}[\mathbf{I}]}_{S,P}(\lambda)$ is negligible for all computational predictors P. We say S is *statistically unpredictable* if $\mathsf{Adv}^{\mathsf{pred}[\mathbf{I}]}_{S,P}(\lambda)$ is negligible for all statistical predictors P. We let $\mathcal{S}^{\mathsf{cup}}$ be the class of all polynomial time, computationally unpredictable sources and $\mathcal{S}^{\mathsf{sup}} \subseteq \mathcal{S}^{\mathsf{cup}}$ be the class of all polynomial time statistically unpredictable sources.[3]

Reset-secure sources. Let S be a source. Consider the game $\mathsf{Reset}^{R}_{\mathbf{I},S}(\lambda)$ of Fig. 3.2 associated to S and an adversary R called the reset adversary. The latter wins if given the leakage L it can distinguish between f^0 used by the source S and an independent f^1 where $f^0, f^1 \xleftarrow{\$} \mathbf{I}_\lambda$. For $\lambda \in \mathbb{N}$ we let

$$\mathsf{Adv}^{\mathsf{reset}[\mathbf{I}]}_{S,R}(\lambda) = 2\Pr\left[\mathsf{Reset}^{R}_{\mathbf{I},S}(\lambda)\right] - 1 \ . \tag{3.3}$$

We say that R is a *computational reset adversary* if it is polynomial time, and it is a statistical reset adversary if there exists a polynomial q such that for all $\lambda \in \mathbb{N}$, reset adversary R makes at most $q(\lambda)$ oracle queries in game $\mathsf{Reset}^{R}_{\mathbf{I},S}(\lambda)$. We stress that in this case the reset adversary need not be polynomial time. We say S is *computationally reset-secure* if $\mathsf{Adv}^{\mathsf{reset}[\mathbf{I}]}_{S,R}(\lambda)$ is negligible for all computational reset adversaries R. We say S is *statistically reset-secure* if $\mathsf{Adv}^{\mathsf{reset}[\mathbf{I}]}_{S,R}(\lambda)$ is negligible for all statistical reset adversaries

[3] We note that computational unpredictability is only meaningful for sufficiently restricted classes of sources or in ideal models, as otherwise security against $\mathcal{S}^{\mathsf{cup}}$ is not achievable assuming IO, using essentially the same attack as [39].

MAIN $\mathsf{CP}[\mathbf{I} \to \mathbf{J}]^A_{\mathsf{M},\mathsf{Sim}}(\lambda)$:	ORACLE $\mathsf{Func}(\mathbf{x})$:	ORACLE $\mathsf{Prim}(\mathbf{u})$:
$b \xleftarrow{\$} \{0,1\}; f \xleftarrow{\$} \mathbf{I}_\lambda; g \xleftarrow{\$} \mathbf{J}_\lambda$	**if** $b = 1$ **then**	**if** $b = 1$ **then**
$\mathsf{st} \xleftarrow{\$} A_1^{\mathsf{Func}}(1^\lambda)$	**return** $\mathsf{M}^f(\mathbf{x})$	**return** $f(\mathbf{u})$
$b' \xleftarrow{\$} A_2^{\mathsf{Prim}}(1^\lambda, \mathsf{st})$	**else**	**else**
return $b' = b$	**return** $g(\mathbf{x})$	**return** $\mathsf{Sim}^g(\mathbf{u})$

Figure 3.3: Game CP used to define cpi-security for a construction M implementing the primitive **J** using primitive **I**. Here, Sim is the simulator and $A = (A_1, A_2)$ is the two-stage distinguisher.

R. We let $\mathcal{S}^{\mathsf{crs}}$ be the class of all polynomial time, computationally reset-secure sources and $\mathcal{S}^{\mathsf{srs}} \subseteq \mathcal{S}^{\mathsf{crs}}$ the class of all polynomial time statistically reset-secure sources.

Relationships. For the case of psPRPs, we mention the following fact, which is somewhat less obvious than in the UCE case, and in particular only holds if the permutation's domain grows with the security parameter.

Proposition 4 *For all $n \in \omega(\log \lambda)$, we have $\mathsf{psPRP}[n, \mathcal{S}^{\star\mathsf{rs}}] \subseteq \mathsf{psPRP}[n, \mathcal{S}^{\star\mathsf{up}}]$ where $\star \in \{\mathsf{c}, \mathsf{s}\}$.*

Proof: [Sketch] In the reset secure game, consider the event that R queries its oracle $\mathcal{O}$ on input (i, σ, x) which was queried by S already as an $\mathcal{O}(i, \sigma, x)$ query, or it was the answer to a query $\mathcal{O}(i, \overline{\sigma}, y)$. Here (like elsewhere in the paper), we use the notational convention $\overline{+} = -$ and $\overline{-} = +$. The key point here is proving that as long as this bad event does not happen, the $b = 0$ and $b = 1$ case are hard to distinguish. A difference with the UCE case is that due to the permutation property, they will not be perfectly indistinguishable, but a fairly standard (yet somewhat tedious) birthday argument suffices to show that indistinguishability still holds as long as the overall number of $\mathcal{O}$ queries (of S and R) is below $2^{n(\lambda)/2}$, which is super-polynomial for $n(\lambda) = \omega(\log \lambda)$. ■

3.2 Reductions and Indifferentiability

We present general theorems that we will use to obtain reductions between psPRPs and UCEs. Our general notation for public-seed pseudorandom primitives allows us to capture the reductions through two general theorems.

3.2.1 CP-sequential indifferentiability

Indifferentiability was introduced in [36] by Maurer, Renner, and Holenstein to formalize reductions between ideal primitives. Following their general treatment, it captures the fact that a (key-less) construction M using primitive $\mathbf{I}$ (which can be queried by the adversary directly) is *as good* as another primitive $\mathbf{J}$ by requiring the existence of a simulator that can simulate $\mathbf{I}$ consistently by querying $\mathbf{J}$.

Central to this paper is a weakening of indifferentiability that we refer to as *CP-sequential indifferentiability*, where the distinguisher A makes all of its *construction* queries to $\mathsf{M}^{\mathbf{I}}$ (or $\mathbf{J}$) *before* moving to making *primitive queries* to $\mathbf{I}$ (or $\mathsf{Sim}^{\mathbf{J}}$, where Sim is the simulator). Note that this remains a non-trivial security goal, since Sim does not learn the construction queries made by A, but needs to simulate correctly nonetheless. However, the hope is that because A has committed to its queries before starting its interaction with Sim, the simulation task will be significantly easier. (We will see that this is indeed the case.)

More concretely, the notion is concerned with constructions which implement $\mathbf{J}$ from $\mathbf{I}$, and need to at least satisfy the following syntactical requirement.

Definition 2 ($(\mathbf{I} \to \mathbf{J})$-compatibility) *Let $\mathbf{I} = (\mathbf{I}.\Sigma, \mathbf{I}.\mathcal{T})$ and $\mathbf{J} = (\mathbf{J}.\Sigma, \mathbf{J}.\mathcal{T})$ be ideal primitives. A construction M is called $(\mathbf{I} \to \mathbf{J})$-compatible if for every $\lambda \in \mathbb{N}$, and every $f \in \mathbf{I}.\Sigma_\lambda$, the construction M implements a function $x \mapsto \mathsf{M}^f(1^\lambda, x)$ which is in $\mathbf{J}.\Sigma_\lambda$.*

The game CP is described in Fig. 3.3. For ideal primitives $\mathbf{I}, \mathbf{J}$, a two-stage adversary $A = (A_1, A_2)$, an $(\mathbf{I} \to \mathbf{J})$-compatible construction M, and simulator Sim, as well as security parameter $\lambda \in \mathbb{N}$, we define

$$\mathsf{Adv}^{\mathsf{cpi}[\mathbf{I}\to\mathbf{J}]}_{\mathsf{M},\mathsf{Sim},A}(\lambda) = 2 \cdot \Pr\left[\mathsf{CP}[\mathbf{I} \to \mathbf{J}]^{A}_{\mathsf{M},\mathsf{Sim}}(\lambda)\right] - 1 \; . \tag{3.4}$$

We remark that the CP-sequential indifferentiability notion is the exact dual of sequential indifferentiability as introduced by Mandal, Patarin, and Seurin [46], which postpones construction queries *to the end*. As we will show below in Section **??**, there are CP-indifferentiable constructions which are not sequentially indifferentiable in the sense of [46].

Multi-user Indifferentiability. It will also be notationally convenient, for our proofs below, to define a multi-user version of the CP-indifferentiability game, where an additional stage A_0 first chooses the number of instances it intends to attack. This is formalized by the game muCP at the bottom of Fig. 3.4, for which we define analogously the quantity $\mathsf{Adv}^{\mathsf{m-cpi}[\mathbf{I}\to\mathbf{J}]}_{\mathsf{M},\mathsf{Sim},A}(\lambda)$. Then, the following lemma follows by a simple hybrid argument, and its proof is omitted.

Lemma 12 (Single-user to multi-user CP-indifferentiability) *Let* M *be a* $(\mathbf{I} \to \mathbf{J})$*-compatible construction. For all distinguishers* $A = (A_0, A_1, A_2)$*, such that on input* 1^λ *the output of* A_0 *is bounded by* $N(\lambda)$*, there exists a distinguisher* $A' = (A'_1, A'_2)$ *(given explicitly in the proof) such that for all* $\lambda \in \mathbb{N}$*,*

$$\mathsf{Adv}^{\mathsf{m-cpi}[\mathbf{I}\to\mathbf{J}]}_{\mathsf{M},\mathsf{Sim},A}(\lambda) \leq N(\lambda) \cdot \mathsf{Adv}^{\mathsf{cpi}[\mathbf{I}\to\mathbf{J}]}_{\mathsf{M},\mathsf{Sim},A'}(\lambda) \; , \tag{3.5}$$

In particular, if A *is PPT,* $\mathbf{I}$*,* $\mathbf{J}$ *are efficiently implementable, and* M *is also polynomial-time, then* A' *is polynomial time.*

MAIN $\mathsf{muCP}[\mathbf{I} \to \mathbf{J}]^{A}_{\mathsf{M},\mathsf{Sim}}(\lambda)$:
$(1^n, \mathsf{st}_0) \xleftarrow{\$} A_0(1^\lambda)$; $b \xleftarrow{\$} \{0,1\}$
$f_1, \ldots, f_n \xleftarrow{\$} \mathbf{I}_\lambda$
$g_1, \ldots, g_n \xleftarrow{\$} \mathbf{J}_\lambda$
$\mathsf{st}_1 \xleftarrow{\$} A_1^{\mathsf{Func}}(1^\lambda, \mathsf{st}_0)$
$b' \xleftarrow{\$} A_2^{\mathsf{Prim}}(1^\lambda, \mathsf{st}_1)$
return $b' = b$

ORACLE $\mathsf{Func}(i, \mathbf{x})$:
if $b = 1$ **then**
  return $\mathsf{M}^{f_i}(\mathbf{x})$
else
  return $g_i(\mathbf{x})$

ORACLE $\mathsf{Prim}(i, \mathbf{u})$:
if $b = 1$ **then**
  return $f_i(\mathbf{u})$
else
  return $\mathsf{Sim}^{g_i}(\mathbf{u})$

Figure 3.4: Game muCP used to define the m−cpi-security for M. Here, Sim is the simulator and $A = (A_0, A_1, A_2)$ is the two-stage distinguisher with an additional A_0 stage to choose the number of instances.

3.2.2 Reductions

We show that CP-sequential indifferentiability yields a reduction between public-seed pseudorandomness notions. A special case was shown in [44] by Bellare, Hoang, and Keelvedhi for domain extension of UCEs. Our result goes beyond in that: (1) It is more general, as it deals with arbitrary ideal primitives, (2) It only relies on CP-sequential indifferentiability, as opposed to full indifferentiability, and (3) The reduction of [44] only considered reset-secure sources, whereas we show that under certain conditions on the construction, the reduction also applies to unpredictable sources. Nonetheless, our proofs follow the same approach of [44], and the main contribution is conceptual.

We let $\mathsf{F} = (\mathsf{F.Kg}, \mathsf{F.Eval})$ be a function family which is Σ-compatible with an ideal primitive $\mathbf{I}$. Further, let M be an $(\mathbf{I} \to \mathbf{J})$-compatible construction. Then, overloading notation, we define the new function family $\mathsf{M}[\mathsf{F}] = (\mathsf{M.Kg}, \mathsf{M.Eval})$, where $\mathsf{M.Kg} = \mathsf{F.Kg}$, and for every $k \in [\mathsf{M.Kg}(1^\lambda)]$, we let

$$\mathsf{M.Eval}(1^\lambda, k, x) = \mathsf{M}^{O}(1^\lambda, x) \ , \tag{3.6}$$

where $O(z) = \mathsf{F.Eval}(1^\lambda, k, z)$.

Reset-secure sources.

The following is our general reduction theorem for the case of reset-secure sources.

Theorem 7 (Composition theorem, reset-secure case) *Let* M, F, $\mathbf{I}$, *and* $\mathbf{J}$ *be as above. Fix any simulator* Sim. *Then, for every source-distinguisher pair* (S, D), *where* S *requests at most* $N(\lambda)$ *keys, there exists a source-distinguisher pair* $(\overline{S}, \overline{D})$, *and a further distinguisher* A, *such that*

$$\mathsf{Adv}^{\mathsf{pspr}[\mathbf{J}]}_{\mathsf{M}[\mathsf{F}],S,D}(\lambda) \leq \mathsf{Adv}^{\mathsf{pspr}[\mathbf{I}]}_{\mathsf{F},\overline{S},\overline{D}}(\lambda) + N(\lambda) \cdot \mathsf{Adv}^{\mathsf{cpi}[\mathbf{I}\to\mathbf{J}]}_{\mathsf{M},\mathsf{Sim},A}(\lambda) \ . \tag{3.7}$$

Here, in particular: The complexities of D *and* $\overline{D}$ *are the same. Moreover, if* S, D, *and* M *are polynomial time, and* $\mathbf{I}$, $\mathbf{J}$ *are efficiently implementable, then* A, $\overline{S}$ *and* $\overline{D}$ *are also polynomial-time.*

Moreover, for every reset adversary R, *there exists a reset adversary* R' *and a distinguisher* B *such that*

$$\mathsf{Adv}^{\mathsf{reset}[\mathbf{I}]}_{\overline{S},R}(\lambda) \leq \mathsf{Adv}^{\mathsf{reset}[\mathbf{J}]}_{S,R'}(\lambda) + 3N(\lambda) \cdot \mathsf{Adv}^{\mathsf{cpi}[\mathbf{I}\to\mathbf{J}]}_{\mathsf{M},\mathsf{Sim},B}(\lambda) \ , \tag{3.8}$$

where R' *makes a polynomial number of query / runs in polynomial time if* R *and* Sim *make a polynomial number of queries / run in polynomial time, and* $\mathbf{I}, \mathbf{J}$ *are efficiently implementable.*

Proof: The proof follows the lines of [44]. Consider a source-distinguisher pair S, D using at most $N(\lambda)$ keys in the game $\mathsf{psPR}^{S,D}_{\mathsf{M}[\mathsf{F}],\mathbf{J}}(\lambda)$ for $\mathsf{M}[\mathsf{F}]$.

As the first step, we construct a new source-distinguisher pair $(\overline{S}, \overline{D})$ for the game $\mathsf{psPR}^{\overline{S},\overline{D}}_{\mathsf{F},\mathbf{I}}(\lambda)$. In particular, we first let $D = \overline{D}$ (note that the seed for $\mathsf{M}[\mathsf{F}]$ is a seed for F, so this is a syntactically correct distinguisher). Moreover, $\overline{S}$, is described on the left of Fig. 3.5, and works as follows: given access to its oracle $\mathcal{O}$, $\overline{S}$ simply runs S with a simulated oracle $\mathcal{O}'$. A query $(i, \mathbf{x})$ for $i \in [n]$ made by S to $\mathcal{O}'$ is answered by running

SOURCE $\overline{S}(1^\lambda, \varepsilon)$:	ADV. $A'_0(1^\lambda)$:	ADV. $R'^{\mathcal{O}}(1^\lambda, 1^n, L)$:
$(1^n, \mathsf{st}) \xleftarrow{\$} S(1^\lambda, \varepsilon)$	$(1^n, \mathsf{st}) \xleftarrow{\$} S(1^\lambda, \varepsilon)$	$b' \xleftarrow{\$} R^{\mathcal{O}'}(1^\lambda, 1^n, L)$
return $(1^n, \mathsf{st})$	**return** $(1^n, (1^n, \mathsf{st}))$	**return** $(1^n, (1^n, \mathsf{st}))$
SOURCE $\overline{S}^{\mathcal{O}}(1^\lambda, \mathsf{st})$:	ADV. $A'_1(1^\lambda, (1^n, \mathsf{st}))$:	ORACLE $\mathcal{O}'(i, x)$:
// $\mathsf{st} \neq \varepsilon$	$L \xleftarrow{\$} S^{\mathsf{Func}}(1^\lambda, \mathsf{st})$	$y \xleftarrow{\$} \mathsf{Sim}_i^{\mathcal{O}(i,\cdot)}(x)$
$L \xleftarrow{\$} S^{\mathcal{O}'}(1^\lambda, \mathsf{st})$	**return** $(1^n, L)$	**return** y
return L		
	ADV. $A'_2(1^\lambda, (1^n, L))$:	
ORACLE $\mathcal{O}'(i, x)$:	$k_1, \ldots, k_n \xleftarrow{\$} \mathsf{F.Kg}(1^\lambda)$	
$y \leftarrow \mathsf{M}^{\mathcal{O}(i,\cdot)}(x)$	$b' \xleftarrow{\$} D(1^\lambda, L, k_1, \ldots, k_n)$	
return y	**return** b'	

Figure 3.5: Pseudocode descriptions for the proof of Theorem 7. ORACLE Func is as described in Fig. 3.4 for the muCP game of M. Due to space constraints we use ADV. instead of ADVERSARY.

construction M, where each oracle query z made by M is in turn answered by $\mathcal{O}(i, z)$. Finally, when S outputs leakage $L \in \{0,1\}^*$, $\overline{S}$ simply makes L its own output.

We can then fairly easily show that

$$\mathsf{Adv}^{\mathsf{pspr}[\mathbf{J}]}_{\mathsf{M}[\mathsf{F}],S,D}(\lambda) \leq \mathsf{Adv}^{\mathsf{pspr}[\mathbf{I}]}_{\mathsf{F},\overline{S},\overline{D}}(\lambda) + \mathsf{Adv}^{\mathsf{m-cpi}[\mathbf{I}\to\mathbf{J}]}_{\mathsf{M},\mathsf{Sim},A'}(\lambda) \ ,$$

for an adversary A' we specify in Figure 3.5 (and explain shortly) and *any* simulator Sim. Indeed, note that the only difference between $\mathsf{psPR}^{\overline{S},\overline{D}}_{\mathsf{F},\mathbf{I}}(\lambda)$ and $\mathsf{psPR}^{S,D}_{\mathsf{M}[\mathsf{F}],\mathbf{J}}(\lambda)$ is that in the *ideal* case, S accesses independent instances of $\mathbf{J}$, whereas in the former S (simulated within $\overline{S}$) has access to independent instances of $\mathsf{M}^{\mathbf{I}}$, i.e., each instance has M query an independent instance of $\mathbf{I}$. Therefore, it is sufficient to consider the multi-user CP-indifferentiability adversary A' as described in Figure 3.5.

Further, using Lemma 12, we can build from A' an adversary A such that

$$\mathsf{Adv}^{\mathsf{pspr}[\mathbf{J}]}_{\mathsf{M}[\mathsf{F}],S,D}(\lambda) \leq \mathsf{Adv}^{\mathsf{pspr}[\mathbf{I}]}_{\mathsf{F},\overline{S},\overline{D}}(\lambda) + N(\lambda) \cdot \mathsf{Adv}^{\mathsf{cpi}[\mathbf{I}\to\mathbf{J}]}_{\mathsf{M},\mathsf{Sim},A}(\lambda) \ ,$$

ADV. $C_0^{(1)}(1^\lambda)$:	ADV. $C_0^{(2)}(1^\lambda)$:	ADV. $C_0^{(2)}(1^\lambda)$:
$(1^n, \mathsf{st}) \xleftarrow{\$} S(1^\lambda, \varepsilon)$	$(1^n, \mathsf{st}) \xleftarrow{\$} S(1^\lambda, \varepsilon)$	$(1^n, \mathsf{st}) \xleftarrow{\$} S(1^\lambda, \varepsilon)$
return $(1^n, (1^n, \mathsf{st}))$	**return** $(1^n, (1^n, \mathsf{st}))$	**return** $(1^n, (1^n, \mathsf{st}))$
ADV. $C_1^{(1)}(1^\lambda, (1^n, \mathsf{st}))$:	ADV. $C_1^{(2)}(1^\lambda, (1^n, \mathsf{st}))$:	ADV. $C_1^{(2)}(1^\lambda, (1^n, \mathsf{st}))$:
$L \xleftarrow{\$} S^{\mathsf{Func}}(1^\lambda, \mathsf{st})$	$g_1, \ldots, g_n \xleftarrow{\$} \mathbf{J}_\lambda$	$L \xleftarrow{\$} S^{\mathsf{Func}}(1^\lambda, \mathsf{st})$
return $(1^n, L)$	$L \xleftarrow{\$} S^{\mathcal{O}''}(1^\lambda, \mathsf{st})$	**return** $(1^n, L)$
	return $(1^n, L)$	
ADV. $C_2^{(1)}(1^\lambda, (1^n, L))$:		ADV. $C_2^{(2)}(1^\lambda, (1^n, L))$:
$f_1, \ldots, f_n \xleftarrow{\$} \mathbf{I}_\lambda$	ADV. $C_2^{(2)}(1^\lambda, (1^n, L))$:	$b' \leftarrow R^{\mathsf{Prim}}(1^n, L)$
$b' \leftarrow R^{\mathcal{O}'}(1^n, L)$	$b' \leftarrow R^{\mathsf{Prim}}(1^n, L)$	**return** $1 - b'$
return b'	**return** b'	
ORACLE $\mathcal{O}'(i, x)$:	ORACLE $\mathcal{O}''(i, x)$:	
$y \xleftarrow{\$} f_i(x)$	$y \xleftarrow{\$} g_i(x)$	
return y	**return** y	

Figure 3.6: Pseudocode descriptions for the proof of Theorem 7. ORACLES Func and Prim are as described in Figure 3.4 for the muCP game of M. Due to space constraints we use ADV. instead of ADVERSARY.

given A' always outputs (in its first stage) n such that $n \leq N(\lambda)$ on input 1^λ.

Relating reset-security. It remains to relate the reset-security of S and $\overline{S}$, which is the core of the proof. In particular, fix an arbitrary R for the game $\mathsf{Reset}^R_{\mathbf{I},\overline{S}}(\lambda)$. Then, we are going to build R' for $\mathsf{Reset}^{R'}_{\mathbf{J},S}(\lambda)$ as follows. (A more formal description is on the right of Figure 3.5.) R' runs R with the given leakage L, however rather than giving R direct access to its oracle $\mathcal{O}$, R is run with a simulated oracle $\mathcal{O}'$. To do this, R' runs $N(\lambda)$ independent instances of the simulator Sim – call them $\mathsf{Sim}_1, \ldots, \mathsf{Sim}_{N(\lambda)}$. A query $\mathcal{O}'(i, x)$ is then answered by querying x to Sim_i, where each of Sim's queries y is answered

as $\mathcal{O}(i, y)$.

To continue with our analysis, we denote as $\mathsf{Reset}^{R}_{\mathbf{I},\overline{S}}(\lambda, c)$ and $\mathsf{Reset}^{R'}_{\mathbf{J},S}(\lambda, c)$ for $c \in \{0,1\}$, the games obtained from the original games by fixing the challenge b to c, and the game output is the adversary's output bit b'. Then, by a standard argument,

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{reset}[\mathbf{I}]}_{\overline{S},R}(\lambda) &= \mathsf{Pr}\left[\mathsf{Reset}^{R}_{\mathbf{I},\overline{S}}(\lambda,1)\right] - \mathsf{Pr}\left[\mathsf{Reset}^{R}_{\mathbf{I},\overline{S}}(\lambda,0)\right] \\
&= \mathsf{Pr}\left[\mathsf{Reset}^{R}_{\mathbf{I},\overline{S}}(\lambda,1)\right] - \mathsf{Pr}\left[\mathsf{Reset}^{R'}_{\mathbf{J},S}(\lambda,1)\right] \\
&\quad + \mathsf{Pr}\left[\mathsf{Reset}^{R'}_{\mathbf{J},S}(\lambda,1)\right] - \mathsf{Pr}\left[\mathsf{Reset}^{R'}_{\mathbf{J},S}(\lambda,0)\right] \\
&\quad + \mathsf{Pr}\left[\mathsf{Reset}^{R'}_{\mathbf{J},S}(\lambda,0)\right] - \mathsf{Pr}\left[\mathsf{Reset}^{R}_{\mathbf{I},\overline{S}}(\lambda,0)\right] \\
&= \mathsf{Pr}\left[\mathsf{Reset}^{R}_{\mathbf{I},\overline{S}}(\lambda,1)\right] - \mathsf{Pr}\left[\mathsf{Reset}^{R'}_{\mathbf{J},S}(\lambda,1)\right] \\
&\quad + \mathsf{Adv}^{\mathsf{reset}[\mathbf{J}]}_{S,R'}(\lambda) + \mathsf{Pr}\left[\mathsf{Reset}^{R'}_{\mathbf{J},S}(\lambda,0)\right] - \mathsf{Pr}\left[\mathsf{Reset}^{R}_{\mathbf{I},\overline{S}}(\lambda,0)\right]
\end{aligned} \tag{3.9}$$

We are now going to build a new (multi-user) CP-sequential indifferentiability adversary $\overline{C}$ such that

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{m-cpi}[\mathbf{I}\to\mathbf{J}]}_{\mathsf{M},\mathsf{Sim},\overline{C}}(\lambda) = \frac{1}{3}\Big[&\mathsf{Pr}\left[\mathsf{Reset}^{R}_{\mathbf{I},\overline{S}}(\lambda,1)\right] - \mathsf{Pr}\left[\mathsf{Reset}^{R'}_{\mathbf{J},S}(\lambda,1)\right] \\
&+ \mathsf{Pr}\left[\mathsf{Reset}^{R'}_{\mathbf{J},S}(\lambda,0)\right] - \mathsf{Pr}\left[\mathsf{Reset}^{R}_{\mathbf{I},\overline{S}}(\lambda,0)\right]\Big] .
\end{aligned} \tag{3.10}$$

The adversary C' proceeds as follows. Initially, it selects $i \xleftarrow{\$} \{1,2,3\}$ uniformly at random, and then runs $C^{(i)}$. The adversaries $C^{(1)}, C^{(2)}$, and $C^{(3)}$ are described in Figure 3.6. Then, clearly

$$\mathsf{Adv}^{\mathsf{m-cpi}[\mathbf{I}\to\mathbf{J}]}_{\mathsf{M},\mathsf{Sim},C'}(\lambda) = \frac{1}{3}\sum_{i=1}^{3}\mathsf{Adv}^{\mathsf{m-cpi}[\mathbf{I}\to\mathbf{J}]}_{\mathsf{M},\mathsf{Sim},C^{(i)}}(\lambda)$$

To expand this expression, let us first introduce a hybrid experiment $\overline{\mathsf{Reset}}^{R}_{\mathbf{I},S}(\lambda)$ where in the first stage, S interacts with independent copies of $\mathbf{J}$, and in the second stage, R interacts with independent copies of $\mathbf{I}$. Then,

$$\mathsf{Adv}^{\mathsf{m-cpi}[\mathbf{I}\to\mathbf{J}]}_{\mathsf{M},\mathsf{Sim},C^{(1)}}(\lambda) = \mathsf{Pr}\left[\mathsf{Reset}^{R}_{\mathbf{I},\overline{S}}(\lambda,1)\right] - \mathsf{Pr}\left[\overline{\mathsf{Reset}}^{R}_{\mathbf{I},S}(\lambda)\right] .$$

GAME $\mathsf{EXT}^{S,P}_{\mathsf{M},\mathbf{I},\mathsf{Ext}}(\lambda)$:	ORACLE $\mathcal{O}(i,x)$:
$\mathsf{done} \leftarrow \mathsf{false}$	**if** $\neg\mathsf{done}$ **then** $Q_{\mathbf{I}} \xleftarrow{\cup} \{x\}$
$Q_{\mathbf{I}}, Q_{\mathsf{M}} \leftarrow \emptyset$	**return** $f_i(x)$
$(1^n, \mathsf{st}) \xleftarrow{\$} S(1^\lambda, \varepsilon)$	
$f_1, \ldots, f_n \xleftarrow{\$} \mathbf{I}_\lambda$	ORACLE $\mathcal{O}_{\mathsf{M}}(i,x)$:
$L \xleftarrow{\$} S^{\mathcal{O}_{\mathsf{M}}}(1^\lambda, 1^n, \mathsf{st})$	**if** $\neg\mathsf{done}$ **then** $Q_{\mathsf{M}} \xleftarrow{\cup} \{x\}$
$\mathsf{done} \leftarrow \mathsf{true}$	$y \leftarrow \mathsf{M}^{\mathcal{O}(i,\cdot)}(x)$
$Q \xleftarrow{\$} P^{\mathcal{O}}(1^\lambda, 1^n, L)$; $Q^* \leftarrow \mathsf{Ext}^{\mathcal{O}}(Q)$	**return** y
return $((Q \cap Q_{\mathbf{I}} \neq \emptyset) \wedge (Q^* \cap Q_{\mathsf{M}} = \emptyset))$	

Figure 3.7: Game $\mathsf{EXT}^{S,P}_{\mathsf{M},\mathbf{I},\mathsf{Ext}}(\lambda)$ in the definition of query extractability.

Further,

$$\mathsf{Adv}^{\mathsf{m-cpi}[\mathbf{I}\to\mathbf{J}]}_{\mathsf{M},\mathsf{Sim},C^{(2)}}(\lambda) = \mathsf{Pr}\left[\overline{\mathsf{Reset}}^{R}_{\mathbf{I},S}(\lambda)\right] - \mathsf{Pr}\left[\mathsf{Reset}^{R'}_{\mathbf{J},S}(\lambda,1)\right] .$$

Finally, it is not hard to see that

$$\mathsf{Adv}^{\mathsf{m-cpi}[\mathbf{I}\to\mathbf{J}]}_{\mathsf{M},\mathsf{Sim},C^{(3)}}(\lambda) = \mathsf{Pr}\left[\mathsf{Reset}^{R'}_{\mathbf{J},S}(\lambda,0)\right] - \mathsf{Pr}\left[\mathsf{Reset}^{R}_{\mathbf{I},\overline{S}}(\lambda,0)\right] .$$

Note that in $C^{(3)}$ the flipping of the bits is necessary because the execution of $\mathsf{Reset}^{R'}_{\mathbf{J},S}(\lambda,0)$ corresponds to the ideal world in the indifferentiability experiment, whereas $\mathsf{Reset}^{R}_{\mathbf{I},\overline{S}}(\lambda,0)$ corresponds to the real world, and thus we have to switch signs. To conclude the proof, we can apply Lemma 12 to obtain, from C', an attacker against the single-user CP-indifferentiability of C. ■

Query extractable constructions.

Next, we show that under strong conditions on the construction M, Theorem 7 extends to the case of unpredictability.

In particular, we consider constructions which we term *query extractable*. Roughly, what such constructions guarantee is that every query made by M to an underlying ideal

primitive $\mathbf{I}$ can be assigned to a (small) set of possible inputs to M that would result in this query during evaluation. Possibly, this set of inputs may be found by making some additional queries to $\mathbf{I}$. We define this formally through the game $\mathsf{EXT}^{S,P}_{\mathsf{M},\mathbf{I},\mathsf{Ext}}(\lambda)$ in Fig. 3.7. It involves a *source* S and a *predictor* P, as well as an extractor Ext. Here, S selects an integer n, which results in n instances $f_1, \ldots, f_n$ of $\mathbf{I}$ being spawned, and then makes queries to n instances of M^{f_i}, gives some leakage to the predictor P, and the predictor makes further query to the $\mathbf{I}$-instances, until it outputs a set Q. Then, we run the extractor Ext on Q, and the extractor can also make additional queries to the $\mathbf{I}$-instances, and outputs an additional set Q^*. We are interested in the event that Q contains one of queries made to the f_i's by M in the first stage of the game, yet Q^* does not contain any of S's queries to M^{f_i} for some i. In particular, we are interested in

$$\mathsf{Adv}^{\mathsf{ext}[\mathbf{I}]}_{\mathsf{M},S,P,\mathsf{Ext}}(\lambda) = \mathsf{Pr}\left[\mathsf{EXT}^{S,P}_{\mathsf{M},\mathbf{I},\mathsf{Ext}}(\lambda)\right] .$$

We say that M is *query extractable with respect to* $\mathbf{I}$ if there exists a polynomial time Ext such that $\mathsf{Adv}^{\mathsf{ext}[\mathbf{I}]}_{\mathsf{M},S,P,\mathsf{Ext}}(\lambda)$ is negligible for all PPT P and S. We say it is *perfectly* query extractable if the advantage is 0, rather than simply negligible.

The next theorem provides an alternative to Theorem 7 for the case of unpredictable sources whenever M guarantees query extractability.

Theorem 8 (Composition theorem, unpredictable case) *Let* M, F, $\mathbf{I}$, *and* $\mathbf{J}$ *be as before. Fix any simulator* Sim. *Then, for every source-distinguisher pair* (S, D), *where* S *requests at most* $N(\lambda)$ *keys, there exists a source-distinguisher pair* $(\overline{S}, \overline{D})$, *and a further distinguisher* A, *such that*

$$\mathsf{Adv}^{\mathsf{pspr}[\mathbf{J}]}_{\mathsf{M}[\mathsf{F}],S,D}(\lambda) \leq \mathsf{Adv}^{\mathsf{pspr}[\mathbf{I}]}_{\mathsf{F},\overline{S},\overline{D}}(\lambda) + N(\lambda) \cdot \mathsf{Adv}^{\mathsf{cpi}[\mathbf{I}\to\mathbf{J}]}_{\mathsf{M},\mathsf{Sim},A}(\lambda) . \tag{3.11}$$

Here, in particular: The complexities of D *and* $\overline{D}$ *are the same. Moreover, if* S, D, *and* M *are polynomial time, and* $\mathbf{I}$, $\mathbf{J}$ *are efficiently implementable, then* A, $\overline{S}$ *and* $\overline{D}$ *are also polynomial-time.*

MAIN $\mathsf{G}_0(\lambda)$,$\mathsf{G}_1(\lambda)$:	ORACLE $\mathsf{Func}(i,x)$: // G_0	ORACLE $\mathsf{Func}(i,x)$: // G_1
$\mathsf{done} \leftarrow \mathsf{false}$	$Q_{\mathsf{M}} \xleftarrow{\cup} \{x\}$	$Q_{\mathsf{M}} \xleftarrow{\cup} \{x\}$
$Q_{\mathbf{I}} \leftarrow \emptyset$	$y \leftarrow \mathsf{M}^{\mathsf{Prim}(i,\cdot)}(x)$	$y \leftarrow g_i(x)$
$(1^n, t) \xleftarrow{\$} S(1^\lambda, \varepsilon)$	**return** y	**return** y
$f_1, \ldots, f_n \xleftarrow{\$} \mathbf{I}_\lambda$		
$g_1, \ldots, g_n \xleftarrow{\$} \mathbf{J}_\lambda$	ORACLE $\mathsf{Prim}(i,x)$: // G_0	ORACLE $\mathsf{Prim}(i,x)$: // G_1
$L \xleftarrow{\$} S^{\mathsf{Func}}(1^\lambda, t)$	**if** $\neg\mathsf{done}$ **then** $Q_{\mathbf{I}} \xleftarrow{\cup} \{x\}$	**if** $\neg\mathsf{done}$ **then** $Q_{\mathbf{I}} \xleftarrow{\cup} \{x\}$
$\mathsf{done} \leftarrow \mathsf{true}$	**return** $f_i(x)$	**return** $\mathsf{Sim}_i^{g_i}(x)$
$Q \xleftarrow{\$} P^{\mathsf{Prim}}(1^\lambda, 1^n, L)$		
$Q^* \xleftarrow{\$} \mathsf{Ext}^{\mathsf{Prim}}(Q)$		
return $(Q_{\mathsf{M}} \cap Q^* \neq \emptyset)$		

Figure 3.8: Games used in the proof of Theorem 8.

Moreover, for every predictor P and extractor Ext*, there exists a predictor adversary P' and a distinguisher B such that*

$$\mathsf{Adv}^{\mathsf{pred}[\mathbf{I}]}_{\overline{S},P}(\lambda) \leq \mathsf{Adv}^{\mathsf{pred}[\mathbf{J}]}_{S,P'}(\lambda) + \mathsf{Adv}^{\mathsf{ext}[\mathbf{I}]}_{\mathsf{M},S,P,\mathsf{Ext}}(\lambda) + N(\lambda) \cdot \mathsf{Adv}^{\mathsf{cpi}[\mathbf{I}\to\mathbf{J}]}_{\mathsf{M},\mathsf{Sim},B}(\lambda)\ , \tag{3.12}$$

where P' makes a polynomial number of query / runs in polynomial time if P, Sim *and* Ext *make a polynomial number of queries / run in polynomial time, and $\mathbf{I}, \mathbf{J}$ are efficiently implementable.*

Proof: Note that (3.11) is established exactly as in the proof of Theorem 7; the definitions of $\overline{S}$ and $\overline{D}$ are identical, and the statement is independent of the specific properties of the source.

Therefore, the rest of this proof relates the unpredictability of S with that of $\overline{S}$. In particular, given P, we construct P' as follows. On input 1^λ, 1^n and L, P' runs $P(1^\lambda, 1^n, L)$, but answers its oracle calls (i,x) with $\mathsf{Sim}^{\mathcal{O}(i,\cdot)}(x)$, i.e., using the simulator Sim. When finally P outputs Q, P' runs $\mathsf{Ext}^{\mathcal{O}}(Q)$, and outputs its output Q^*.

To establish (3.12), we consider two games, G_0 and G_1, as depicted in Figure 3.8.

Game G_0 is the same as game $\mathsf{Pred}^{P}_{\mathbf{I},\overline{S}}(\lambda)$, except that the winning condition is set when Q^* extracted by Ext contains one of the queries to Func made by S, rather than one of the Prim queries made by M while evaluating S's queries. Therefore, the fact that $\mathsf{Pr}[A] \leq \mathsf{Pr}[B] + \mathsf{Pr}[A \wedge \neg B]$ for all events A, B yields

$$\begin{aligned}\mathsf{Adv}^{\mathsf{pred}[\mathbf{I}]}_{\overline{S},P}(\lambda) &= \mathsf{Pr}[Q_{\mathbf{I}} \cap Q \neq \emptyset] \\ &\leq \mathsf{Pr}[Q_{\mathsf{M}} \cap Q^* \neq \emptyset] + \mathsf{Pr}[(Q_{\mathbf{I}} \cap Q \neq \emptyset) \wedge (Q_{\mathsf{M}} \cap Q^* = \emptyset)] \\ &\leq \mathsf{Pr}[\mathsf{G}_0] + \mathsf{Adv}^{\mathsf{ext}[\mathbf{I}]}_{\mathsf{M},S,P,\mathsf{Ext}}(\lambda) \ .\end{aligned}$$

Game G_1 simply modifies oracle Func to reply to queries (i, x) using g_i, where $g_1, \ldots, g_n \xleftarrow{\$} \mathbf{J}_\lambda$. Similarly, it runs n copies of the simulator Sim – call them $\mathsf{Sim}_1, \ldots, \mathsf{Sim}_n$ – and queries $\mathsf{Prim}(i, x)$ are answered by $\mathsf{Sim}_i^{g_i}$. it is now straightforward to build a distinguisher B' for the multi-user CP-sequential indifferentiability game such that

$$\mathsf{Adv}^{\mathsf{m-cpi}[\mathbf{I}\to\mathbf{J}]}_{\mathsf{M},\mathsf{Sim},B'}(\lambda) = \mathsf{Pr}[\mathsf{G}_0] - \mathsf{Pr}[\mathsf{G}_1] \ .$$

This is because B' can easily keep track of all sets Q_{M} and $Q_{\mathbf{I}}$ and simulate the rest of G_0 or G_1 using the given oracles Func and Prim. The final distinguisher B can be obtained from B' by applying Lemma 12. ■

3.3 Public-seed Pseudorandomness of Feistel

This section presents the main result of this chapter which builds psPRPs from UCEs, namely that the five-round Feistel construction, when its round functions are instantiated from a $\mathsf{UCE}[\mathcal{S}^{\star\mathsf{rs}}]$-secure function family (for $\star \in \{\mathsf{c}, \mathsf{s}\}$), yields a $\mathsf{psPRP}[\mathcal{S}^{\star\mathsf{rs}}]$-secure permutation family.

CP-indifferentiability of Feistel. Let $n : \mathbb{N} \to \mathbb{N}$ be a (polynomially bounded) function. We define the following construction Ψ_5, which, for security parameter λ,

implements an invertible permutation on $2n(\lambda)$-bit strings, and makes calls to an oracle $f : [5] \times \{0,1\}^{n(\lambda)} \to \{0,1\}^{n(\lambda)}$. In particular, on input 1^λ and $X = X_0 \,\|\, X_1$, where $X_0, X_1 \in \{0,1\}^{n(\lambda)}$, running $\Psi_5^f(1^\lambda, (+, X))$ outputs $X_5 \,\|\, X_6$, where

$$X_{i+1} \leftarrow X_{i-1} \oplus f(i, X_i) \text{ for all } i = 1, \ldots, 5 \,. \tag{3.13}$$

Symmetrically, upon an inverse query, $\Psi_5^f(1^\lambda, (-, Y = X_5 \,\|\, X_6))$ simply computes the values backwards, and outputs $X_0 \,\|\, X_1$. Construction Ψ_5 is clearly $(\mathbf{R}_{n,n}^5 \to \mathbf{P}_{2n})$-compatible, where we use the notation $\mathbf{R}_{n,n}^k$ to denote the k-fold combination of independent random functions which takes queries of the form (i, x) that are answered by evaluating on x the i-th function.

The following theorem establishes CP-indifferentiability for Ψ_5. We discuss below its consequences, and give a detailed description of our simulation strategy and the full analysis of the simulation strategy – which employes the randomness-mapping technique of [31] in Section 3.4

Theorem 9 (CP-indifferentiability of Feistel) *Let* $\mathbf{R} = \mathbf{R}_{n,n}^5$ *and* $\mathbf{P} = \mathbf{P}_{2n}$. *Then, there exists a simulator* Sim *(described in Fig. 3.9) such that for all distinguisher* A *making at most* $q(\lambda)$ *queries,*

$$\mathsf{Adv}^{\mathsf{cpi}[\mathbf{R}\to\mathbf{P}]}_{\Psi_5,\mathsf{Sim},A}(\lambda) \leq \frac{360q(\lambda)^6}{2^{n(\lambda)}} \,. \tag{3.14}$$

Here, Sim *makes at most* $2q(\lambda)^2$ *queries, and otherwise runs in time polynomial in the number of queries answered, and* n.

This, together with Theorem 7, gives us immediately the following corollary: Given a keyed function family $\mathsf{F} = (\mathsf{F.Kg}, \mathsf{F.Eval})$, where for all $\lambda \in \mathbb{N}$, $k \in [\mathsf{F.Kg}(1^\lambda)]$, $\mathsf{F.Eval}(1^\lambda, k, \cdot)$ is a function from $n(\lambda) + 3$ bits to $n(\lambda)$ bits, interpreted as a function $[5] \times \{0,1\}^{n(\lambda)} \to \{0,1\}^{n(\lambda)}$, then define the keyed function family $\Psi_5[\mathsf{F}] = (\Psi.\mathsf{Kg}, \Psi.\mathsf{Eval})$ obtained by instantiating the round function using F.

Corollary 1 *For any polynomially bounded* $n = \omega(\log \lambda)$*, if* $\mathsf{F} \in \mathsf{UCE}[n+3, \mathcal{S}^{\star\mathsf{rs}}]$*, then* $\Psi_5[\mathsf{F}] \in \mathsf{psPRP}[2n, \mathcal{S}^{\star\mathsf{rs}}]$*, where* $\star \in \{\mathsf{c}, \mathsf{s}\}$*.*

Remarks. Theorem 9 is interesting in its own right, as part of the line of works on (full-fledged) indifferentiability of Feistel constructions. Coron et al. [47] show that six rounds are necessary for achieving indifferentiability, and proofs of indifferentiability have been given for 14, 10, and 8 rounds, respectively [31, 47, 34, 35]. Thus, our result shows that CP-indifferentiability is a strictly weaker goal in terms of round-complexity of the Feistel construction. (Also for sequential indifferentiability as in [46], six rounds are necessary.) As we will see in the next paragraph, our simulation strategy departs substantially from earlier proofs.

Two obvious problems remain open. (1) First off, we know four rounds are necessary (as they are needed for indistinguishability alone [21]), but we were unable to make any progress on whether CP-sequential indifferentiability (or psPRP security) is achievable. (2) The second is the case of unpredictable sources. We note that a heavily unbalanced Feistel construction (where each round function outputs one bit) would be query extractable, as the input of the round function leaves little uncertainty on the inner state, and the extractor can evaluate the round functions for other rounds to infer the input/output of the construction. Thus, if we could prove CP-indifferentiabilty, we could combine this with Theorem 8. Unfortunately, such a proof appears beyond our current understanding.

In Chapter 4, we solve both questions, and even more in fact, showing that the Naor-Reingold (NR) construction [22] solves *both* (1) and (2) which furthermore requires only two-calls to the underlying UCE hash function. We direct the reader to Chapter 4 for full details.

3.4 Sequential Indifferentiability of 5-round Feistel

In this section, we provide the proof of sequential indifferentiability of the 5-round Feistel network Ψ_5. In Section 3.4.1 we provide a high level overview of our simulator. Then, in Section 3.4.2 we develop some formal notation for the proof and describe the hybrids used. Finally, we provide a detailed analysis of the indistinguishability of the hybrids in Section 3.4.3 through Section 3.4.5 which completes the full proof.

3.4.1 Indifferentiability Simulator Overview

We explain now our simulation strategy, which is described formally in Fig. 3.9. We note that our approach inherits the chain-completion technique from previous proofs, but it will differ substantially in how and when chains are completed.

Recall that in the ideal case, in the first stage of the CP-indifferentiability game, A_1 makes queries to Func implementing a random permutation, and then passes the control of the game to A_2 which interacts with Sim. Our Sim maintains tables G_k for $k \in [5]$ to simulate the round functions. We denote by $G_k[X] = \perp$ that the table entry for X is undefined, and we assume all values are initially undefined. Also, we refer to a tuple (X_k, X_{k+1}, k) as a *partial chain* where $G_k[X_k] \neq \perp$ and $G_{k+1}[X_{k+1}] \neq \perp$ for $k \in \{1, 4\}$, $X_k, X_{k+1} \in \{0,1\}^n$.

For any query (k, X) by A_2, Sim checks if $G_k[X] = \perp$. If not then the image $G_k[X]$ is returned. Otherwise, depending on the value of k, Sim takes specific steps as shown in Fig. 3.9. If $k \in \{2, 4\}$ then Sim sets $G_k[X]$ to a uniformly random n-bit string by calling the procedure $\mathsf{F}^{\text{inner}}$. At this point, Sim considers newly formed tuples $(X_1, X_2) \in G_1 \times \{X\}$ (when $k = 2$) and detects partial chains $C = (X_1, X_2, 1)$. The notation $X_1 \in G_1$ is equivalent to $G_1[X_1] \neq \perp$. For every partial chain C that Sim detects, it queries Func on (X_0, X_1) and gets (X_5, X_6) where $X_0 = G_1[X_1] \oplus X_2$. If (X_0, X_1) does not appear in

PROCEDURE $\mathsf{Sim}(k, X)$:

1: **if** $G_k[X] = \perp$ **then**
2: **if** $k = 2$ **then**
3: $\mathsf{F}^{\text{inner}}(k, X)$
4: **foreach** $(X_1, X_2) \in G_1 \times \{X\}$ **do**
5: **if** $(X_1, X_2, 1) \notin$ CompletedChains **then**
6: $X_0 \leftarrow \mathsf{F}^{\text{inner}}(1, X_1) \oplus X_2$
7: $(X_5, X_6) \leftarrow \mathsf{Func}(+, X_0||X_1)$
8: $C \leftarrow (X_1, X_2, 1)$
9: **if** $G_5[X_5] \neq \perp$ **then** // Immediate Completion
10: $\mathsf{Complete}(C, (X_5, X_6))$
11: **else** // Completion is delayed
12: $X_3 \leftarrow \mathsf{F}^{\text{inner}}(2, X_2) \oplus X_1$
13: Chains$[3, X_3] \leftarrow (5, X_5)$, Chains$[5, X_5] \xleftarrow{\cup} \{(C, (X_5, X_6))\}$
14: **elseif** $k = 4$ **then**
15: $\mathsf{F}^{\text{inner}}(k, X)$
16: **foreach** $(X_4, X_5) \in \{X\} \times G_5$ **do**
17: **if** $(X_4, X_5, 4) \notin$ CompletedChains **then**
18: $X_6 \leftarrow \mathsf{F}^{\text{inner}}(4, X_4) \oplus X_5$
19: $(X_0, X_1) \leftarrow \mathsf{Func}(-, X_5||X_6)$
20: $C \leftarrow (X_4, X_5, 4)$
21: **if** $G_1[X_1] \neq \perp$ **then** // Immediate Completion
22: $\mathsf{Complete}(C, (X_0, X_1))$
23: **else** // Completion is delayed
24: $X_3 \leftarrow \mathsf{F}^{\text{inner}}(4, X_4) \oplus X_5$
25: Chains$[3, X_3] \leftarrow (1, X_1)$, Chains$[1, X_1] \xleftarrow{\cup} \{(C, (X_0, X_1))\}$

Figure 3.9: The code for simulator Sim. Sim has access to the Func oracle and maintains data structures G_k, Chains and CompletedChains as global variables.

```
26:     elseif k ∈ {1,5} then
27:       F^inner(k, X)
28:       foreach (C, (U,V)) ∈ Chains[k, X] do
29:         if C ∉ CompletedChains then // Delayed Completion
30:           Complete(C, (U,V))
31:     elseif Chains[3, X] ≠ ⊥ then
32:       Sim(Chains[k, X])
33:   return F^inner(k, X)

PROCEDURE F^inner(i, X_i):
34:   if G_i[X_i] = ⊥ then
35:     G_i[X_i] ←$ {0,1}^n
36:   return G_i[X_i]

PROCEDURE ForceVal(X, Y, l):
37:   G_l[X] ← Y
```

Figure 3.9: (continued) The code for subroutines used by simulator Sim.

one of the queries/responses by/to A_1 then it is unlikely for A_2 to guess the corresponding (X_5, X_6) pair. Therefore, if $G_5[X_5] \neq \perp$ then Sim assumes that C is a chain that most likely corresponds to a query by A_1. We refer to partial chains that correspond to the queries by A_1 as *relevan*t chains. In this case, Sim *immediately* completes C by calling the procedure Complete. C is completed by forcing the values of $G_3[X_3]$ and $G_4[X_4]$ to be consistent with the Func query where $X_3 \leftarrow G_2[X_2] \oplus X_1$ and $X_4 \leftarrow G_5[X_5] \oplus X_6$.

If $G_5[X_5] = \perp$ then either C is not a relevant chain or C is a relevant chain but A_2 has not queried $(5, X_5)$ yet. An aggressive strategy would be to complete C, thereby asking Sim to complete every partial chain ever detected. The resulting simulation strat-

PROCEDURE $\mathsf{Complete}(C, (U, V))$:

38: $(X, Y, i) \leftarrow C$

39: **if** $i = 1$ **then**

40: $X_1 \leftarrow X, X_2 \leftarrow Y,\ X_3 \leftarrow \mathsf{F}^{\text{inner}}(2, X_2) \oplus X_1$

41: $(X_5, X_6) \leftarrow (U, V)$

42: $X_4 \leftarrow \mathsf{F}^{\text{inner}}(5, X_5) \oplus X_6$

43: $\mathsf{ForceVal}(X_3, X_4 \oplus X_2, 3)$, $\mathsf{ForceVal}(X_4, X_5 \oplus X_3, 4)$

44: **elseif** $i = 4$ **then**

45: $X_4 \leftarrow X, X_5 \leftarrow Y,\ X_3 \leftarrow \mathsf{F}^{\text{inner}}(4, X_4) \oplus X_5$

46: $(X_0, X_1) \leftarrow (U, V)$

47: $X_2 \leftarrow \mathsf{F}^{\text{inner}}(1, X_1) \oplus X_0$

48: $\mathsf{ForceVal}(X_3, X_4 \oplus X_2, 3)$, $\mathsf{ForceVal}(X_2, X_1 \oplus X_3, 2)$

49: CompletedChains $\overset{\cup}{\leftarrow} \{(X_1, X_2, 1), (X_4, X_5, 4)\}$

Figure 3.9: (continued) The code for subroutines used by simulator Sim.

egy will however end up potentially managing an exponential number of partial chains, contradicting our goal of efficient simulation. Hence, Sim *delays* the completion and only completes C on A_2's query to either $(3, X_3)$ or $(5, X_5)$ where $X_3 = G_2[X_2] \oplus X_1$. The completion is delayed by storing information about X_3 and X_5, that fall on the chain C, in the table Chains. In particular, Sim stores a pointer to $(5, X_5)$ at Chains$[3, X_3]$. The inputs $((X_1, X_2, 1), (X_5, X_6))$ to the $\mathsf{Complete}$ call on C are stored in Chains$[5, X_5]$. As many chains can share the same X_5, we allow Chains$[5, X_5]$ to be a set. The idea of delaying the chain completions is unique to our simulation strategy and it translates to an efficient Sim which consistently completes chains in the eyes of A. Sim works symmetrically when $k = 4$.

For queries of the form (k, X) where $k \in \{1, 5\}$, Sim always assigns $G_k[X]$ to a uniform random n-bit string by calling $\mathsf{F}^{\text{inner}}$. Moreover as discussed earlier, X could

be on previously detected partial chains whose completion was delayed. Therefore after the assignment, Sim picks up all partial chains C' (if any) stored in Chains$[k, X]$ and completes them. This is where Sim captures a relevant partial chain which was delayed for completion. Finally for queries $(3, X)$, Sim checks if this X was on a partial chain that was detected but not completed. If Chains$[3, X] = \perp$ then Sim assigns $G_3[X]$ a uniform random n-bit string otherwise it follows the pointer to Chains$[3, X]$ to complete the chain X was on. Since Chains$[3, X]$ just stores a tuple (instead of a set) there can be at most one chain C that Chains$[3, X]$ can point to at any time. In the execution, Chains$[3, X]$ can get overwritten which may lead to inconsistencies in chain completions. However, we show that there are no overwrites in either tables G_k or the data-structure Chains, except with negligible probability. This allows Sim to Complete chains consistently in the eyes of A. Furthermore, to avoid completing the same chains again, Sim maintains a set of all CompletedChains and completes any chain if it is not in CompletedChains. A pictorial description of Sim is found in Fig. 3.10.

3.4.2 Notation and Description of Hybrids

For the entire proof we will let n denote the security parameter and we omit it from definitions of games for ease of notation. To prove our claim, we fix a deterministic distinguisher $A = (A_1, A_2)$ that makes at most q queries overall, i.e., we sum the number of queries in its first and second stage. [4] For ease of description, we further assume that A_1 makes $q_c \leq q$ (construction) queries and denote its queries as $(\sigma, X^i_k, X^i_{k+1})$ for $(\sigma, k) \in \{(+, 0), (-, 5)\}$ and $i \in [q_c]$. We refer to these queries as relevant queries. From now on, any query described with superscript i must be understood as either the query by A_1 or its response. We also assume that A_2 issues primitive queries corresponding to

[4] We may assume A is deterministic without loss of generality, as the advantage of any probabilistic distinguisher is at most the advantage of the corresponding deterministic distinguisher obtained by optimally fixing its random tape.

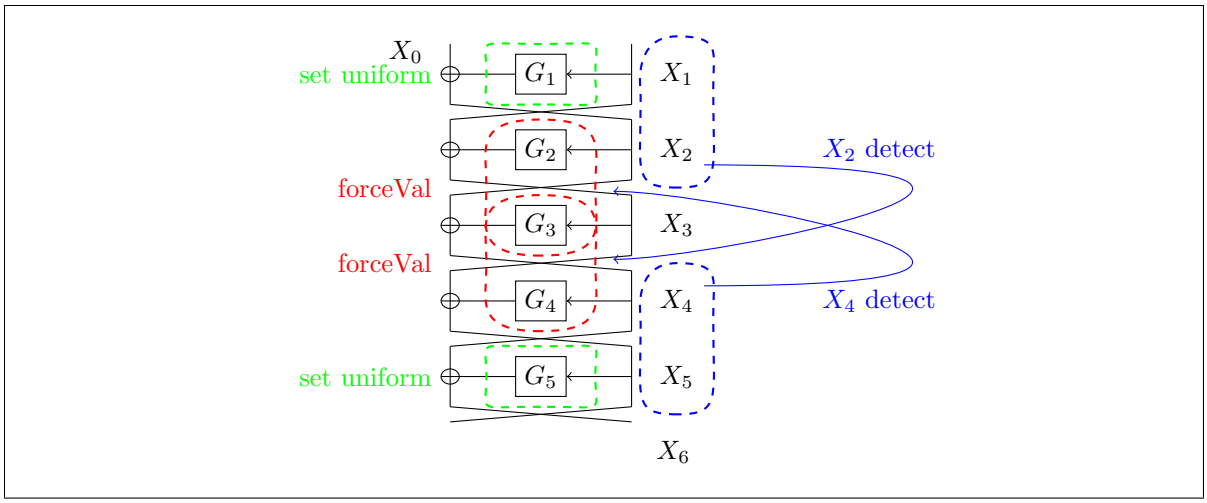


Figure 3.10: The 5-round Feistel where Sim sets $G_1[X_1]$ and $G_5[X_5]$ uniformly at random (green). Sim detects chains at either (X_1, X_2) or (X_4, X_5) (blue) and adapts at (X_3, X_4) and (X_2, X_3) respectively (red).

all construction queries by A_1. This can be achieved by allowing A_2 run the procedure AllPrimitive described in Figure 3.11 at the end and blowing up the number of queries only by a factor of 5.

Our proof considers four games denoted as G_i for $i \in [4]$ which are described in Figures 3.11, 3.14. Furthermore we define the quantity $\Delta(\mathsf{G}_i, \mathsf{G}_{i+1})$,

$$\Delta(\mathsf{G}_i, \mathsf{G}_{i+1}) = |\mathsf{Pr}[\mathsf{G}_i] - \mathsf{Pr}[\mathsf{G}_{i+1}]| \ . \tag{3.15}$$

We describe our games next.

Game G_1: G_1 described in Figure 3.11 is exactly the case when $b = 0$ in the CP-sequential indifferentiability game of Ψ_5. The Func oracle implements a permutation via lazy sampling and Prim oracle invokes the procedure $\mathsf{Sim}(k, X)$ on queries (k, X). In addition, Sim inherits oracle access to Func from Prim.

Game G_2: replacing permutation with a two-sided random function: In this game, we modify the oracle Func such that now instead of a permutation it acts like a two-sided random function working with pre-sampled randomness ρ. The description of Func for game G_2 is given in Figure 3.11. Func maintains a table P to simulate the permutation. For a (forward) query $(+, X_0||X_1)$ to Func, if $P[+, X_0, X_1]$ is already defined then the image $P[+, X_0, X_1]$ is returned as the response. Otherwise, Func gets a $2n$-bit string (X_5, X_6) from $\rho[+, X_0, X_1]$ and maps (X_0, X_1) to (X_5, X_6) and vice versa. This is shown in Figure 3.11. Similarly for a (backward) query $(-, X_5||X_6)$ to Func such that $P[-, X_5, X_6]$ is not defined Func gets a $2n$-bit string from $\rho[-, X_5, X_6]$ and updates P accordingly. The random table ρ, therefore, contains a $2n$-bit string for all $(+, X_0, X_1)$ and $(-, X_5, X_6)$.

Moreover, we also sample the randomness f used by Sim at the beginning of the game G_2. The table f contains an n-bit string for all indices (k, X) where $k \in [5]$ and $X \in \{0,1\}^n$. We refer to the simulator with presampled randomness f as $\mathsf{Sim}(f)$. Whenever Sim lazily samples randomness to assign $G_k[X]$ (line 37 of Figure 3.9), $\mathsf{Sim}(f)$ instead accesses $f[k, X]$. Sampling f at the beginning does not change the distribution of the simulation. This is because Sim considers an entry of f at most once. The code for $\mathsf{Sim}(f)$ is shown in Figure 3.9 alongside the code of Sim.

After A_2 has output its guess b', we allow Sim to run an additional procedure called $\mathsf{AllComplete}$ (described in Figure 3.11). We refer to the execution of G_2 until the point when A_2 outputs its guess as Phase 1 and the remaining execution as Phase 2. The output guess of A_2 is independent of Phase 2 as A_2 learns nothing about it. Note that in Phase 2, Sim does not make any calls to Func, it just completes incomplete chains that were delayed. This extension to Phase 2 is necessary to argue about the indistinguishability of G_2 with G_3 (defined next).

By $\mathcal{F}$, $\mathcal{R}$ we denote the set of all tables f and ρ respectively as described above. We

consider an alternative game $\mathsf{G}_2^{f,\rho}$ where the first step of sampling random tables from game G_2 is omitted and instead we use (f, ρ) as the tables. In other words, game $\mathsf{G}_2^{f,\rho}$ can be thought of as running G_2 with the fixed (f, ρ).

Game G_3: replacing two-sided random function with $\Psi_5(h)$: In G_3, we replace the two-sided random function with the Feistel construction $\Psi_5(h)$ i.e. Func now evaluates the Feistel construction to reply to its queries. Like in the previous game, we again sample the randomness $h \in \mathcal{F}$ at the beginning of the game. One major difference between G_2 and G_3 is that in G_3 the randomness h is shared between Prim (and hence Sim) and Func oracles. The game G_3 maintains tables H_k which have the same semantics as G_k. As shown in Figure 3.14, whenever Func needs to access the value of the kth round function on X it accesses $H_k[X]$ if it is defined. Otherwise, it sets $H_k[X] \leftarrow h[k, X]$ and then accesses $H_k[X]$. Sim does the same when it needs to set $G_k[X]$. The entire working of Sim is the same as in G_2 (and G_1) except the procedure $\mathsf{F}^{\text{inner}}$ which is shown in Figure 3.14. Like in G_2, we allow Sim to run the procedure AllComplete after A_2 has output its guess.

We consider an alternative game G_3^h where the first step of the sampling random table h is omitted from game G_3 and instead we use h as the table. In other words, game G_3^h can be thought of as running G_3 with the fixed h.

Game G_4: removing the simulator Sim: In this game, we modify Prim such that it behaves like a random function as in the case of $b = 1$ in the CP-sequential indifferentiability game of the five-round Feistel construction. The Func oracle which implements the Feistel five-round construction has access to Prim for round functions.

From the choice of our games, it follows that $\mathsf{Adv}^{\mathsf{cpi}[\mathbf{R}\to\mathbf{P}]}_{\Psi_5,\mathsf{Sim},A}(n) \leq \Delta(\mathsf{G}_1, \mathsf{G}_4)$. Our focus from now on will be to derive $\Delta(\mathsf{G}_i, \mathsf{G}_{i+1})$ for $i \in [3]$ and then by the triangle inequality

we derive a bound on $\Delta(\mathsf{G}_1, \mathsf{G}_4)$.

Next, we prove that each of the adjacent hybrids G_j and G_{j+1} are indistinguishable, thereby concluding the proof of Theorem 9.

3.4.3 Indistinguishability of G_1 and G_2

In this section, we derive a bound for $\Delta(\mathsf{G}_1, \mathsf{G}_2)$ by invoking a standard argument about replacing a permutation with a two-sided random function. We will consider (Sim, A) as a coupled distinguisher D that makes q' queries to the permutation/two-sided random function and therefore has advantage $q'^2/2^{2n}$ in distinguishing the random permutation from the two-sided random function. In Lemma 13, we will prove Sim's query complexity and thereby establish $\Delta(\mathsf{G}_1, \mathsf{G}_2)$ in Lemma 14.

Lemma 13 *At any point in* G_2, $|G_k| \leq q + 2q^2$ *and* $|P| \leq q_c + 2q^2$ *where* $k \in [5]$ *and* $|G_k|$ *and* $|P|$ *is the size of the respective tables.*

Proof: In Phase 1 of the game G_2, for every element that gets added to G_1 and G_5 there exists a query by A_2 to Prim. This is because elements get added to G_1 (in Phase 1) either due to a query $(1, X_1)$ by A_2 such that $G_1[X_1] = \perp$ or due to a query $(3, X_3)$ by A_2 such that $\mathsf{Chains}[3, X_3] = (1, X_1)$ and $G_3[X_3] = \perp$ (lines 31-32 in Figure 3.9). Therefore, $|G_1| \leq q$ and $|G_5| \leq q$ at any point in Phase 1.

Now, we bound the size of the table P. Sim issues queries to Func only when A_2 queries $\mathsf{Prim}(k, X)$ where $k \in \{2, 4\}$ and $G_k[X] = \perp$. Since queries to Func are only issued in Phase 1 and $|G_1| \leq q$, then for every query $(2, X)$ by A_2 there are at most q queries to Func. Over q queries by A_2, there are at most q^2 queries to Func by Sim. Following the same analysis for Prim queries of the form $(4, \cdot)$, Sim issues at most $2q^2$ queries to Func in G_2.

Therefore, $|P| \leq q_c + 2q^2 \leq 3q^2$ where the inequality is due to $q_c \leq q$.

Next we bound $|G_k|$ and $|\text{Chains}[j]|$ where $k \in [5]$ and $j \in \{1,3,5\}$ where

$$\begin{aligned} \text{Chains}[l] &= \bigcup_{(l,X):\text{Chains}[l,X]\neq\perp} \text{Chains}[l,X] \qquad \text{where } l \in \{1,5\} \ , \\ \text{Chains}[3] &= \bigcup_{(3,X):\text{Chains}[3,X]} \{\text{Chains}[3,X]\} \ . \end{aligned} \tag{3.16}$$

At any point in G_2 elements get added to $\text{Chains}[l]$ only after a Func query and a Func query can add at most one element in $\text{Chains}[j]$. Hence it is easy to see that $|\text{Chains}[j]| \leq 2q^2$. In G_2, for any call to $\mathsf{Complete}$ (immediate or delayed) there is exactly one Func query by Sim. As proved earlier, there are at most $2q^2$ calls to $\mathsf{Complete}$ and each call adds (via $\mathsf{ForceVal}$) at most one element in $|G_k|$. In Phase 1, elements get added to G_k either due to a query to Prim by A_2 or due to $\mathsf{ForceVal}$ calls inside $\mathsf{Complete}$. Moreover in Phase 2, elements also get added to G_k due to $\mathsf{ForceVal}$ inside $\mathsf{Complete}$ or due to $\mathsf{F}^{\text{inner}}$ calls just before calling $\mathsf{Complete}$. Therefore, we have that $|G_k| \leq q + 2q^2$ because there are at most $2q^2$ calls to $\mathsf{Complete}$. ■

By Lemma 13 we have the following lemma.

Lemma 14 $\Delta(\mathsf{G}_1, \mathsf{G}_2) \leq 9q^4/2^{2n}$.

Proof: Let $D = (\mathsf{Sim}, A)$ be a distinguisher that makes at most $3q^2$ ($\geq q_c + 2q^2$) to the oracle Func. By a standard argument, the lemma holds. ■

3.4.4 Indistinguishability of G_2 and G_3

In this section, we prove the indistinguishability of games G_2 and G_3 by defining a map τ that maps (f,ρ) to h such that the output distribution of A in G_2 is not very different from that in G_3. Infact, we prove that for certain pairs (f,ρ) the image $h = \tau(f,\rho)$ is such that the output distributions of both games are identical. We refer to such pairs as good and the absence of certain bad events in $\mathsf{G}_2^{f,\rho}$ makes (f,ρ) a good pair. We define these events next and later in Lemma 21 show that most pairs (f,ρ) are good.

Definition of good pairs

For the following definitions, consider the game G_2 using a pair $(f, \rho) \in \mathcal{F} \times \mathcal{R}$. i.e. $\mathsf{G}_2^{f,\rho}$. We are going to define a few bad events that occur when executing G_2 with randomness fixed to such a pair (f, ρ). We say that a pair (f, ρ) is *good* if no such event occurs in an execution, and otherwise it is *bad*.

Definition 3 *The event* BadP *occurs in an execution of* $\mathsf{G}_2^{f,\rho}$ *if one the following holds,*

1. *Right after a call* $(X_5, X_6) \leftarrow \rho[+, X_0, X_1]$ *either* $G_5[X_5] \neq \perp$ *or* $P[-, X_5, X_6] \neq \perp$.
2. *Right after a call* $(X_0, X_1) \leftarrow \rho[-, X_5, X_6]$ *either* $G_1[X_1] \neq \perp$ *or* $P[+, X_0, X_1] \neq \perp$.

Definition 4 *The event* BadOutside[5] *occurs in an execution of* $\mathsf{G}_2^{f,\rho}$ *if one of the following holds,*

1. *Right after an assignment* $G_1[X] \leftarrow f[1, X]$ *there exist* X_2 *and* $(+, X_0, X_1)$ *such that* $G_2[X_2] \neq \perp$, $P[+, X_0, X_1] \neq \perp$ *and* $G_1[X] = X_2 \oplus X_0$.
2. *Right after an assignment* $G_5[X] \leftarrow f[5, X]$ *there exist* X_4 *and* $(-, X_5, X_6)$ *such that* $G_4[X_4] \neq \perp$, $P[-, X_5, X_6] \neq \perp$ *and* $G_5[X] = X_4 \oplus X_6$.

Definition 5 *The event* BadGutShot *occurs in an execution of* $\mathsf{G}_2^{f,\rho}$ *if one of the following holds,*

1. *Right after an assignment* $G_2[X] \leftarrow f[2, X]$ *there exist* X_1 *and* X_3 *such that* $G_1[X_1] \neq \perp$, $G_3[X_3] \neq \perp$ *and* $G_2[X] = X_1 \oplus X_3$.
2. *Right after an assignment* $G_2[X] \leftarrow f[2, X]$ *there exist* X_1 *and* X_3 *such that* $G_1[X_1] \neq \perp$, $\mathit{Chains}[3, X_3] \neq \perp$ *and* $G_2[X] = X_1 \oplus X_3$.

[5] The names of the events BadOutside and BadGutShot are adopted from straight draws in the card game of Poker.

3. *Right after an assignment* $G_4[X] \leftarrow f[4,X]$ *there exist* X_5 *and* X_3 *such that* $G_5[X_5] \neq \perp$, $G_3[X_3] \neq \perp$ *and* $G_4[X] = X_5 \oplus X_3$.

4. *Right after an assignment* $G_4[X] \leftarrow f[4,X]$ *there exist* X_5 *and* X_3 *such that* $G_5[X_5] \neq \perp$,$Chains[3,X_3] \neq \perp$ *and* $G_4[X] = X_5 \oplus X_3$.

Let the i-th query issued by A_1 be $(\sigma, X_k^i, X_{k+1}^i)$ where $(\sigma,k) \in \{(+,0),(-,5)\}$. Before the control of G_2 is passed to A_2, the tuples $(X_0^i, X_1^i, X_5^i, X_6^i)$ corresponding to the the ith query by A_1 are defined. Since (f,ρ) is sampled at the beginning of G_2 and Sim always replies to queries $(1,X)$ and $(5,X)$ using the values $f[1,X]$ and $f[5,X]$, respectively, we can further define

$$X_2^i = f[1,X_1^i] \oplus X_0^i \ ; \ X_4^i = f[5,X_5^i] \oplus X_6^i \ . \tag{3.17}$$

The following events are assuming $(X_0^i, X_1^i, X_2^i, X_4^i, X_5^i, X_6^i)$ are defined for $i \in [q_c]$. We call such tuples *relevant chains* or *relevant tuples*.

Definition 6 *The event* $\mathsf{BadlyCollide}$ *occurs in an execution of* $\mathsf{G}_2^{f,\rho}$ *if one of the following holds,*

1. $X_2^{i_1} = X_2^{i_2}$ *for* $i_1 \neq i_2 \in [q_c]$.

2. $X_4^{i_1} = X_4^{i_2}$ *for* $i_1 \neq i_2 \in [q_c]$.

Definition 7 *The event* $\mathsf{BadP2}$ *occurs in an execution of* $\mathsf{G}_2^{f,\rho}$ *if one the following holds,*

1. *Right after a call* $(X_5,X_6) \leftarrow \rho[+,X_0,X_1]$ *where* $(X_0,X_1) \neq (X_0^i,X_1^i)$ *for all* $i \in [q_c]$ *there exists* $i_1 \in [q_c]$ *such that* $X_5 = X_5^{i_1}$.

2. *Right after a call* $(X_0,X_1) \leftarrow \rho[-,X_5,X_6]$ *where* $(X_5,X_6) \neq (X_5^i,X_6^i)$ *for all* $i \in [q_c]$ *there exists* $i_1 \in [q_c]$ *such that* $X_1 = X_1^{i_1}$.

Definition 8 *The event* BadOutside2 *occurs in an execution of* $\mathsf{G}_2^{f,\rho}$ *if one of the following holds,*

1. *Right after an assignment* $G_1[X] \leftarrow f[1,X]$ *where* $X \neq X_1^i$ *for all* $i \in [q_c]$ *there exist* $((X_4, X_5, 4), (X_0, X)) \in \mathit{Chains}[1, X]$ *and* $i_1 \in [q_c]$ *such that* $X_2^{i_1} = G_1[X] \oplus X_0$.

2. *Right after an assignment* $G_5[X] \leftarrow f[5,X]$ *where* $X \neq X_5^i$ *for all* $i \in [q_c]$ *there exist* $((X_1, X_2, 1), (X, X_6)) \in \mathit{Chains}[5, X]$ *and* $i_1 \in [q_c]$ *such that* $X_4^{i_1} = G_5[X] \oplus X_6$.

Most pairs (f, ρ) are good

We next show that most pairs $(f, \rho) \in \mathcal{F} \times \mathcal{R}$ are good, i.e., that a randomly sampled (f, ρ) pair is good except with small probability.

For all the following proofs we consider the occurrence of any of the bad events defined above at any point in the execution of G_2. Also, in order to simplify the calculation of the probability that (f, ρ) is not good, we will often alternate viewing G_2 as lazily sampling (f, ρ) (occasionally only partially), rather than pre-sampling it. (This will obviously not affect the probability, as the experiments are equivalent.) For the following calculations, it will be convenient to first observe the following: from Lemma 13 we know that $|P| \leq t(q)$, $|G_i| \leq t(q)$ for all $i \in [5]$ and $|\text{Chains}[l]| \leq t(q)$ where $l \in \{1, 3, 5\}$ and $t(q) = 3q^2$. (We will occasionally write t rather than $t(q)$.)

Lemma 15 $\mathsf{Pr}[\mathsf{BadP}] \leq 2t^2/2^n$.

Proof: For any query to $(X_5, X_6) \leftarrow \rho[+, X_0, X_1]$, BadP occurs if either $P[-, X_5, X_6] \neq \perp$ which happens with probability $t/2^{2n}$ or if $G_5[X_5] \neq \perp$ which happens with probability $t/2^n$. Since $|P| \leq t$ at any point in G_2, there can be at most t such calls and hence by the union bound we have the lemma. ■

Lemma 16 $\mathsf{Pr}[\mathsf{BadOutside}] \leq 2t^3/2^n$.

Proof: For any assignment of the form $G_1[X] \leftarrow f[1, X]$, BadOutside occurs if there exists X_2 and $(+, X_0, X_1)$ such that $G_2[X_2] \neq \perp$, $P[+, X_0, X_1] \neq \perp$ and $G_1[X] = X_2 \oplus X_0$. Since $|P| \leq t$ and $|G_2| \leq t$, there are at most t^2 such pairs for which BadOutside happens with probability $1/2^n$. The symmetric argument holds for assignments of the form $G_5[X] \leftarrow f[5, X]$.

Since $|G_1|, |G_5| \leq t$ at any point in G_2, there can be at most $2t$ such assignments and hence by the union bound we have the lemma. ■

Lemma 17 $\mathsf{Pr}[\mathsf{BadGutShot}] \leq 4t^3/2^n$.

Proof: For any assignment of the form $G_2[X] \leftarrow f[2, X]$, BadGutShot occurs if there exists X_1 and X_3 such that $G_1[X_1] \neq \perp$, $G_3[X_3] \neq \perp$ (Chains$[3, X_3] \neq \perp$) and $G_1[X] = X_2 \oplus X_0$. Since $|\text{Chains}[3]| \leq t$, $|G_3| \leq t$ and $|G_1| \leq t$, there are at most $2t^2$ such pairs for which BadGutShot happens with probability $1/2^n$. Since $|G_2|, |G_4| \leq t$ at any point in G_2, there can be at most $2t$ such assignments and hence by the union bound we have the lemma. ■

Lemma 18 $\mathsf{Pr}[\mathsf{BadlyCollide}] \leq 2q^2/2^n$.

Proof: To find the probability of BadlyCollide, we visualize the game G_2 in the following way. At the beginning of the game, we just sample $\rho \xleftarrow{\$} \mathcal{R}$ and run the game with A_1. After A_1 is done with all its q_c queries, we define X_2^i and X_4^i as follows:

$$X_2^i \leftarrow r_i \oplus X_0^i \; ; \; X_4^i \leftarrow s_i \oplus X_6^i \; , \tag{3.18}$$

where $r_i, s_i \xleftarrow{\$} \{0,1\}^n$ for $i \in [q_c]$. Note that the distribution of X_2^i as defined above is the same as defined in Equation 3.17.

It is easy to see that the probability of collision among the X_2^i values is at most $q^2/2^n$. Similarly, probability of collision among the X_4^i values is at most $q^2/2^n$. Hence by the union bound the lemma holds. ■

For the following proofs, we will visualize G_2 as sampling randomness lazily until A_1 is done with its queries. At this point, we can define the respective relevant tuples $(X_0^i, X_1^i, X_2^i, X_4^i, X_5^i, X_6^i)$. Now the control is passed to A_2 where we sample randomness lazily again. Note that BadP2 and BadOutside2 are defined for calls/assignments not involving the relevant tuples. Hence resorting to lazy sampling in the second stage of the game does not change the output distribution in this variant of G_2.

Lemma 19 $\mathsf{Pr}\left[\mathsf{BadP2}\right] \leq tq/2^n$.

Proof: For any query to $(X_5, X_6) \leftarrow \rho[+, X_0, X_1]$ where $(X_0, X_1) \neq (X_0^i, X_1^i)$ for all $i \in [q_c]$, BadP2 occurs if there exists $i_1 \in [q_c]$ such that $X_5 = X_5^{i_1}$. This happens with probability at most $q/2^n$. Since $|P| \leq t$ at any point in G_2, there can be at most t such calls to ρ and hence by the union bound we have the lemma. ■

Lemma 20 $\mathsf{Pr}\left[\mathsf{BadOutside2}\right] \leq 2qt^2/2^n$.

Proof: For any assignment of the form $G_1[X] \leftarrow f[1, X]$ where $X \neq X_1^i$ for all $i \in [q_c]$, BadOutside2 occurs if there exists $((X_4, X_5, 4), (X_0, X)) \in \mathrm{Chains}[1, X]$ and $i_1 \in [q_c]$ such that $G_1[X] = X_2^{i_1} \oplus X_0$. Since $|\mathrm{Chains}[1]| \leq t$ at any point in G_2, the above happens with probability $tq/2^n$. Similar reasoning holds for assignment of the form $G_5[X] \leftarrow f[5, X]$ where $X \neq X_5^{i_1}$.

Since $|G_1|, |G_5| \leq t$ at any point in G_2, there can be at most $2t$ such assignments and hence by the union bound we have the lemma. ■

Using the union bound with all the above lemmas, and combining this with Lemma 13, we obtain the following final lemma giving us an overall bound on the probability that (f, ρ) is bad.

Lemma 21 $\mathsf{Pr}\left[(f, \rho) \xleftarrow{\$} \mathcal{F} \times \mathcal{R} : (f, \rho) \textit{ is bad}\right] \leq 13t^3/2^n \leq 351q^6/2^n$.

Properties of good executions

By Lemma 21, we know that most pairs (f, ρ) are good, and now we want to focus on executions in $\mathsf{G}_2^{f,\rho}$ for such good pairs (so-called "good executions"). This then allows us to define a map τ that maps (f, ρ) to h such that the executions of $\mathsf{G}_2^{f,\rho}$ and G_3^{h} are identical. To formalize this, we first define a notion of a transcript of such an execution.

Transcript of Game G_i: We define the transcripts of an execution of G_i as a sequence of tuples $T_1, T_2, \ldots$ recording each access to the tables P and G made during this execution. In particular, each T_j is one of the following:

1. $T_j = (Y_j, k_j, X_j)$ where $k_j \in [5]$ such that $\mathsf{F}^{\mathsf{inner}}$ was called on input (k_j, X_j) and Y_j was the value returned by $\mathsf{F}^{\mathsf{inner}}$.

2. $T_j = (Y_j, k_j, X_j)$ where $k_j \in \{2, 3, 4\}$ such that $\mathsf{ForceVal}$ was called on input (X_j, Y_j, k_j).

3. $T_j = (\sigma, X_k^j, X_{k+1}^j, X_{k'}^j, X_{k'+1}^j)$, where $(\sigma, k, k') \in \{(+, 0, 5), (-, 5, 0)\}$ such that Func was called on input $(\sigma, X_k^j || X_{k+1}^j)$ and $(X_{k'}^j, X_{k'+1}^j)$ was the value returned by Func.

The subtranscript $T[1 : j]$ which is the sequence $T_1, \ldots, T_j$. Next we describe a few properties of the execution and its transcript corresponding to a good pair (f, ρ).

Some notational conventions. In the following proofs we make references to line numbers. If not specified then it must be assumed that the line numbers are w.r.t. the Sim's pseudocode shown in Figure 3.9. Also, for ease of notation, we use $x \in T$ and $T[x] \neq \perp$ interchangeably but both denote that the key x is in the table T. For eg: $X \in G_k$ implies $G_k[X] \neq \perp$.

No Chains overwrites. Our first case is going to deal entering values in Chains$[3, X]$ for some X. We show that this value is never overwritten during a good execution.

Lemma 22 *For any assignment of the form* Chains$[3, X] \leftarrow (k, X_k)$ *in* $\mathsf{G}_2^{f,\rho}$ *(lines 13,25 in Figure 3.9),* Chains$[3, X] = \perp$ *just before the assignment.*

Proof: Consider an assignment Chains$[3, X] \leftarrow (5, X_5)$. This occurs only on a query $(2, X_2)$ by A_2 such that $G_2[X_2] = \perp$ just before the query. To reply to $(2, X_2)$, Sim assigns $G_2[X_2] \leftarrow f[2, X_2]$ and considers all tuples in $G_1 \times \{X_2\}$. For each of these tuples (X_1, X_2) it either schedules the partial chain $(X_1, X_2, 1)$ for immediate completion (line 9-10) or delays its completion (lines 11-13). The assignment to Chains$[3, X]$ can only occur when the chain's completion gets delayed (lines 11,23).

Now consider $X = G_2[X_2] \oplus X_1$, since (f, ρ) is a good pair and the event BadGutShot does not occur Chains$[3, X] = \perp$. Otherwise on assignment $G_2[X_2] \leftarrow f[2, X_2]$ there would exist X_1 and X such that $G_1[X_1] \neq \perp$, Chains$[3, X] \neq \perp$ where $X = G_2[X_2] \oplus X_1$. The other assignment Chains$[3, X] \leftarrow (1, X_1)$ is symmetrically handled. Hence the lemma holds. ■

No overwrites in G_k. Next we prove that in $\mathsf{G}_2^{f,\rho}$ where (f, ρ) is a good pair, there are no overwrites in G_k. In particular, this will mean that whenever Sim is attempting to force a value (via ForceVal) it can do so without making anything inconsistent.

Lemma 23 *For any call to* ForceVal$(X, \cdot, l)$ *in* $\mathsf{G}_2^{f,\rho}$*, we have* $G_l[X] = \perp$ *just before the call.*

Proof: A call to ForceVal is triggered only from inside Complete (lines 43,48). Moreover, Sim calls the procedure Complete only when it is attempting to complete a partial chain which was not completed before. If the lemma fails to hold then there

exists a call to Complete during which it fails. Consider such a call to Complete and let C be the corresponding partial chain that is being completed. Furthermore, note that $C \notin$ CompletedChains just before the Complete call. We analyse the case when the partial chain C is of the form $(X_1, X_2, 1)$. The other instance when $C = (X_4, X_5, 4)$ can be handled symmetrically, and is omitted.

Therefore, let us assume that during the completion of $C = (X_1, X_2, 1)$, the lemma fails to hold for one of the ForceVal calls. There are two cases in which Complete can get called on $C = (X_1, X_2, 1)$ and we analyse them next.

- Case A: Complete$(C = (X_1, X_2, 1), \cdot)$ was called from line 10 of Figure 3.9. We refer to such call to Complete as an *immediate* chain completion. Immediate chain completions happen only in Phase 1 (before Sim runs AllComplete), as they are triggered due to a Prim query by A_2.

  Note that just before the call to Complete$((X_1, X_2, 1), \cdot)$ there must have been a query $(2, X_2)$ to Prim by A_2 such that $G_2[X_2] = \perp$ before the query. To respond to $(2, X_2)$ query of A_2, Sim assigns $G_2[X_2] \leftarrow f[2, X_2]$. After the assignment, Sim iterates through every tuple in $G_1 \times \{X_2\}$ (line 4 of Figure 3.9) to either schedule an immediate completion of C (lines 9-10) or delay its completion (lines 11-13). Since this call to Complete was made from line 10, we must have $G_1[X_1] \neq \perp$, $G_5[X_5] \neq \perp$ where $(X_5, X_6) = P[+, X_0, X_1]$, $X_0 = G_2[X_2] \oplus X_1$. Furthermore, let $X_3 = G_2[X_2] \oplus X_1$ and $X_4 = G_5[X_5] \oplus X_6$ where $G_2[X_2]$ is assigned to $f[2, X_2]$ just before this call to Complete. To complete the chain C, Complete will force $G_3[X_3]$ and $G_4[X_4]$ through ForceVal$(X_3, \cdot, 3)$ and ForceVal$(X_4, \cdot, 4)$ respectively. If the lemma fails to hold for C then either $G_3[X_3] \neq \perp$ or $G_4[X_4] \neq \perp$ before their respective ForceVal calls.

  First, we argue that $G_3[X_3] = \perp$ before the call ForceVal$(X_3, \cdot, 3)$. This is because

the pair (f, ρ) is good and the event BadGutShot does not occur on assignment $G_2[X_2] \leftarrow f[2, X_2]$. Furthermore, Chains$[3, X_3] = \perp$ by the same argument. This ensures that by forcing $G_3[X_3]$ to a value (inside ForceVal) we are not making any previously completed chains inconsistent ($G_3[X_3] = \perp$) nor running into the risk of making this chain completion inconsistent due to a future delayed chain completion (Chains$[3, X_3] = \perp$).

Therefore, if the lemma fails to hold for C, then it must be that $G_4[X_4] \neq \perp$ before the call to ForceVal$(X_4, \cdot, 4)$. Note that in order to complete the chain C, Sim queries Func with $(+, X_0||X_1)$ (line 7). It can either be that $(+, X_0||X_1)$ was a fresh query, that is, $P[+, X_0, X_1] = \perp$ before Sim's query to Func or that it was already in the table P before Sim's query. Furthermore, a tuple gets added to the table P either when Sim makes a query to Func or A_1 makes a query to Func. Assuming, $G_4[X_4] \neq \perp$, we are going to analyse when/how the tuple $(+, X_0, X_1)$, corresponding to query by Sim (to Func in line 7), was added to the table P and arrive at a contradiction to either the goodness of the pair (f, ρ) or that $(X_1, X_2, 1) \notin$ CompletedChains.

- Case A.1: $(+, X_0, X_1)$ gets added to P due to the Func query in line 7. In this case, the query $(+, X_0||X_1)$ to Func is such that $P[+, X_0, X_1] = \perp$. Therefore, to reply to $(+, X_0||X_1)$, Func accesses $\rho[+, X_0, X_1]$ (as $P[+, X_0, X_1] = \perp$) to get (X_5, X_6) and returns it. However, since we are considering an immediate chain completion, we know that $G_5[X_5] \neq \perp$. This means the event BadP occurs when $\rho[+, X_0, X_1]$ is accessed, which is a contradiction as (f, ρ) is a good pair. Therefore, $(+, X_0, X_1) \in P$ even before Func was called in line 7.
- Case A.2: $(+, X_0, X_1)$ was added to P due to an earlier query to Func by Sim. Sim's queries to Func are distinct therefore if $(+, X_0, X_1)$ got added to P due to a Sim query then Sim must have previously queried Func$(-, X_5||X_6)$. This

could only happen on a query $(4, X_4)$ by A_2 where $X_4 = G_5[X_5] \oplus X_6$. Sim assigns $G_4[X_4] \leftarrow f[4, X_4]$. After the assignment, Sim would have iterated through all tuples in $\{X_4\} \times G_5$ (line 16). It would also consider (X_4, X_5) and make the $\mathsf{Func}(-, X_5, X_6)$ query.

If $X_1 \in G_1$ at this point, this chain $(X_4, X_5, 4)$ would have been immediately completed (line 21-22) and hence $C = (X_1, X_2, 1)$ would already be completed even before the query $(2, X_2)$ by A_2 which is a contradiction as $C \notin$ CompletedChains.

If $X_1 \notin G_1$ at this point then $(X_4, X_5, 4)$'s completion is delayed (line 23-25) where Chains$[3, X_3] \leftarrow (1, X_1)$ and $((X_4, X_5, 4), (X_0, X_1))$ gets inserted in Chains$[1, X_1]$. But as $G_1[X_1] \neq \perp$ before the A_2's query $(2, X_2)$ and we are still in Phase 1, it can only be the case that A_2 queried $(1, X_1)$ to Prim or $(3, X_3')$ to Prim such that Chains$[3, X_3'] = (1, X_1)$ and $G_3[X_3'] = \perp$ whichever earlier. Here, X_3' corresponds to one of the chains C' such that $(C', (U', V')) \in$ Chains$[1, X_1]$ [6]. In either case, Complete gets called on $(X_4, X_5, 4)$ and hence $(X_1, X_2, 1)$ would be completed even before the query $(2, X_2)$ by A_2 which is again contradiction to $C \notin$ CompletedChains. Therefore, the tuple $(+, X_0, X_1)$ corresponding to the query by Sim (to Func) can only get added to P due to a query (to Func) by A_1.

Infact, recall that a $(2, X_2)$ query by A_2 can trigger more than one immediate chain completions as Sim loops over all newly formed partial chains $C' = (X_1', X_2, 1)$ where $X_1' \in G_1$. It follows from the arguments made in cases A.1 and A.2 that all of Sim's queries $(+, X_0' || X_1')$ are such that the tuple $(+, X_0', X_1')$ was added to P due to a query by A_1.

[6] By Lemma 22 there are no overwrites in Chains$[3, X]$. This ensures that a Prim query to $(3, X_3')$ such that $G_3[X_3'] = \perp$ will lead to a call to $\mathsf{Sim}(5, X_5)$ (line 32). We will use this argument in the future without repeating this explanation.

– Case A.3: $(+, X_0, X_1)$ was added to P due to a query by A_1. The only remaining case for the tuple $(+, X_0, X_1)$ is that it was added to P due to a query by A_1. Therefore, the tuple $(X_0, X_1, X_3, X_4, X_5, X_6)$ falls on a relevant chain, that is, there exists $i \in [q_c]$ such that $X_j = X_j^i$ for $j \in \{0, 1, 2, 4, 5, 6\}$, thereby making C a relevant partial chain. As stated above, a $(2, X_2)$ query by A_2 can trigger immediate completion of more than one partial chains. As discussed at the end of the case A.2, all these partial chains are relevant chains.

Recall that if the lemma fails to hold for C, then $G_4[X_4^i] \neq \perp$ before the $\mathsf{ForceVal}(X_4^i, \cdot, 4)$ call. Note that an element X gets added to a table G_4 either due to ForceVal call (issued by Sim from inside a call to Complete) or due to a direct query $(4, X)$ by A_2. Assuming that $G_4[X_4^i] \neq \perp$ before the ForceVal call (i.e., X_4^i is in the table P), we are going to analyse when/how X_4^i was added to G_4 and arrive at a contradiction to either the goodness of the pair (f, ρ) or that $(X_1, X_2, 1) \notin$ CompletedChains.

* Case A.3.1: X_4^i was added due to one of the Complete calls due to $(2, X_2)$. A query to $(2, X_2)$ by A_2 may trigger multiple calls to Complete. However, we just observed that all these calls must correspond to relevant chains. Therefore, if X_4^i gets added to G_4 due to a Complete call triggered by the query $(2, X_2)$ on some chain $C' \neq C$, this would imply that two relevant chains C and C' share their X_4s and the event BadlyCollide occurs. This contradicts the fact that (f, ρ) is a good pair.

* Case A.3.2: X_4^i was added due to a Complete call before A_2's query $(2, X_2)$. Let us assume that it was due to a call to Complete on some $(X_1', X_2', 1) \neq (X_1, X_2, 1)$. Again as (f, ρ) is a good pair, $(X_1', X_2', 1)$ cannot be a relevant chain (relevant chains donot share X_2s). Hence, when Sim queries

$(+, X'_0||X'_1)$ where $X'_0 = G_1[X'_1] \oplus X'_2$, Func replies to $(+, X'_0||X'_1)$ by accessing $\rho[+, X'_0, X'_1]$. Then, (f, ρ) being good, the events BadP and BadP2 do not occur. This ensures that $X'_5 \notin G_5$ and also that $X'_5 \neq X_5^{i_1}$ for all $i_1 \in [q_c]$. Since $X'_5 \notin G_5$, chain $(X'_1, X'_2, 1)$'s completion gets delayed (lines 11-13). We are inside the call to Complete on $(X'_1, X'_2, 1)$ and Sim must be executing lines 27-30 for $(5, X'_5)$. $G_5[X'_5]$ is assigned to $f[5, X'_5]$ and then Complete is called on C' where $(C', (X'_5, X'_6)) \in \text{Chains}[5, X'_5]$. If $G_5[X'_5] \oplus X'_6 = X_4^i$ (as we have assumed) then BadOutside2 will occur on assignment $G_5[X'_5]$ where $X_5 \neq X_5^{i_1}$, which is a contradiction as (f, ρ) is a good pair.

* Case A.3.3: X_4^i gets added to G_4 due to a query $(4, X_4^i)$ by A_2. The only remaining case in which X_4^i can get added to G_4 is if A_2 makes a Prim query $(4, X_4^i)$. Note that such a query $(4, X_4^i)$ (if at all) must occur before $(2, X_2)$. Furthermore, when A_2 queries $(4, X_4^i)$, X_5^i must already be in G_5. Otherwise, BadOutside occurs on the assignment $G_5[X_5^i] \leftarrow f[5, X_5^i]$. This leaves us with the state that $X_5^i \in G_5$ when A_2 queries $(4, X_4^i)$ and $G_4[X_4^i] = \perp$ just before A_2's query $(4, X_4^i)$. We have already argued for a similar case in Case A.2 where we derive a contradiction to $(X_1, X_2, 1) \notin$ CompletedChains before the query $(2, X_2)$ by A_2.

Therefore, in all the cases where $G_4[X_4^i] \neq \perp$ before the ForceVal$(X_4^i, \cdot, 4)$ leads to a contradiction.

Therefore, if the lemma fails to hold for C then it must be the case that the Complete called on C is from line 30. We analyse this case next.

- Case B: Complete$(C = (X_1, X_2, 1), \cdot)$ was called from line 30 of Figure 3.9. We refer such a call to Complete as *delayed* chain completion. Delayed chain completions

can happen in either of the Phases in G_2.

When C's completion was delayed (lines 11-13), $\text{Chains}[3, X_3] = (5, X_5)$ and $(X_1, X_2, 1) \in \text{Chains}[5, X_5]$ where $X_3 = G_2[X_2] \oplus X_1$ and $(X_5, X_6) = P[+, X_0, X_1]$ and $X_0 = G_1[X_1] \oplus X_2$.

To complete the chain C, Complete will force $G_3[X_3]$ and $G_4[X_4]$ (X_4 is not yet defined) through $\mathsf{ForceVal}(X_3, \cdot, 3)$ and $\mathsf{ForceVal}(X_4, \cdot, 4)$ respectively. If the lemma fails to hold for C then either $G_3[X_3] \neq \perp$ or $G_4[X_4] \neq \perp$ before their respective ForceVal calls.

First, we argue that before the call $\mathsf{ForceVal}(X_3, \cdot, 3)$, $G_3[X_3] = \perp$. Prior to this (delayed) call to Complete, $\text{Chains}[3, X_3] = (5, X_5)$ (because there are no overwrites in $\text{Chains}[3, \cdot]$ by Lemma 22). Such a (delayed) call to Complete is triggered on an A_2 query to either $(5, X_5)$ or $(3, X'_3)$ or within AllComplete procedure in the Phase 2 such that $\text{Chains}[3, X'_3] = (5, X_5)$, whichever earlier. Here X'_3 corresponds to any chain C' such that $(C', \cdot) \in \text{Chains}[5, X_5]$. Let us consider the earliest point in the execution when either A_2 queries $(5, X_5)$ or $(3, X'_3)$ or there is a call to AllComplete such that $\text{Chains}[3, X'_3] = (5, X_5)$. We assume that $\text{Chains}[3, X'_3]$ was assigned to $(5, X_5)$ due to a Func query on some (X'_0, X'_1) where $G_1[X'_1] \neq \perp$, $G_2[X'_2] \neq \perp$, $X'_0 = G_1[X'_1] \oplus X'_2$ and $X'_3 = G_2[X'_2] \oplus X'_1$. The other case when Func query is on some (X'_5, X'_6) is symmetrically handled.

We claim that for all X'_3 such that $\text{Chains}[3, X'_3] = (5, X_5)$, $G_3[X'_3] \neq \perp$. The claim may not hold because X'_3 got added to G_3 during a Complete call on some chain C'' by Sim. Note that C'' can be of the form $(X''_1, X''_2, 1)$ or $(X''_4, X''_5, 4)$ as Complete calls on either can lead to $\mathsf{ForceVal}(X'_3, \cdot, 3)$. We refute the case when $C'' = (X''_1, X''_2, 1)$ and the other case can be similarly handled. More concretely for C'' it is the case

that $G_2[X_2''] \oplus X_1'' = X_3'$. The chain C'' could be considered for completion only on a query $(2, X_2'')$ by A_2. At this point $G_1[X_1''] \neq \perp$ and $G_2[X_2'']$ was assigned to $f[2, X_2'']$. If A_2 issues $(2, X_2'')$ after $(2, X_2')$ and $G_2[X_2''] \oplus X_1'' = G_2[X_2'] \oplus X_1' = X_3'$ then the event BadGutShot is triggered on assignment $G_2[X_2''] \leftarrow f[2, X_2'']$. Similarly, if A_2 issues $(2, X_2'')$ earlier then BadGutShot is triggered on the assignment $G_2[X_2'] \leftarrow f[1, X_2']$. Since (f, ρ) is a good pair, our claim holds for all X_3'. Hence $G_3[X_3] = \perp$ before its ForceVal call.

Now, the partial chain C can either be a relevant chain or a non-relevant chain. Below, we analyse both cases separately.

- Case B.1: $C = (X_1, X_2, 1)$ is a non-relevant chain. If C is not a relevant chain then $X_5 \neq X_5^i$ for all $i \in [q_c]$. Otherwise the event BadP2 would have occured on the Func query $(+, X_0||X_1)$ made by Sim to schedule the delayed completion of C, where $X_0 = G_1[X_1] \oplus X_2$. Since this is a non-relevant chain's delayed completion, $G_5[X_5]$ gets assigned to $f[5, X_5]$ (line 27) just before the Complete call. Then for all $(C', (X_5, X_6')) \in \text{Chains}[5, X_5]$ we must have $G_4[X_4'] = \perp$ where $X_4' = G_5[X_5] \oplus X_6'$. Otherwise there would exist $(-, X_5, X_6') \in P$ and $X_5 \in G_5$ immediately after the assignment $G_5[X_5] \leftarrow f[5, X_5]$ such that $G_4[X_4'] \neq \perp$, which contradicts the goodness of the pair (f, ρ). Since for every X_4', $G_4[X_4'] = \perp$, therefore $G_4[X_4] = \perp$ as well.

  Therefore, if the lemma fails to hold for C, then it can only be that the Complete call (for the lemma does not hold) was a delayed completion of a relevant chain C. We analyse this case next.

- Case B.2: $C = (X_1, X_2, 1)$ is a relevant chain. If C is a relevant chain then $(X_5 = X_5^i \; \exists \; i \in [q_c])$ $\text{Chains}[5, X_5^i]$ only consists of tuples $(C', (X_5^i, X_6'))$ such that the partial chain C' is relevant. Otherwise, if there exist a non-relevant

partial chain C' then the event BadP2 would have occured on the Func query made by Sim to schedule the completion of C'. Now, since all C' are relevant chains then $X'_4 = G_5[X^i_5] \oplus X'_6$ ($G_5[X^i_5]$ is defined before the control is transferred to A_2) for every chain C' in Chains$[5, X^i_5]$ are distinct, as BadlyCollide does not occur.

Therefore, if the lemma fails to hold for C, then $G_4[X^i_4] \neq \perp$ before the ForceVal$(X^i_4, \cdot, 4)$ call where $X^i_4 = G_5[X^i_5] \oplus X^i_6$. We, infact, argue something stronger, that is, for all $(C', (X^i_5, X'_6)) \in$ Chains$[5, X^i_5]$, that $G_4[X'_4] = \perp$ where $X'_4 = G_5[X^i_5] \oplus X'_6$. Let us assume that $G_4[X'_4] \neq \perp$ before the corresponding call to Complete on C'. As seen in case $A.3$, X'_4 can get added to G_4 either during a chain completion performed by Sim or a direct query $(4.X'_4)$ by A_2. Similar to the argument presented in case A.3.3. (for direct query) and the fact that (f, ρ) is a good pair (relevant chains cannot share X_4s), if $G_4[X'_4] \neq \perp$ then X'_4 got added to G_4 during the completion of some non-relevant chain $C'' = (X''_1, X''_2, 1)$. Furthermore, the corresponding X''_5 (of chain C'') is such that $X''_5 \neq X^i_5$ as the event BadP2 does not occur in a good execution. Therefore during the completion of chain C'', $G_5[X''_5]$ is assigned to $f[5, X''_5]$ and if $G_5[X''_5] \oplus X''_6 = X'_4$ then the event BadOutside2 would occur, contradicting that (f, ρ) is a good pair. Therefore, we conclude that $G_4[X'_4] = \perp$ for every X'_4 (as described above). Hence, $G_4[X_4] = \perp$ before the call ForceVal$(X_4, \cdot, 4)$ during the delayed completion of $C = (X_1, X_2, 1)$.

Therefore even for a delayed chain completion of $C = (X_1, X_2, 1)$, the calls to ForceVal$(X_l, \cdot, l)$ are such that $G_l[X_l] = \perp$ before the call.

We have now analysed all the cases for that call to Complete on C for which we assumed the lemma fails to hold. Since, we have arrived at a contradiction in each of

these cases, the lemma indeed holds. ■

Next, we define a predicate isTrue on the transcript (described in Figure 3.15) which captures that all queries made to Func $(\sigma, X_k, X_{k+1}, X_{k'}, X_{k'+1})$ are such that $(X_{k'}, X_{k'+1})$ look like the result of a Feistel Ψ_5 evaluation on (σ, X_k, X_{k+1}) given the transcript T of the execution. We prove that the isTrue returns true on a transcript T corresponding to an execution with a good pair (f, ρ).

Lemma 24 *For all* $(\sigma, X_k, X_{k+1}, X_{k'}, X_{k'+1}) \in T$ *that correspond to* Func *queries by* A_1, $\mathsf{Check}(\sigma, X_k, X_{k+1}, X_{k'}, X_{k'+1})$ *returns* true *at the end of the execution for* $(\sigma, k, k') \in \{(+, 0, 5), (-, 5, 0)\}$.

Proof: Consider any tuple $(\sigma, X_k, X_{k+1}, X_{k'}, X_{k'+1}) \in T$ which corresponds to the query by A_1. We prove the lemma for $(\sigma, k, k') = (+, 0, 5)$, that is, tuples of the form $(+, X_0, X_1, X_5, X_6)$. The other case of $(\sigma, k, k') = (+, 0, 5)$ can be handled symmetrically, and is omitted.

Since A_2 makes all primitive queries corresponding to the construction queries by A_1, then there exists at least one occurence of $(Y_1, 1, X_1)$ and $(Y_5, 5, X_5)$ in T. Consider the first occurrences of both the tuples, furthermore consider the earliest of the two.

Let us w.l.o.g. assume that the earlier tuple in the sequence T is $(Y_1, 1, X_1)$. Since this is the earliest call to $\mathsf{F}^{\text{inner}}(1, X_1)$, inside $\mathsf{F}^{\text{inner}}$, $G_1[X_1]$ gets assigned to $f[1, X_1]$. This defines $X_2 = G_1[X_1] \oplus X_0$ such that $G_2[X_2] = \perp$ (otherwise on the assignment $G_1[X_1] \leftarrow f[1, X_1]$ the event BadOutside occurs in an execution with a good pair (f, ρ)). Hence the tuple $(Y_2, 2, X_2)$ hasn't occured in T at this point. Since A_2 makes all primitive queries there is at least one occurrence of $(Y_2, 2, X_2)$ in T.

At this point there are two cases possible: one where $(Y_5, 5, X_5)$ is the earliest tuple after $(Y_1, 1, X_1)$ in T and the other where $(Y_2, 2, X_2)$ is the earliest tuple after $(Y_1, 1, X_1)$ in T. We consider both these cases and show that Check returns true.

- Case A: $(Y_5, 5, X_5)$ is earlier. Since this is also the earliest call to $\mathsf{F}^{\text{inner}}(5, X_5)$, we have the assignment $G_5[X_5] \leftarrow f[5, X_5]$. This defines $X_4 = G_5[X_5] \oplus X_6$ such that $G_4[X_4] = \perp$ (otherwise on the assignment $G_5[X_5] \leftarrow f[5, X_5]$ the event BadOutside would occur (in an execution with a good pair (f, ρ))). Hence the tuple $(\cdot, 4, X_4)$ hasn't occured in T yet. Since A_2 makes all primitive queries, there exists at least one ocurrence of $(Y_4, 4, X_4)$ in T. Finally, we consider the earlier of $(Y_2, 2, X_2)$ and $(Y_4, 4, X_4)$. We analyse the case when $(Y_2, 2, X_2)$ is the earlier tuple in T. The other case when $(Y_4, 4, X_2)$ occurs earlier than $(Y_2, 2, X_2)$ is symmetrical, and is omitted.

  We claim (prove below) that the tuple $(\cdot, 2, X_2)$ corresponds to a call to $\mathsf{F}^{\text{inner}}$ on $(2, X_2)$. Moreover it was due to a query $(2, X_2)$ by A_2 to Prim. And since this is the earliest occurrence of $(\cdot, 2, X_2)$, $G_2[X_2] = \perp$ before the call to $\mathsf{F}^{\text{inner}}$. To respond to A_2's query, Sim (lines 3-7 in Figure 3.9) issues a call to $\mathsf{F}^{\text{inner}}$ where $G_2[X_2]$ is assigned to $f[2, X_2]$. After the call to $\mathsf{F}^{\text{inner}}$, Sim considers all possible tuples in $G_1 \times \{X_2\}$ (line 4) and makes appropriate Func queries. Therefore, it issues $\mathsf{Func}(+, X_0||X_1)$ and gets (X_5, X_6). Since $G_5[X_5] \neq \perp$ (as the tuple $(Y_5, 5, X_5)$ has already occurred in T), Complete is called on input $((X_1, X_2, 1), (X_5, X_6))$ (lines 9-10 in Figure 3.9). It is easy to see that after Complete returns, Check on $(+, X_0, X_1, X_5, X_6)$ is returns true. From Lemma 23 we know that there are no overwrites in G_k. Hence, Check will return true even at the end of the execution.

  Now, we get back to proving our claim that $(\cdot, 2, X_2)$ corresponds to a call to $\mathsf{F}^{\text{inner}}$. Moreover this was because A_2 queries $(2, X_2)$. Let us assume that $(\cdot, 2, X_2)$ does not correspond to a call to $\mathsf{F}^{\text{inner}}$. Therefore, it corresponds to a call to $\mathsf{ForceVal}(X_2, \cdot, 2)$. Since ForceVal is called only from inside Complete, there exists a partial chain $C' = (X'_4, X'_5, 4)$ during whose completion the ForceVal call was made. Since (f, ρ) is a good pair, all relevant chains have distinct X_2s (as the event

BadlyCollide does not occur). Since $(X_1, X_2, 1)$ is a relevant chain (because we are only concerned with relevant chains in this proof) then C' cannot be a relevant chain.

Since C' is not a relevant chain, Sim at some point must have queried $\mathsf{Func}(-, X'_5, X'_6)$ to get (X'_0, X'_1). The query is answered by Func using ρ. Since (f, ρ) is a good pair it is the case that $G_1[X'_1] = \perp$ ($\overline{\mathsf{BadP}}$) and $X'_1 \neq X^i_1$ for $i \in [q_c]$ ($\overline{\mathsf{BadP2}}$). Moreover $G_1[X'_1] = \perp$ ensures that $(X'_4, X'_5, 4)$ is delayed for completion. Currently we are inside the call to Complete on $(X'_4, X'_5, 4)$. Hence, $G_1[X'_1] \leftarrow f[1, X'_1]$ and if $G_1[X'_1] \oplus X'_0 = X_2$ we would trigger the event BadOutside2 which contradicts that the pair (f, ρ) is good. Hence $(Y_2, 2, X_2)$ cannot correspond to a call to $\mathsf{ForceVal}(X_2, \cdot, 2)$.

Therefore for Case A, we have proved that the lemma holds.

- Case B: $(Y_2, 2, X_2)$ is earlier. By the exact same argument as above, we know that $(Y_2, 2, X_2)$ corresponds to a call to $\mathsf{F}^{\text{inner}}$ and this was due to the query $(2, X_2)$ by A_2. Since this is the earliest call $\mathsf{F}^{\text{inner}}(2, X_2)$, inside the call to $\mathsf{F}^{\text{inner}}$, $G_2[X_2]$ is assigned to $f[2, X_2]$ and then Sim considers all tuples (lines 4-7) in $G_1 \times \{X_2\}$ and makes appropriate queries to Func. Since $(Y_2, 2, X_2)$ is the earlier query we know that $G_5[X_5] = \perp$ at this point. Therefore $(X_1, X_2, 1)$ is delayed for completion (lines 11-13). Moreover, $\text{Chains}[3, X_3] = (5, X_5)$ where $X_3 = G_2[X_2] \oplus X_1$ and $((X_1, X_2, 1)), (X_5, X_6)) \in \text{Chains}[5, X_5]$. Since A_2 issues all primitive queries then there exists at least one occurence of $(Y_3, 3, X_3)$ in T. Like before, we will consider the earlier of the two $(Y_3, 3, X_3)$ and $(Y_5, 5, X_5)$. Unlike before, the cases here are not symmetrically and hence we explicitly consider them in the following:

  - Case B.1: $(Y_5, 5, X_5)$ is earlier. Since this is the earliest occurence of $(Y_5, 5, X_5)$ in T or equivalently the earliest call to $\mathsf{F}^{\text{inner}}$ on $(5, X_5)$, $G_5[X_5] = \perp$. This

call to $\mathsf{F}^{\text{inner}}$ could be because of a direct query $(5, X_5)$ by A_2 to Prim or due to a direct query $(3, X_3')$ by A_2 to Prim such that $\text{Chains}[3, X_3'] = (5, X_5)$. It cannot be the case that $(Y_5, 5, X_5)$ was due to a call to $\mathsf{F}^{\text{inner}}$ by Sim from inside AllComplete. This is because A_2 makes all primitives queries and hence it would query $(5, X_5)$ before Sim gets to run AllComplete. In either case, Sim is going to execute lines 27-30 on $(5, X_5)$. Hence, Complete is called on $((X_1, X_2, 1), (X_5, X_6))$ in $\text{Chains}[5, X_5]$. It is easy to see that after Complete returns, Check on $(+, X_0, X_1, X_5, X_6)$ returns true. From Lemma 23 we know that there are no overwrites in G_k. Hence, Check will return true even at the end of the execution.

– Case B.2: $(Y_3, 3, X_3)$ is earlier. The tuple $(Y_3, 3, X_3)$ occurs in T either due to a $\mathsf{F}^{\text{inner}}$ query $(3, X_3)$ or due to a ForceVal call. We consider these two cases separately below:

  * Case B.2.1: $(Y_3, 3, X_3)$ corresponds to a ForceVal call. A call to ForceVal occurs only from inside Complete. Let us assume that Complete was called on $(C', (U, V))$. Furthermore, we let $C' = (X_1', X_2', 1)$ and $(U, V) = (X_5', X_6')$ (the other case is symmterical). Therefore, the chain C' would have been considered by Sim for completion on a query $(2, X_2')$ by A_2. Infact this must correspond to an $\mathsf{F}^{\text{inner}}$ call to $(2, X_2')$ and hence the tuple $(Y_2', 2, X_2')$ occurs in T.

    If $(Y_2', 2, X_2')$ occurs later in the sequence T than $(Y_2, 2, X_2)$ and $G_2[X_2'] \oplus X_1' = X_3$ then on assignment $G_2[X_2'] \leftarrow f[2, X_2']$ there exists $X_1' \in G_1$ and $(3, X_3) \in \text{Chains}$ such that $X_3 = G_2[X_2'] \oplus X_1'$. Since this (f, ρ) is a good pair, the event BadGutShot does not occur. Therefore, $(Y_2', 2, X_2')$ occurs before $(Y_2, 2, X_2)$ in T. If $(Y_2', 2, X_2')$ occurs earlier then depending

on whether $X'_5 \in G_5$, either X'_3 gets added to G_3 or $\text{Chains}[3, X'_3] \leftarrow (5, X_5) \neq \perp$ where $X'_3 \leftarrow G_2[X'_2] \oplus X'_1$. Finally, moving ahead in the T to the point where $(Y_2, 2, X_2)$ occurs and $G_2[X_2] \leftarrow f[2, X_2]$ if $X_3 = X'_3$ where $X_3 \leftarrow G_2[X_2] \oplus X_1$ we would trigger $\mathsf{BadGutShot}$.

Since the pair (f, ρ), the tuple $(Y_3, 3, X_3)$ in T cannot correspond to a call to $\mathsf{ForceVal}$. The only other case, that is, $(Y_3, 3, X_3)$ corresponding to $\mathsf{F}^{\text{inner}}$ call is discussed next.

* Case B.2.2: $(Y_3, 3, X_3)$ corresponds to $\mathsf{F}^{\text{inner}}$ query. The only case where the tuple $(Y_3, 3, X_3)$ may correspond to an $\mathsf{F}^{\text{inner}}$ query is due to a direct query $(3, X_3)$ by A_2. However, as $\text{Chains}[3, X_3] = (5, X_5)$, on a direct query $(3, X_3)$ Sim executes lines 31-32 and ends up calling itself on $(5, X_5)$. Therefore, the tuple $(Y_5, 5, X_5)$ occurs earlier than $(Y_3, 3, X_3)$ which is a contradiction. Hence, the tuple $(Y_3, 3, X_3)$ cannot correspond to $\mathsf{F}^{\text{inner}}$ query as well.

Therefore, the tuple $(Y_3, 3, X_3)$ (in an execution with a good pair (f, ρ)) cannot occur in T before $(Y_5, 5, X_5)$. And the other case is already analysed in case B.1.

Therefore for case B, we have proved that the lemma holds.

■

Lemma 25 *For all* $(\sigma, X_k, X_{k+1}, X_{k'}, X_{k'+1}) \in T$, $\mathsf{Check}(\sigma, X_k, X_{k+1}, X_{k'}, X_{k'+1})$ *returns* true *at the end of the execution where* $(\sigma, k, k') \in \{(+, 0, 5), (-, 5, 0)\}$. *Moreover,* $\mathsf{isTrue}(T) = \mathsf{true}$.

Proof: In Lemma 24 we have already argued about the tuple $(\sigma, X_k, X_{k+1}, X_{k'}, X_{k'+1})$ that correspond to queries by A_1. Therefore, we will focus only on tuples that correspond

to queries by Sim to Func. Moreover, when proving Lemma 24 we have seen that for every tuple corresponding to A_1 there is a tuple in T corresponding to the Func query made by Sim.

Therefore, we focus on tuples $(\sigma, X_k, X_{k+1}, X_{k'}, X_{k'+1})$ which correspond to non-relevant chains. We prove the lemma for $(\sigma, k, k') = (+, 0, 5)$, that is, tuples of the form $(+, X_0, X_1, X_5, X_6)$. Since this tuple corresponds to a non-relevant chain, there exists a direct query $(2, X_2)$ by A_2 (such that $G_2[X_2] = \perp$) which lead to Sim query Func on $(+, X_0||X_1)$. Since $G_2[X_2] = \perp$, $G_2[X_2]$ is assigned to $f[2, X_2]$ on A_2 and then Sim would have considered all tuples in $G_1 \times \{X_2\}$ and issued appropriate queries to Func. Since $(X_1, X_2, 1)$ is not a relevant chain, Func replies to $(+, X_0||X_1)$ using ρ and hence $G_5[X_5] = \perp$. This chain is then delayed for completion. Moreover, Chains$[3, X_3] = (5, X_5)$ and $((X_1, X_2, 1), (X_5, X_6))$ is added to the set Chains$[5, X_5]$ and $G_3[X_3] = \perp$.

Since Sim runs AllComplete in Phase 2, we know that at the end of the execution, $G_3[X_3] \neq \perp$ and $G_5[X_5] \neq \perp$. However, X_3 (resp., X_5) might get added to the table G_3 (resp. G_5) in Phase 1 as well. We analyse each of these possibilities below and show that there is a call to Complete on $(X_1, X_2, 1)$.

- Case A: Either $G_3[X_3] \neq \perp$ or $G_5[X_5] \neq \perp$ at the end of Phase 1. If at the end of Phase 1 either $X_3 \in G_3$ or $X_5 \in G_5$ then there must occur a tuple $(Y_3, 3, X_3)$ or $(Y_5, 5, X_5)$ in T. Consider the first occurrences of the two tuple and moreover consider the earlier among them.

  - Case A.1: $(Y_5, 5, X_5)$ is earlier. This tuple corresponds to a call to $\mathsf{F}^{\text{inner}}$ on $(5, X_5)$. This could either be due to a direct query $(5, X_5)$ by A_2 or due to a direct query $(3, X'_3)$ by A_2 such that Chains$[3, X'_3] \leftarrow (5, X_5)$. In any case, Sim will be executing lines 27-30 and there is a call to Complete on $((X_1, X_2, 1), (X_5, X_6))$.

– Case A.2: $(Y_3, 3, X_3)$ is earlier. This tuple may correspond to an $\mathsf{F}^{\text{inner}}$ call to $(3, X_3)$ which is a result of a query $(3, X_3)$ by A_2. In this case, Sim executes lines 31-32 and ends up calling itself on $(5, X_5)$. We know by Lemma 22 that there are no overwrites in Chains$[3, X_3]$ and hence it calls to precisely this $(5, X_5)$. Therefore by the above argument there is a call to Complete. This tuple cannot correspond to a call to ForceVal$(X_3, Y_3, 3)$ (an earlier argument made in Case B.2.1 of Lemma 24 suffices, we will trigger the event BadGutShot).

- Case B: $G_3[X_3] = \perp$ and $G_5[X_5] = \perp$ at the end of Phase 1. If at the end of Phase 1, $G_3[X_3] = \perp$ and $G_5[X_5] = \perp$. Then in Phase 2, Sim runs AllComplete and hence there occurs a call to Complete on $(X_1, X_2, 1)$ as $((X_1, X_2, 1), (X_5, X_6)) \in$ Chains$[5, X_5]$.

Therefore in all cases, we have proved that there is a call to Complete. It is easy to see that after Complete returns, Check on $(+, X_0, X_1, X_5, X_6)$ is going to return true. From Lemma 23 we know that there are no overwrites in G_k. Hence, Check will return true even at the end of the execution. ■

The randomness-mapping argument

After proving the properties of good executions, that is, executions of $\mathsf{G}_2^{f,\rho}$ with a good pair (f, ρ), we are ready to define the map τ that maps (f, ρ) to h such that the executions $\mathsf{G}_2^{f,\rho}$ and G_3^{h} are identical. This allows us to show the indistinguishability of G_2 and G_3 (proved in Lemma 28).

Definition 9 (The map τ) *For $(f, \rho) \in \mathcal{F} \times \mathcal{R}$, $h \leftarrow \tau(f, \rho)$ is defined as follows:*

1. *Set $h = f$ and run $\mathsf{G}_2^{f,\rho}$ with $A = (A_1, A_2)$.*

2. *If for some* (k, X) *there is a* $\mathsf{ForceVal}(X, Y, k)$ *call then set* $h[k, X] \leftarrow Y$ *for the first such call.*

In the following lemma we finally prove that for a good pair (f, ρ) the transcript of $\mathsf{G}_2^{f,\rho}$ is identical to the transcript of G_3^h where $h = \tau(f, \rho)$.

Lemma 26 $T(\mathsf{G}_2^{f,\rho}) = T(\mathsf{G}_3^{h=\tau(f,\rho)})$. *Hence* $\mathsf{Pr}\left[\mathsf{G}_2^{f,\rho}\right] = \mathsf{Pr}\left[\mathsf{G}_3^h\right]$.

Proof: Let for ease of notation $T_2 = T(\mathsf{G}_2^{f,\rho})$ and $T_3 = T(\mathsf{G}_3^h)$ and let T_{2_j} (T_{3_j}) denote the jth tuple in the sequence T_2 (T_3).

The first tuple in both transcripts is going to correspond to the query by A_1. We analyse when $T_{21} = (+, X_0, X_1, X_5, X_6)$ and the other case is symmetric. A_1 queries $\mathsf{Func}(+, X_0||X_1)$ which returns (X_5, X_6) by accesing $\rho[+, X_0, X_1]$. Since (f, ρ) is good and $\mathsf{isTrue}(T_2)$ is true (Lemma 25), we know that (X_5, X_6, X_0, X_1) obey the following relation:

$$X_2 = G_1[X_1] \oplus X_0;\ X_3 = G_2[X_2] \oplus X_1;\ X_4 = G_3[X_3] \oplus X_2;$$
$$X_5 = G_4[X_4] \oplus X_3;\ X_6 = G_5[X_5] \oplus X_4;$$

Since A is deterministic, it must be the case that $T_{31} = (+, X_0, X_1, X_5', X_6')$. Func performs the following computations on (X_0, X_1) to return (X_5', X_6').

$$X_2' = H_1[X_1] \oplus X_0;\ X_3' = H_2[X_2'] \oplus X_1;\ X_4 = H_3'[X_3'] \oplus X_2';$$
$$X_5' = H_4[X_4'] \oplus X_3';\ X_6' = H_5[X_5'] \oplus X_4';$$

Since (f, ρ) is good and by Lemma 23 there are no overwrites in the tables G_k (in G_2), the image h of the map is such that $G_k[X] = h[k, X]$. Since there are no overwrites in H_k, it is also the case that $H_k[X] = h[k, X]$. Therefore, $(X_5', X_6') = (X_5, X_6)$ and $T_{21} = T_{31}$.

Let us assume that $T_{2_i} = T_{3_i}$ for $i \in [j-1]$. We next argue about the equality of the jth tuple. Since the transcripts are identical till $(j-1)$, it must be the case that the jth

tuple in both transcripts have the same format i.e. both of them correspond to either $\mathsf{F}^{\text{inner}}$,Func or ForceVal.

- Case A: $T_{2j} = (Y, k, X)$ and it corresponds to $\mathsf{ForceVal}(X, Y, k)$. As transcripts are equivalent till $(j-1)$, and the view of Sim is identical till now it must be the case that if T_{2j} corresponds to a ForceVal call then $T_{2j} = T_{3j}$.

- Case B: $T_{2j} = (Y, k, X)$ and it corresponds to an $\mathsf{F}^{\text{inner}}$ call on (k, X). If this is not the first occurrence of the tuple $(\cdot, k, X)$ then the state of tables G_k is identical in both games (as transcripts are identical until this point). Therefore, $T_{2j} = T_{3j}$. Hence, let us consider that this is the first occurrence of the tuple (Y, k, X) in T_2. Since the transcripts are identical until this point so is the view of Sim, then $T_{3j} = (Y', k, X)$. As this is the first occurrence of (Y, k, X) in T_2 we have $Y = f[k, X] = G_k[X]$. Similarly in T_3 we have $Y' = h[k, X]$. By definition of the map τ, $Y = Y'$.

- Case C: $T_{2j} = (\sigma, X_k, X_{k+1}, X_{k'}, X_{k'+1})$. If Func replies to (σ, X_k, X_{k+1}) without accessing ρ then it must be the case that $T_{2j} = T_{3j}$. If Func replies using ρ, then by the same argument of the equality of T_{21} and T_{31}, we have that $T_{2j} = T_{3j}$.

Therefore, $T_2 = T_3$. Consider the last tuple in T_2 after which A_2 outputs the bit b'. Let it be the mth tuple. Since $T_2[1 \dots m] = T_3[1 \dots m]$, A_2 in G_3^h necessarily outputs the same guess b'. Hence the output distributions of $\mathsf{G}_2^{f,\rho}$ and G_3^h are identical. ■

Next, we consider a key combinatorial property of the map τ restricted to the set $\mathsf{GOOD} \subseteq \mathcal{F} \times \mathcal{R}$.

$$\mathsf{GOOD} = \{(f, \rho) \in \mathcal{F} \times \mathcal{R} : (f, \rho) \text{ is a good pair}\} .$$

Lemma 27 *$\tau : \mathsf{GOOD} \to \mathcal{F}$ is $|\mathcal{R}|$-regular.*

Proof: Let $h \in \mathcal{F}$ have at least one good pre-image (f_h, ρ_h). In the game $\mathsf{G}_2^{f_h,\rho_h}$, both A_1 and Sim issue queries to Func. Some of these calls to Func may access the randomness ρ_h. Let α be the number of such calls to Func that access ρ_h. Therefore, from the entire table ρ_h only α indices are accessed during the entire execution. Let these α indices be $\{I_1, \ldots, I_\alpha\}$.

For every chain completed by Sim there exists a unique query to Func which accesses the randomness ρ_h. If the completed chain is relevant then Func query (that accesses ρ_h) is by A_1. For non-relevant chains that are completed the Func query (that accesses ρ_h) is by Sim. Moreover, because the Sim at the end executes the $\mathsf{AllComplete}$ function and the predicate isTrue on the transcript T of the execution is true (Lemma 25), Sim completes chains corresponding to all its Func queries. This implies that there are exactly α chains that are completed during the execution of $\mathsf{G}_2^{f_h,\rho_h}$. Therefore, there were exactly 2α calls to $\mathsf{ForceVal}$. Since there are no overwrites in an execution (Lemma 23) with a good pair, there are 2α indices in f that are not carried over to h (in the definition of τ). These correspond to all the $\mathsf{ForceVal}(X, \cdot, k)$ calls. Let these 2α indices be $\{J_1, \ldots, J_{2\alpha}\}$.

Consider a $\mathcal{FR}_h \subseteq \mathcal{F} \times \mathcal{R}$ such that every element $(f, \rho) \in \mathcal{FR}_h$ are as follows:

$$\begin{aligned} f[x] &= f_h[x] && \forall x \in [5] \times \{0,1\}^n \setminus \{J_1, \ldots, J_{2\alpha}\} \\ \rho[X] &= \rho_h[X] && \forall X \in \{I_1, \ldots, I_\alpha\}. \end{aligned} \tag{3.19}$$

We claim that $|\mathcal{FR}_h| = |\mathcal{R}|$. To show that τ is $|\mathcal{R}|$-regular, we need to show that $\mathcal{FR}_h \subseteq \mathsf{GOOD}$ and that $\tau(f, \rho) \neq h$ where $(f, \rho) \notin \mathcal{FR}$. Since $(f, \rho) \in \mathcal{FR}_h$ agree with (f_h, ρ_h) on all indices that are accessed (which define the goodness of the pair), therefore if (f_h, ρ_h) is a good pair then so is (f, ρ). Consider $(f, \rho) \notin \mathcal{FR}$, then either f disagrees with f_h on an index other than J_i or ρ disagrees with ρ_h on an index I_j. It is easy to see that in either case there exists an index L such that $h[L] \neq \tau(f, \rho)[L]$. ■

We are now ready to consider $\Delta(\mathsf{G}_2, \mathsf{G}_3)$.

$$\begin{aligned}
\mathsf{Pr}\left[\mathsf{G}_2\right] &= \mathsf{Pr}\left[(f,\rho) \xleftarrow{\$} \mathcal{F}\times\mathcal{R} : \mathsf{G}_2^{f,\rho}\right] \\
&= \sum_{(f,\rho)\in\mathsf{GOOD}} \mathsf{Pr}\left[(f,\rho)\right]\cdot \mathsf{Pr}\left[\mathsf{G}_2^{f,\rho}\right] + \sum_{(f,\rho)\in\overline{\mathsf{GOOD}}} \mathsf{Pr}\left[(f,\rho)\right]\cdot \mathsf{Pr}\left[\mathsf{G}_2^{f,\rho}\right] \\
&\leq \sum_{(f,\rho)\in\mathsf{GOOD}} \Pr\left[(f,\rho)\right]\cdot \Pr\left[\mathsf{G}_2^{f,\rho}\right] + \Pr\left[(f,\rho) \xleftarrow{\$} \mathcal{F}\times\mathcal{R} : (f,\rho)\in\overline{\mathsf{GOOD}}\right] \qquad (3.20) \\
&= \sum_{(f,\rho)\in\mathsf{GOOD}} \Pr\left[(f,\rho)\right]\cdot \Pr\left[\mathsf{G}_3^{h=\tau(f,\rho)}\right] \\
&\quad + \Pr\left[(f,\rho) \xleftarrow{\$} \mathcal{F}\times\mathcal{R} : (f,\rho)\in\overline{\mathsf{GOOD}}\right] \ ,
\end{aligned}$$

where the last equality is due to Lemma 26.

$$\mathsf{Pr}\left[\mathsf{G}_2\right] \leq \frac{1}{|\mathcal{F}|\times|\mathcal{R}|} \sum_{(f,\rho)\in\mathsf{GOOD}} \Pr\left[\mathsf{G}_3^{h=\tau(f,\rho)}\right] + \Pr\left[(f,\rho) \xleftarrow{\$} \mathcal{F}\times\mathcal{R} : (f,\rho)\in\overline{\mathsf{GOOD}}\right]$$

$$= \frac{1}{|\mathcal{F}|\times|\mathcal{R}|}\cdot|\mathcal{R}|\cdot \sum_{h\in\tau(\mathsf{GOOD})} \Pr\left[\mathsf{G}_3^{h}\right] + \Pr\left[(f,\rho) \xleftarrow{\$} \mathcal{F}\times\mathcal{R} : (f,\rho)\in\overline{\mathsf{GOOD}}\right] \ , \qquad (3.21)$$

where the last equality is due to Lemma 27.

$$\begin{aligned}
\mathsf{Pr}\left[\mathsf{G}_2\right] &\leq \frac{1}{|\mathcal{F}|}\cdot \sum_{h\in\tau(\mathsf{GOOD})} \Pr\left[\mathsf{G}_3^{h}\right] + \Pr\left[(f,\rho) \xleftarrow{\$} \mathcal{F}\times\mathcal{R} : (f,\rho)\in\overline{\mathsf{GOOD}}\right] \\
&= \sum_{h\in\tau(\mathsf{GOOD})} \Pr[h]\cdot \Pr\left[\mathsf{G}_3^{h}\right] + \Pr\left[(f,\rho) \xleftarrow{\$} \mathcal{F}\times\mathcal{R} : (f,\rho)\in\overline{\mathsf{GOOD}}\right] \qquad (3.22) \\
&\leq \Pr[\mathsf{G}_3] + \Pr\left[(f,\rho) \xleftarrow{\$} \mathcal{F}\times\mathcal{R} : (f,\rho)\in\overline{\mathsf{GOOD}}\right]
\end{aligned}$$

Lemma 28 $\Delta(\mathsf{G}_2, \mathsf{G}_3) \leq 351q^6/2^n$.

Proof: From Equation 3.22 and Lemma 21 the lemma holds. ■

3.4.5 Indistinguishability of G_3 and G_4

Lemma 29 *In* G_3, Prim *always replies with* $h[k, X]$ *for any query* (k, X).

Proof: Sim in G_3^h responds to (k, X) with $G_k[X]$. During the execution, Sim sets $G_k[X] \leftarrow H_k[X]$ for all $k \in \{1, 5\}$. Therefore for such assignments the lemma holds as $H_k[X] = h[k, X]$ throughout the execution of G_3. For $k \in \{2, 3, 4\}$ there are two cases: (a) $G_k[X]$ is assigned to $H_k[X]$ or (b) $G_k[X]$ is forced to a value Y in a call to $\mathsf{ForceVal}(X, Y, k)$. For the case (a) the lemma holds by the same argument as $k \in \{1, 5\}$. For case (b) some analysis is in order. A call to ForceVal occurs only when Complete gets called on some partial chain C. This is true for both type of calls to Complete i.e. direct or immediate. We will consider the case when $C = (X_1, X_2, 1)$ and hence argue about $k \in \{3, 4\}$. The other case $C = (X_4, X_5, 4)$ can be symmetrically handled.

For $\mathsf{Complete}(X_1, X_2, 1)$ there was a query to Func i.e. $\mathsf{Func}(+, X_0||X_1)$ where $X_0 = G_1[X_1] \oplus X_2$. To answer this query Func evalutes the Feistel construction forward by accessing H (hence h). Then it must be the case that both calls to $\mathsf{ForceVal}(X, Y, k)$ it holds that $Y = h[k, X]$. Therefore $G_k[X] = H_k[X] = h[k, X]$ for all (k, X) such that $G_k[X] \neq \perp$. Hence the lemma holds.

■

Lemma 30 $\Delta(\mathsf{G}_3, \mathsf{G}_4) = 0$.

Proof: Due to Lemma 29 the lemma holds. ■

From Lemmas 14,28,30 we have that $\Delta(\mathsf{G}_1, \mathsf{G}_4) \leq 360q^6/2^n$ where Sim makes at most $2q^2$ queries.

MAIN $\mathsf{G}_1^A(n)$:

$P^{+}, P^{-} \leftarrow \phi$

$\mathsf{st} \xleftarrow{\$} A_1^{\mathsf{Func}}(1^n)$

$b' \xleftarrow{\$} A_2^{\mathsf{Prim}}(1^n, \mathsf{st})$

return b'

ORACLE $\mathsf{Prim}(k, X)$:

$Y \xleftarrow{\$} \mathsf{Sim}(k, X)$

return Y

ORACLE $\mathsf{Func}(+, X_0||X_1)$:

if $P[+, X_0, X_1] = \perp$ **then**

  $(X_5, X_6) \xleftarrow{\$} \{0,1\}^n \times \{0,1\}^n \setminus P^{-}$

  $P^{-} \xleftarrow{\cup} \{(X_5, X_6)\}$, $P^{+} \xleftarrow{\cup} \{(X_0, X_1)\}$

  $P[+, X_0, X_1] \leftarrow (X_5, X_6)$

  $P[-, X_5, X_6] \leftarrow (X_0, X_1)$

return $P[+, X_0, X_1]$

ORACLE $\mathsf{Func}(-, X_5||X_6)$

if $P[-, X_5, X_6] = \perp$ **then**

  $(X_0, X_1) \xleftarrow{\$} \{0,1\}^n \times \{0,1\}^n \setminus P^{+}$

  $P^{-} \xleftarrow{\cup} \{(X_5, X_6)\}$, $P^{+} \xleftarrow{\cup} \{(X_0, X_1)\}$

  $P[+, X_0, X_1] \leftarrow (X_5, X_6)$

  $P[-, X_5, X_6] \leftarrow (X_0, X_1)$

return $P[-, X_5, X_6]$

MAIN $\mathsf{G}_2^A(n)$:

$(f, \rho) \xleftarrow{\$} \mathcal{F} \times \mathcal{R}$

$\mathsf{st} \leftarrow A_1^{\mathsf{Func}}(1^n)$

$b' \leftarrow A_2^{\mathsf{Prim}}(1^n, \mathsf{st})$

$\mathsf{AllComplete}()$

return b'

ORACLE $\mathsf{Prim}(k, X)$:

$Y \leftarrow \mathsf{Sim}(f)(k, X)$

return Y

ORACLE $\mathsf{Func}(+, X_0||X_1)$

if $P[+, X_0, X_1] = \perp$ **then**

  $(X_5, X_6) \leftarrow \rho[+, X_0, X_1]$

  $P[+, X_0, X_1] \leftarrow (X_5, X_6)$

  $P[-, X_5, X_6] \leftarrow (X_0, X_1)$

return $P[+, X_0, X_1]$

ORACLE $\mathsf{Func}(-, X_5||X_6)$

if $P[-, X_5, X_6] = \perp$ **then**

  $(X_0, X_1) \leftarrow \rho[-, X_5, X_6]$

  $P[-, X_5, X_6] \leftarrow (X_0, X_1)$

  $P[+, X_0, X_1] \leftarrow (X_5, X_6)$

return $P[-, X_5, X_6]$

Figure 3.11: Game G_1 is described on the left along with the Prim and Func oracles. Game G_2 is defined on the right along with the Prim and Func oracles where the randomness f, ρ is sampled uniformly at the beginning of the game.

PROCEDURE AllPrimitive():

foreach $i \in [q_c]$ **do**
  $V \leftarrow X_1^i, U \leftarrow X_0^i$
  foreach $i \in [5]$ **do**
    $Y \leftarrow \mathsf{Prim}(i, V) \oplus U$
    $U \leftarrow V$, $V \leftarrow Y$

PROCEDURE AllComplete():

foreach $(k, X) \in \text{Chains.keys}()$ **do**
  if $k \in \{1, 5\}$ **then**
    $\mathsf{F}^{\text{inner}}(k, X)$
    foreach $(C, (U, V)) \in \text{Chains}[k, X]$ **do**
      if $C \notin \text{CompletedChains}$ **then**
        $\mathsf{Complete}(C, (U, V))$

Figure 3.12: The procedure AllPrimitive (run by A_2) making primitive queries corresponding to all construction queries by A_1. Here, (X_0^i, X_1^i) (passed to A_2 through st) is either the ith query by A_1 or its response by Func. The procedure AllComplete (run by Sim) at the end of G_2 issues calls to Complete for all incomplete chains whose completion was delayed. T.keys() returns the set of all keys in the hashtable T.

MAIN $\mathsf{G}_3^A(n)$:

$h \xleftarrow{\$} \mathcal{F}$

$\mathsf{st} \leftarrow A_1^{\mathsf{Func}}(1^n)$

$b' \leftarrow A_2^{\mathsf{Prim}}(1^n, \mathsf{st})$

$\mathsf{AllComplete}()$

return b'

ORACLE $\mathsf{Func}(+, X_0||X_1)$:

if $P[+, X_0, X_1] = \perp$ **then**

  $U \leftarrow X_0$, $V \leftarrow X_1$

  foreach $i \in [1, 2, 3, 4, 5]$ **do**

    if $H_i[V] = \perp$ **then**

      $H_i[V] \leftarrow h[i, X]$

    $X \leftarrow H_i[V] \oplus U$

    $U \leftarrow V$, $V \leftarrow X$

  $P[+, X_0, X_1] \leftarrow (U, V)$

  $P[-, U, V] \leftarrow (X_0, X_1)$

return $P[+, X_0, X_1]$

ORACLE $\mathsf{Prim}(k, X)$:

$Y \leftarrow \mathsf{Sim}(k, X)$

return Y

PROCEDURE $\mathsf{F}^{\mathrm{inner}}(k, X)$:

if $G_k[X] = \perp$ **then**

  if $H_k[X] = \perp$ **then**

    $H_k[X] \leftarrow h[k, X]$

  $G_k[X] \leftarrow H_k[X]$

return $G_k[X]$

ORACLE $\mathsf{Func}(-, X_5||X_6)$:

if $P[-, X_5, X_6] = \perp$ **then**

  $V \leftarrow X_5$, $U \leftarrow X_6$

  foreach $i \in [5, 4, 3, 2, 1]$ **do**

    if $H_i[V] = \perp$ **then**

      $H_i[V] \leftarrow h[i, X]$

    $A \leftarrow H_i[V] \oplus U$

    $U \leftarrow V$, $V \leftarrow A$

  $P[-, X_5, X_6] \leftarrow (U, V)$

  $P[+, U, V] \leftarrow (X_5, X_6)$

return $P[-, X_5, X_6]$

Figure 3.13: Game G_3 with its oracles Prim and Func. The Prim oracle behaves exactly like Sim as described in Figure 3.9 except the $\mathsf{F}^{\mathrm{inner}}$ procedure which is shown here.

MAIN $\mathsf{G}_4^A(n)$:

$\mathsf{st} \xleftarrow{\$} A_1^{\mathsf{Func}}(1^n)$

$b' \xleftarrow{\$} A_2^{\mathsf{Prim}}(1^n, \mathsf{st})$

return b'

ORACLE $\mathsf{Prim}(k, X)$:

if $G_k[X] = \perp$ **then**

$G_k[X] \xleftarrow{\$} \{0,1\}^n$

return $G_k[X]$

ORACLE $\mathsf{Func}(\mathsf{+}, X_0||X_1)$:

if $P[\mathsf{+}, X_0, X_1] = \perp$ **then**

$U \leftarrow X_0$, $V \leftarrow X_1$

foreach $i \in [1,2,3,4,5]$ **do**

$X \xleftarrow{\$} \mathsf{Prim}(i, V) \oplus U$

$U \leftarrow V$, $V \leftarrow X$

$P[\mathsf{+}, X_0, X_1] \leftarrow (U, V)$

$P[\mathsf{-}, U, V] \leftarrow (X_0, X_1)$

return $P[\mathsf{+}, X_0, X_1]$

ORACLE $\mathsf{Func}(\mathsf{-}, X_5||X_6)$:

if $P[\mathsf{-}, X_5, X_6] = \perp$ **then**

$V \leftarrow X_5$, $U \leftarrow X_6$

foreach $i \in [5,4,3,2,1]$ **do**

$X \xleftarrow{\$} \mathsf{Prim}(i, V) \oplus U$

$U \leftarrow V$, $V \leftarrow X$

$P[\mathsf{-}, X_5, X_6] \leftarrow (U, V)$

$P[\mathsf{+}, U, V] \leftarrow (X_5, X_6)$

return $P[\mathsf{-}, X_5, X_6]$

Figure 3.14: Game G_4 with its oracle Prim and Func.

PROCEDURE isTrue(T):

foreach $(\sigma, X_k, X_{k+1}, X_{k'}, X_{k'+1}) \in T$ **do**

  if $\neg$Check$(\sigma, X_k, X_{k+1}, X_{k'}, X_{k'+1})$ **then return** false

return true

PROCEDURE Check$(\sigma, X_k, X_{k+1}, X_{k'}, X_{k'+1})$

if $\sigma =$ + **then**

  $U \leftarrow X_k, V \leftarrow X_{k+1}$

  for $i \in [1, 2, 3, 4, 5]$ **do**

    if $(Y, i, V) \notin T$ **then return** false

    $A \leftarrow Y \oplus U$, $U \leftarrow V$, $V \leftarrow A$

  if $(U, V) \neq (X_{k'}, X_{k'+1})$ **then return** false

else

  $V \leftarrow X_k, U \leftarrow X_{k+1}$

  for $i \in [5, 4, 3, 2, 1]$ **do**

    if $(Y, i, V) \notin T$ **then return** false

    $A \leftarrow Y \oplus U$, $U \leftarrow V$, $V \leftarrow A$

  if $(U, V) \neq (X_{k'}, X_{k'+1})$ **then return** false

return true

Figure 3.15: The predicate isTrue defined on the transcript T.

Chapter 4

Public-seed PRFs to Public-seed PRPs via Naor-Reingold

The main result of this chapter show that psPRP security as introduced in Chapter 3 can be achieved in just two-calls, thereby improving the five-call result from Chapter 3. To show this result we first provide an overview of our techniques in Section 4.1. Then introduce some preliminary notation in Section 4.2 then formally define the NR construction and argue its indistinguishability in Section 4.3. Then, we prove that it transforms a UCE into a psPRP, for both unpredictable (Section 4.4) and reset-secure sources (Section 4.5).

4.1 Overview of Techniques

Let us briefly recall the setting: For some PPT source S, which queries a permutation oracle on $2n$-bit strings to produce a leakage L, we need to show that any PPT distinguisher D which learns L *and* $\vec{s} = (s, s^{\mathsf{in}}, s^{\mathsf{out}})$ cannot tell apart whether S was accessing NR using a UCE H (with seed $\vec{s}$) or a truly random permutation. We assume S is either (statistically) unpredictable or reset-secure.

The source $\overline{S}$. The natural approach we follow is to build another source, $\overline{S}$ from S, for which H should be a secure UCE. This source thus accesses an oracle $\mathcal{O}$ implementing a function from $n+1$ to n bits. It first samples seeds $s^{\mathsf{in}}, s^{\mathsf{out}}$ for P, and then simulates an execution of S. The oracle calls by the latter are processed by evaluating the NR construction using $s^{\mathsf{in}}, s^{\mathsf{out}}$, and the oracle $\mathcal{O}(\cdot)$ in lieu of $\mathsf{H}(s,\cdot)$. Finally, when S produces its output L, $\overline{S}$ outputs $(L, s^{\mathsf{in}}, s^{\mathsf{out}})$. We will show the following two facts:

- FACT 1. If S is unpredictable (w.r.t. the psPRP notion), then $\overline{S}$ is unpredictable (w.r.t. the UCE notion).

- FACT 2. If S is reset-secure (w.r.t. the psPRP notion), then $\overline{S}$ is reset-secure (w.r.t. the UCE notion).

The Theorem follows from Facts 1 and 2, respectively, by a fairly straightforward application of the (classical) indistinguishability of the NR construction with *random* round functions.[1]

The unpredictable case. Our approach to establish Fact 1 is inspired by an elegant proof of secure domain extension for UCEs via Wegman-Carter MACs [44]. (The case of reset-secure sources will be more involved and use new techniques.)

Assume, towards a contradiction, that $\overline{S}$ is *not* unpredictable; then there exists a strategy (not necessarily efficient) that given L and s^{in} and s^{out}, guesses one of the inner oracle queries of $\overline{S}$ with non-negligible probability ε, when $\overline{S}$'s oracle is a random function from $n+1$ to n bits. Imagine now that given $(L, s^{\mathsf{in}}, s^{\mathsf{out}})$ from $\overline{S}$, we *resample* an execution of $\overline{S}$ (which in particular means re-sampling the oracle used by it) consistent with outputting $(L, s^{\mathsf{in}}, s^{\mathsf{out}})$, and look at the inner oracle queries in this virtual, re-sampled execution. Then, one can show that the real and the virtual executions are

[1] A minor caveat is that we need indistinguishability even when s^{in} and s^{out} are revealed at the end of the interaction. We will show this to be true.

likely to share an oracle query, with probability roughly at least ε^2, for our strategy to guess a query must be equally successful on the virtual execution.

We exploit this idea to build a predictor for the original source S, contradicting our hypothesis it is unpredictable. Note that S runs with a random permutation as its oracle, and produces leakage L. Imagine now we sample fresh seeds s^{in}, s^{out} for P, and for each permutation query by S defining an input-output pair (u, v), we define "fake" inputs x_0, x_1 from $x_0 \,\|\, x_1 = \mathsf{P}_{s^{\mathsf{in}}}(u)$ and $x_3 \,\|\, x_2 = \mathsf{P}_{s^{\mathsf{out}}}(v)$. Then, the indistinguishability of the NR construction from a random permutation, and the construction of $\overline{S}$, implies that if we resample a virtual execution of S consistent with leakage L, and compute the resulting fake inputs using s^{in} and s^{out}, then the real and the re-sampled execution will share a fake input with probability approx. ε^2. The properties imposed on P then imply that with probability roughly ε^2 the real and the re-sampled execution must share the input (or output) of a permutation query. This leads naturally to a predictor that just re-samples an execution consistent with the leakage, and picks them as its prediction.

Reset-security. The case of reset-security is somewhat harder. Here we start from the premise that $\overline{S}$ is not reset-secure: Hence, there exists an adversary $\overline{R}$ which receives $L, s^{\mathsf{in}}, s^{\mathsf{out}}$ from $\overline{S}$, and can distinguish (with non-negligible advantage ε) being given access to the same random $f : \{0,1\}^{n+1} \to \{0,1\}^n$ used by $\overline{S}$ from being given access to an independent f'. From this, we would like to build an adversary R which receives L from S, and can distinguish the setting where R and S are given access to the same random permutation ρ, from a setting where they access independent permutations ρ, ρ'.

The challenge here is that we want to simulate $\overline{R}$ correctly, by using a permutation oracle ρ/ρ' rather than f/f'. To better see why this is tricky, say S is the source that queries its permutation oracle on a random $2n$-bit string u, obtaining output v, and leaks

$L = (u, v)$. (This defines the corresponding $\overline{S}$.)[2] A clever $\overline{R}$ on input $(L = (u, v), s^{\mathsf{in}}, s^{\mathsf{out}})$ could do the following: It computes $x_0 \,\|\, x_1 \leftarrow \mathsf{P}(s^{\mathsf{in}}, u)$ and $x_3 \,\|\, x_2 \leftarrow \mathsf{P}(s^{\mathsf{out}}, v)$. Then, it queries x_1 to its oracle, and outputs 1 iff the output equals $x_0 \oplus x_2$. This should always be true when $\overline{R}$ accesses f, and almost never when it accesses f'.

The natural proof approach would now attempt to build R which runs $\overline{R}$ accessing a simulated oracle consistent with the NR construction on the permutation queries made by S. However, the problem is that generically R does not know which queries S has made. Previous work [38] handled this by requiring the construction to satisfy a weaker notion of indifferentiability, called CP-sequential indifferentiability, which essentially implies that there exists a simulator that can simulate f consistently by accessing ρ and ρ^{-1} only, and only needs to know the queries $\overline{R}$ makes to f. This would not work with NR and our $\overline{R}$, as the query x_1 is actually uniformly random, and the simulator would likely fail to set $x_0 \oplus x_2$ as the right output. This is why the approach of [38] ends up using the 5-round Feistel construction, as here $\overline{R}$'s attempt to evaluate the construction are readily detected, and answered consistently.

Our proof strategy via heavy-query sampling. Our main observation is that indifferentiability is an overkill in this setting. There is no reason $\overline{R}$ should act adversarially to the simulator. Even more so, we can use everything $\overline{R}$ knows, namely L, to our advantage! To do this, we use techniques borrowed from impossibility proofs in the random oracle model [95, 100]. Namely, R, on input L from S, first performs a number of permutation queries which are meant to include all of S's likely queries to its oracle, at least when R and S are run with the same permutation oracle ρ. To do this, R samples executions of S consistent with L, and the partial knowledge of the oracle ρ acquired so far. Each time such a partial execution is sampled, all queries contained in it are

[2] The reader should not be confused: $\overline{S}$ is clearly not reset-secure, but remember we are in the setting of a proof by contradiction, so the reduction must work here, too.

made to ρ, and the process is repeated a number of times polynomial in $1/\varepsilon$. Then, R samples $s^{\mathsf{in}}, s^{\mathsf{out}}$, and internally defines an oracle $f : \{0,1\}^{n+1} \to \{0,1\}^n$ that will be used to simulate an execution of $\overline{R}^f(L, s^{\mathsf{in}}, s^{\mathsf{out}})$. To do this, R goes through all input-output pairs (u, v) for queries to ρ it has done while simulating executions of S,[3] and defines

$$f(0 \,\|\, x_1) \leftarrow x_0 \oplus x_2 \;, f(1 \,\|\, x_2) \leftarrow x_1 \oplus x_3 \;,$$

where $x_0 \,\|\, x_1 \leftarrow \mathsf{P}_{s^{\mathsf{in}}}(u)$ and $x_3 \,\|\, x_2 \leftarrow \mathsf{P}_{s^{\mathsf{out}}}(v)$. Then, f is defined to be random on every other input (this can be simulated via lazy sampling). The final output of the simulated $\overline{R}$ is then R's final output.

The core of our proof will show that when S and R share access to ρ, then the probability that R's output is one is similar to that of $\overline{R}$ outputting one when it accesses the same oracle as $\overline{S}$. This will combine properties of the NR construction (allowing us to switch between f and ρ), and similar arguments as those used in [95] to prove that R ensures consistency on all queries that matter.[4]

4.2 Preliminaries

Notational preliminaries. Results in this section will be concrete, but natural asymptotic statements can be made by allowing all parameters to be functions of the security parameter.

A *function family* with input set X and output set Y is a pair of algorithms $\mathsf{F} = (\mathsf{F.Kg}, \mathsf{F.Eval})$, where the randomized *key (or seed) generation algorithm* $\mathsf{F.Kg}$ outputs a seed s, and the deterministic *evaluation algorithm* $\mathsf{F.Eval}$ takes as inputs a valid seed s and an input $x \in X$, and returns $\mathsf{F.Eval}(s, x) \in Y$. If $X = \{0,1\}^m$ and $Y = \{0,1\}^n$,

[3] The actual simulation will be slightly more involved, for the benefit of simplifying the analysis.

[4] We believe we could adapt our proof to use the better strategy of [100] to get slightly better concrete parameters, yet we found adapting it to our setting not immediate.

we say that F is a family of functions from m-bits to n-bits. We usually write $\mathsf{F}(s,\cdot) = \mathsf{F.Eval}(s,\cdot)$. A *permutation family* $\mathsf{P} = (\mathsf{P.Kg}, \mathsf{P.Eval})$ on n bits is the special case where $X = \{+,-\} \times \{0,1\}^n$ and $Y = \{0,1\}^n$, and for every s, there exists a permutation π_s such that $\mathsf{P.Eval}(s,(+,x)) = \pi_s(x)$ and $\mathsf{P.Eval}(s,(-,y)) = \pi_s^{-1}(y)$. We usually write $\mathsf{P}(s,\cdot) = \mathsf{P}(s,(+,\cdot))$ and $\mathsf{P}^{-1}(s,\cdot) = \mathsf{P}(s,(-,\cdot))$.

4.2.1 UCEs and psPRPs

In Chapter 3 we presented the general paradigm of public-seed pseudorandomness $\mathsf{pspr}[\mathbf{I}]$ for an ideal primitive $\mathbf{I}$ and saw UCE and psPRPs as specific instantiations of it corresponding to random functions and random permutations as the underlying ideal primitives. For better readibility we will describe UCE and psPRPs directly in this chapter.

Concretely, let H be function family from m-bits to n-bits and let (S,D) be a source-distinguisher pair. We associate with them the game $\mathsf{UCE}^{S,D}_{m,n,\mathsf{H}}$ depicted in Figure 4.1 obtained by plugging $\mathbf{I} = \mathbf{R}_{m,n}$ in the psPR game defined in Figure 3.1. Similarly, for a family E of permutations on n-bits, the psPRP-security game $\mathsf{psPRP}^{S,D}_{n,\mathsf{E}}$ is defined in Figure 4.1 obtained by plugging $\mathbf{I} = \mathbf{P}_n$ in the psPR game defined in Figure 3.1. The only difference from $\mathsf{UCE}^{S,D}$ game is in $\mathcal{O}$ which here allows for inverse queries, and the ideal object is a random permutation. The corresponding advantage metrics for an (S,D) are defined as

$$\mathsf{Adv}^{\mathsf{uce}}_{m,n,\mathsf{H}}(S,D) = 2\Pr\left[\mathsf{UCE}^{S,D}_{m,n,\mathsf{H}}\right] - 1 \;;\; \mathsf{Adv}^{\mathsf{psprp}}_{n,\mathsf{E}}(S,D) = 2\Pr\left[\mathsf{psPRP}^{S,D}_{n,\mathsf{E}}\right] - 1\,. \quad (4.1)$$

Recall we adopt the multi-key versions of UCE and psPRP security, as they are the most general, and they are not known to follow from the single-key case. Our treatment scales down to the single-key version by forcing the source to always choose $r = 1$.

As before, H is UCE-secure for a class of sources $\mathcal{S}$ if $\mathsf{Adv}^{\mathsf{uce}}_{m,n,\mathsf{H}}(S,D)$ is negligible for

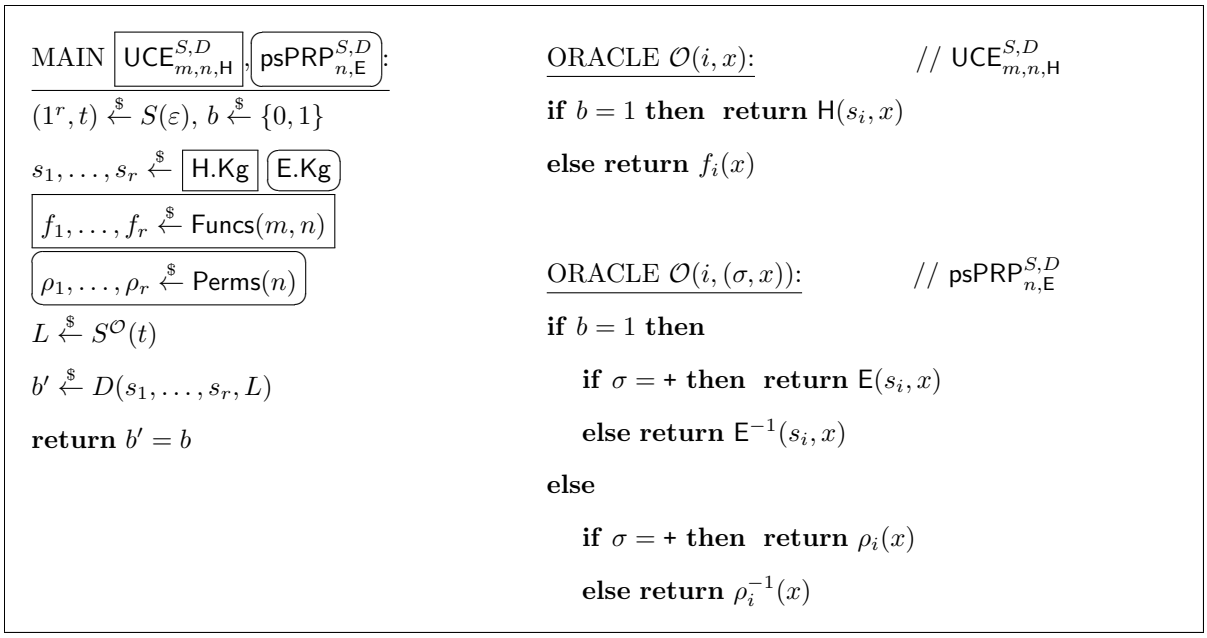


Figure 4.1: Games to define UCE and psPRP security. Here, S is the source and D is the distinguisher. Boxed statements are only executed in the corresponding game.

all PPT D and all sources $S \in \mathcal{S}$. Similarly, E is psPRP secure for $\mathcal{S}$ if $\mathsf{Adv}^{\mathsf{psprp}}_{n,\mathsf{E}}(S, D)$ is negligible for all PPT D and all sources $S \in \mathcal{S}$.

Restricting Sources. As noted in Chapter 3 we restrict our attention to unpredictable and reset-secure sources. The reset-security notion here is identical to the one in Chapter 3 where it was defined for a general ideal primitive $\mathbf{I}$. For unpredictability we next introduce an equivalent but semantically different variant called – *simple* unpredictability. Futhermore, we only consider the statistical versions of both these source classes – UCE and psPRP for the computationally secure sources (defined in Chapter 3) are impossible if IO exists [39] and furthermore are not relevant for applications. For concreteness, we define both simple-unpredictability and reset-security for random function and random permutation respectively.

Simple Unpredictability. Let S be a source and P be an adversary called the *predictor*. We associate with them games $\mathsf{f\text{-}Pred}^P_{m,n,S}$ and $\mathsf{p\text{-}Pred}^P_{n,S}$ of Fig. 4.2 which capture the fact that P cannot predict any of the queries of S (or their inverses), when the latter interacts with a random function from m bits to n bits, or respectively a random permutation on n-bit strings. The corresponding advantage metrics are

$$\mathsf{Adv}^{\mathsf{f\text{-}pred}}_{m,n,S}(P) = \Pr\left[\mathsf{f\text{-}Pred}^P_{m,n,S}\right] \ , \mathsf{Adv}^{\mathsf{p\text{-}pred}}_{n,S}(P) = \Pr\left[\mathsf{p\text{-}Pred}^P_{n,S}\right] \ . \tag{4.2}$$

We say S is *statistically simple-unpredictable or unpredictable* if $\mathsf{Adv}^{\mathsf{f\text{-}pred}}_{m,n,S}(P)$ (respectively, $\mathsf{Adv}^{\mathsf{p\text{-}pred}}_{n,S}(P)$) is negligible for all predictors P outputting a set Q' of polynomial size. The simple unpredictability notion differs from unpredictability as defined in Chapter 3 in that the predictor P is not permitted to query the underlying primitive. The notion was proved equivalent (asymptotically) for UCEs [37] to a version where we give P access to the primitive. A similar proof follows for psPRPs.

Reset-secure Sources. Let S be a source and R be an adversary called the reset-adversary. We associate to them the games $\mathsf{f\text{-}Reset}^R_{m,n,S}$ and $\mathsf{p\text{-}Reset}^R_{n,S}$ of Fig. 4.2 which formalize the reset-security of S against a random function and a random permutation, respectively. The idea here is that R should not be able to tell apart whether S is accessing the same set of oracles it accesses, or not. This is captured via the advantage metrics

$$\mathsf{Adv}^{\mathsf{f\text{-}reset}}_{m,n,S}(R) = 2\Pr\left[\mathsf{f\text{-}Reset}^R_{m,n,S}\right] - 1 \ , \quad \mathsf{Adv}^{\mathsf{p\text{-}reset}}_{n,S}(R) = 2\Pr\left[\mathsf{p\text{-}Reset}^R_{n,S}\right] - 1 \ .$$

We say S is *statistically reset-secure* if the corresponding advantage is negligible for all reset-adversaries R making a polynomial number of *queries* to their oracle, but which are otherwise computationally unrestricted. It is known that a (statistically) unpredictable source is (statistically) reset-secure, for both UCEs [37] and psPRPs [38]. The converse

MAIN $\boxed{\mathsf{f\text{-}Pred}^P_{m,n,S}}$, $\boxed{\mathsf{p\text{-}Pred}^P_{n,S}}$:

$Q \leftarrow \emptyset$

$(1^r, t) \xleftarrow{\$} S(\varepsilon)$

$\boxed{f_1, \ldots, f_r \xleftarrow{\$} \mathsf{Funcs}(m,n)}$

$\boxed{\rho_1, \ldots, \rho_r \xleftarrow{\$} \mathsf{Perms}(n)}$

$L \xleftarrow{\$} S^{\mathcal{O}}(t)$

$Q' \xleftarrow{\$} P(1^r, L)$

return $(Q \cap Q' \neq \emptyset)$

ORACLE $\mathcal{O}(i,x)$: // $\mathsf{f\text{-}Pred}^P_{m,n,S}$

$Q \leftarrow Q \cup \{(i,x)\}$

return $f_i(x)$

ORACLE $\mathcal{O}(i,(\sigma,x))$: // $\mathsf{p\text{-}Pred}^P_{n,S}$

if $\sigma = +$ **then** $y \leftarrow \rho_i(x)$

else $y \leftarrow \rho_i^{-1}(x)$

$Q \leftarrow Q \cup \{(i,x),(i,y)\}$

return y

MAIN $\boxed{\mathsf{f\text{-}Reset}^R_{m,n,S}}$, $\boxed{\mathsf{p\text{-}Reset}^R_{n,S}}$:

$\mathsf{done} \leftarrow \mathsf{false}$; $(1^r, t) \xleftarrow{\$} S(\varepsilon)$

$\boxed{f_1^0, f_1^1, \ldots, f_r^0, f_r^1 \xleftarrow{\$} \mathsf{Funcs}(m,n)}$

$\boxed{\rho_1^0, \rho_1^1, \ldots, \rho_r^0, \rho_r^1 \xleftarrow{\$} \mathsf{Perms}(n)}$

$L \xleftarrow{\$} S^{\mathcal{O}}(t)$; $\mathsf{done} \leftarrow \mathsf{true}$

$b \xleftarrow{\$} \{0,1\}$; $b' \xleftarrow{\$} R^{\mathcal{O}}(1^r, L)$

return $b' = b$

ORACLE $\mathcal{O}(i,x)$: // $\mathsf{f\text{-}Reset}^R_{m,n,S}$

if $\neg\mathsf{done}$ **then return** $f_i^0(x)$

else return $f_i^b(x)$

ORACLE $\mathcal{O}(i,(\sigma,x))$: // $\mathsf{p\text{-}Reset}^R_{n,S}$

if $\neg\mathsf{done}$ **then**

  if $\sigma = +$ **then return** $\rho_i^0(x)$

  else return $\rho_i^{0\,-1}(x)$

else

  if $\sigma = +$ **then return** $\rho_i^b(x)$

  else return $\rho_i^{b\,-1}(x)$

Figure 4.2: Games to define unpredictability (left) and reset-security (right) of sources. Here, S is the source, P is the predictor and R is the reset-adversary. Boxed statements are only executed in the corresponding game.

is not true – S may query a fixed known input, and let L be the empty string. S is reset-secure in the strongest sense, while being easily predictable.

4.3 The NR Construction and its Indistinguishability

This section reviews the NR construction [22] in the public-seed setting and proves a strong statement about its indistinguishability.

Notation. Let P be a permutation family on the $2n$-bit strings. We say that P is α-*right-universal* if $\mathsf{Pr}_{s \xleftarrow{\$} \mathsf{P.Kg}}[\mathsf{P}_1(s,u) = \mathsf{P}_1(s,u')] \leq \alpha$ for all distinct $u, u' \in \{0,1\}^{2n}$, where P_1 denote the second n-bit half of the output of P. Note that a pairwise-independent permutation is a good candidate of P, but a simpler approach is to employ one-round of Feistel with a pairwise independent hash function H as the round function, i.e., $\mathsf{P}(s,(u_0,u_1)) = (u_1, \mathsf{H}(s,u_1) \oplus u_0)$.

The Naor-Reingold (NR) Construction. Let H be a function family from $n+1$ bits to n bits. We define the permutation family $\mathsf{NR} = \mathsf{NR}[\mathsf{P},\mathsf{H}]$ on the $2n$-bit strings, where $\mathsf{NR.Kg}$ outputs $(s, s^{\mathsf{in}}, s^{\mathsf{out}})$ such that $s \xleftarrow{\$} \mathsf{H.Kg}$ and $s^{\mathsf{in}}, s^{\mathsf{out}} \xleftarrow{\$} \mathsf{P.Kg}$. Further, forward evaluation proceeds as follows (the inverse is obvious):

$$\underline{\text{Proc. } \mathsf{NR}((s, s^{\mathsf{in}}, s^{\mathsf{out}}), U)\text{:}}$$
$$x_0 \,\|\, x_1 \leftarrow \mathsf{P}(s^{\mathsf{in}}, U),\ x_2 \leftarrow \mathsf{H}(s, 0 \,\|\, x_1) \oplus x_0,$$
$$x_3 \leftarrow \mathsf{H}(s, 1 \,\|\, x_2) \oplus x_1,\ V \leftarrow \mathsf{P}^{-1}(s^{\mathsf{out}}, x_3 \,\|\, x_2),\ \textbf{return}\ V$$

Naor and Reingold [22] proved that the NR construction with random round functions is indistinguishable from a random permutation under chosen ciphertext attacks. We will need a stronger result, which we prove here, that this is true even when the seed of P is given to the adversary after it stops making queries, and when the adversary can make queries to multiple instances of the construction. It will be convenient to re-use the notation already in place for the psPRP framework, and we denote by $\mathsf{Adv}^{\mathsf{psprp}^+}_{2n,\mathsf{NR}[\mathsf{P},\mathsf{F}]}(S,D)$

the advantage obtained by (S, D) in the $\mathsf{psPRP}^{S,D}_{2n,\mathsf{NR}[\mathsf{P},\mathsf{F}]}$ game, with the modification that D is *not* given the seed for F, only the seeds used by the permutation P.

Proposition 5 (Indistinguishability of the NR construction) *Let* $\mathsf{F} = \mathsf{F}[n+1, n]$ *be the family of all functions from* $n+1$ *to* n *bits, equipped with the uniform distribution. Further, let* P *be* α*-right-universal. For all* S, D*, where* S *makes* q *queries,* $\mathsf{Adv}^{\mathsf{psprp}^+}_{2n,\mathsf{NR}[\mathsf{P},\mathsf{F}]}(S, D) \leq q^2 \cdot \left(2\alpha + \frac{1}{2^{2n}}\right)$.

Proof: The proof follows from a simple application of the H-coefficient technique [101, 102]. In particular, we describe the interaction of S, D in both cases $b = 0$ and $b = 1$ using a *transcript* τ which contains:

- The sequence of queries made by S to the permutation, in either direction, which consists of tuples (i, U, V), indicating that either $\mathcal{O}(i, (+, U))$ was queried, and returned V, or $\mathcal{O}(i, (-, V))$ was queried, returning U.
- The keys $s_1^{\mathsf{in}}, s_1^{\mathsf{out}}, \ldots, s_r^{\mathsf{in}}, s_r^{\mathsf{out}}$ given to D. Recall that in the "real-world", where S interacts with the NR construction, these are the actual keys. In the "ideal world", these keys are generated by running $\mathsf{P.Kg}$ $2r$ times, independently of the permutation.

We also say that the transcript τ is *bad* if there exist two entries (i, U, V) and (i, U', V') for $U' \neq U, V \neq V'$ such that $\mathsf{P}_1(s_i^{\mathsf{in}}, U) = \mathsf{P}_1(s_i^{\mathsf{in}}, U')$ or $\mathsf{P}_1(s_i^{\mathsf{out}}, V) = \mathsf{P}_1(s_i^{\mathsf{out}}, V')$.

Fix a good transcript τ. Let $\mathsf{p}_1(\tau)$ be the interpolation probability of the transcript τ when interacting with NR construction (using random round functions sampled from F), i.e., this is the probability that (S, D) obtains the answer in the transcript when asking the corresponding queries. Let $\mathsf{p}_0(\tau)$ be the probability in the ideal world, where the oracle queries are answered by independent permutations. Also, let q_i be the number

distinct entries of form (i, U, V), and assume wlog $i \in [r]$. Then,

$$\mathsf{p}_0(\tau) = \mathsf{p}(s_1^{\mathsf{in}}, s_1^{\mathsf{out}}, \ldots, s_r^{\mathsf{in}}, s_r^{\mathsf{out}}) \cdot \prod_{i=1}^{r} \prod_{j=0}^{q_i - 1} \frac{1}{2^{2n} - j} \ ,$$

where $\mathsf{p}(s_1^{\mathsf{in}}, s_1^{\mathsf{out}}, \ldots, s_r^{\mathsf{in}}, s_r^{\mathsf{out}})$ is the probability that these $2r$ keys are sampled by $\mathsf{P.Kg}$. Let us compare this with $\mathsf{p}_1(\tau)$. To this end, fix $i \in [r]$, then for every entry (i, U^j, V^j) in τ, $j \in \{1, \ldots, q_i\}$, define the corresponding values $x_0^j \,\|\, x_1^j = \mathsf{P}(s_i^{\mathsf{in}}, U^j)$ and $x_3^j \,\|\, x_2^j = \mathsf{P}(s_i^{\mathsf{out}}, V^j)$. If τ is good, then the values x_1^j are distinct, and so are x_2^j. This in particular means that we are interested in the probability, over the choice of a random f_i, that $f_i(0 \,\|\, x_1^j) = x_0^j \oplus x_2^j$ and $f_i(1 \,\|\, x_2^j) = x_3^j \oplus x_1^j$ for all j, which is exactly 2^{-2nq_i}. Overall, thus, taking a product over all $i = 1, \ldots, r$,

$$\mathsf{p}_1(\tau) = \mathsf{p}(s_1^{\mathsf{in}}, s_1^{\mathsf{out}}, \ldots, s_r^{\mathsf{in}}, s_r^{\mathsf{out}}) \cdot \prod_{i=1}^{r} 2^{-2nq_i} = \mathsf{p}(s_1^{\mathsf{in}}, s_1^{\mathsf{out}}, \ldots, s_r^{\mathsf{in}}, s_r^{\mathsf{out}}) \cdot 2^{-2nq} \ ,$$

which in particular means that

$$\frac{\mathsf{p}_1(\tau)}{\mathsf{p}_0(\tau)} \geq \prod_{i=1}^{r} \prod_{j=0}^{q_i - 1} \left(1 - \frac{j}{2^{2n}}\right) \geq 1 - \sum_{i=1}^{r} q_i^2 / 2^{2n} \geq 1 - \frac{q^2}{2^{2n}} = 1 - \varepsilon \ ,$$

where we have used that $(1-a)(1-b) \geq 1-a-b$ for all $a, b \geq 0$, and we let $\varepsilon = \frac{q^2}{2^{2n}}$. Also, using α-right universality, and the union bound, it is not hard to see that the probability that an ideal transcript is *bad* is at most

$$\delta := \sum_{i=1}^{r} 2q_i^2 \alpha \leq 2q^2 \alpha \ .$$

Then, using the H coefficient method, we can conclude that the bound is at most $\varepsilon + \delta$. This concludes the proof. ■

4.4 The Case of Unpredictable Sources

We first prove that the NR construction transforms a UCE function family for statistically unpredictable sources into a psPRP for statistically unpredictable sources. Our

proof uses a technique inspired from that of Bellare, Hoang, and Keelveedhi [44], given originally in the setting of UCE domain extension. Concretely, we prove the following.

Theorem 10 (NR security for unpredictable sources) *Let* P *be a* α*-right universal family of permutations on* $2n$*-bit strings. Let* H *be a family of functions from* $n+1$ *bits to* n *bits. Then, for all distinguishers* D *and sources* S *making overall* q *queries to their oracle, there exists* $\overline{D}$ *and* $\overline{S}$ *such that*

$$\mathsf{Adv}^{\mathsf{psprp}}_{2n,\mathsf{NR}[\mathsf{P},\mathsf{H}]}(S,D) \leq \mathsf{Adv}^{\mathsf{uce}}_{n+1,n,\mathsf{H}}(\overline{S},\overline{D}) + q^2\left(2\alpha + \frac{1}{2^{2n}}\right) . \tag{4.3}$$

Here, $\overline{D}$ *and* D *are roughly as efficient, and* $\overline{S}$ *and* S *are similarly as efficient. In particular,* $\overline{S}$ *makes* $2q$ *queries. Moreover, for every predictor* $\overline{P}$*, there exists a predictor* P *such that*

$$\mathsf{Adv}^{\mathsf{f\text{-}pred}}_{n+1,n,\overline{S}}(\overline{P}) \leq q^2 \cdot \left(2\alpha + \frac{1}{2^{2n}}\right) + p \cdot \sqrt{2q^2\alpha + \mathsf{Adv}^{\mathsf{p\text{-}pred}}_{2n,S}(P)} , \tag{4.4}$$

where p *is a bound on the size of the set output by* $\overline{P}$*.*

The asymptotic interpretation is that if $n = \omega(\log(\lambda))$ and α is negligible, if S is (statistically) unpredictable, then so is $\overline{S}$. Further, if H is a UCE for all unpredictable sources, then NR is a psPRP for all statistically unpredictable sources.

We stress that the predictor P built in the proof does not preserve the efficiency of $\overline{P}$, which is not a problem, as we only consider statistical notions. While we do not elaborate in the proof, it turns out that the running time of P is *exponential* in the length of S's leakage, thus the statement carries over to computational unpredictability if $L = O(\log \lambda)$.

Proof: We first consider three games, $\mathsf{G}_0, \mathsf{G}_1$, and G_2. Game G_0 is the game $\mathsf{psPRP}^{S,D}_{2n,\mathsf{NR}[\mathsf{P},\mathsf{F}]}$ in the case $b = 1$, and modified to return true if $b' = 1$. Game G_2 is the game $\mathsf{psPRP}^{S,D}_{2n,\mathsf{NR}[\mathsf{P},\mathsf{F}]}$ in the case $b = 0$, and modified to return true if $b' = 1$. The

Proc. $\overline{S}(\varepsilon)$:
$(1^r, t) \xleftarrow{\$} S(\varepsilon)$
return $(1^r, (1^r, t))$
Proc. $\overline{S}^{\mathcal{O}}(1^r, t)$:
$s_1^{\mathsf{in}}, s_1^{\mathsf{out}}, \ldots, s_r^{\mathsf{in}}, s_r^{\mathsf{out}} \xleftarrow{\$} \mathsf{P.Kg}$
$L \xleftarrow{\$} S^{\overline{\mathcal{O}}}(t)$
return $(L, s_1^{\mathsf{in}}, s_1^{\mathsf{out}}, \ldots, s_r^{\mathsf{in}}, s_r^{\mathsf{out}})$

Proc. $\overline{\mathcal{O}}(i, (\sigma, U))$:
if $\sigma =$ **+ then**
$x_0 \,\|\, x_1 \leftarrow \mathsf{P}(s_i^{\mathsf{in}}, U)$
$x_2 \leftarrow \mathcal{O}(i, 0 \,\|\, x_1) \oplus x_0$, $x_3 \leftarrow \mathcal{O}(i, 1 \,\|\, x_2) \oplus x_1$
$V \leftarrow \mathsf{P}^{-1}(s_i^{\mathsf{out}}, x_3 \,\|\, x_2)$
else
$x_3 \,\|\, x_2 \leftarrow \mathsf{P}(s_i^{\mathsf{out}}, U)$
$x_1 \leftarrow \mathcal{O}(i, 1 \,\|\, x_2) \oplus x_3$, $x_0 \leftarrow \mathcal{O}(i, 0 \,\|\, x_1) \oplus x_2$
$V \leftarrow \mathsf{P}^{-1}(s_i^{\mathsf{in}}, x_0 \,\|\, x_1)$
return V

Figure 4.3: The source $\overline{S}$ in the proof of Theorems 10 and 11.

intermediate game G_1 is obtained by modifying G_0 as follows: Initially, r random functions $f_1, \ldots, f_r \xleftarrow{\$} \mathsf{Funcs}(n+1, n)$ are sampled, and when evaluating the NR construction within $\mathcal{O}$ queries, the evaluation of $\mathsf{H}(s_i, b \,\|\, x)$ is replaced by an evaluation of the random function $f_i(b \,\|\, x)$. Then,

$$\mathsf{Adv}^{\mathsf{psprp}}_{2n,\mathsf{NR}[\mathsf{P},\mathsf{H}]}(S, D) = (\mathsf{Pr}\,[\mathsf{G}_0] - \mathsf{Pr}\,[\mathsf{G}_1]) + (\mathsf{Pr}\,[\mathsf{G}_1] - \mathsf{Pr}\,[\mathsf{G}_2]) \ .$$

We can directly get $\mathsf{Pr}\,[\mathsf{G}_1] - \mathsf{Pr}\,[\mathsf{G}_2] \leq q^2 \left(2\alpha + \frac{1}{2^{2n}}\right)$ as a corollary of Proposition 5, since neither of G_1 and G_2 uses the seeds generated by $\mathsf{H.Kg}$.

Going on, let us consider the new source $\overline{S}$ which simulates an execution of S, and uses access to an oracle $\mathcal{O}(i, X)$, implementing for each i a function from $n+1$ bits to n bits, to internally simulate the round functions NR construction used to answer S's queries. A formal description is in Figure 4.3. Also consider the distinguisher $\overline{D}$ such that

$$\overline{D}(L' = (L, \vec{s}^{\,\mathsf{in}}, \vec{s}^{\,\mathsf{out}}), \vec{s}) = D(L, (\vec{s}, \vec{s}^{\,\mathsf{in}}, \vec{s}^{\,\mathsf{out}})) \ ,$$

where $\vec{s} = (s_1, \ldots, s_r)$, $\vec{s}^{\,\mathsf{in}} = (s_1^{\mathsf{in}}, \ldots, s_r^{\mathsf{in}})$, and $\vec{s}^{\,\mathsf{out}} = (s_1^{\mathsf{out}}, \ldots, s_r^{\mathsf{out}})$ Therefore, G_0 and

G_1 behave exactly as $\mathsf{UCE}^{\overline{S},\overline{D}}_{n+1,n,\mathsf{H}}$ with challenge bits $b = 1$ and $b = 0$, respectively, with the only difference of outputting true whenever the distinguisher's output is $b' = 1$. Consequently, $\mathsf{Adv}^{\mathsf{uce}}_{n+1,n,\mathsf{H}}(\overline{S},\overline{D}) = \mathsf{Pr}\left[\mathsf{G}_0\right] - \mathsf{Pr}\left[\mathsf{G}_1\right]$.

The remainder of the proof relates the unpredictability of S and that of $\overline{S}$, establishing (4.4) in the theorem statement. We detail the unpredictability argument next.

Unpredictability Argument in the Proof of Theorem 10. Assume wlog in the following that $\overline{P}$ is deterministic. We consider three games G_{11}, G_{12}, and G_{13}.

Game G_{11} is a small syntactical alteration of Game $\mathsf{f\text{-}Pred}^{\overline{P}}_{n+1,n,\overline{S}}$. It keeps track in a set Q_{P} of the queries issued to $\overline{\mathcal{O}}$ by the simulated S within $\overline{S}$, i.e., an entry (i, U, V) is added to Q_{P} whenever $\overline{\mathcal{O}}(i, (\mathsf{+}, U))$ is queried, returning V, or (symmetrically) $\overline{\mathcal{O}}(i, (\mathsf{-}, V))$ is queried, returning U. Then, at the end of S's execution, we define

$$Q \leftarrow \left\{(i, 0 \,\|\, \mathsf{P}_1(s_i^{\mathsf{in}}, U)), (i, 1 \,\|\, \mathsf{P}_1(s_i^{\mathsf{out}}, V)) \;:\; (i, U, V) \in Q_{\mathsf{P}}\right\} \;,$$

where P_1 denotes the second n-bit half of the output of P. We also let $Q' \leftarrow \overline{P}(L, \vec{s})$ be the output of $\overline{P}$ given the leakage made by L and the seeds $\vec{s} = (s_1^{\mathsf{in}}, s_1^{\mathsf{out}}, \ldots, s_r^{\mathsf{in}}, s_r^{\mathsf{out}})$. Finally, G_{11} returns true iff $Q' \cap Q \neq \emptyset$. Clearly, as Q exactly captures the set of queries $\overline{S}$ makes to its oracle, the game returns true iff $\overline{P}$ predicts correctly, and we thus conclude that $\mathsf{Adv}^{\mathsf{f\text{-}pred}}_{n+1,n,\overline{S}}(\overline{P}) = \mathsf{Pr}\left[\mathsf{G}_{11}\right]$.

The definition of G_{11} however allows to easily introduce G_{12}, which answers S's $\overline{\mathcal{O}}$-queries via r randomly sampled permutations $\pi_1, \ldots, \pi_r$, i.e., $\overline{\mathcal{O}}(i, (\mathsf{+}, U)) = \pi_i(U)$ and $\overline{\mathcal{O}}(i, (\mathsf{-}, V)) = \pi_i^{-1}(V)$. This is the *only* difference. In particular, we still keep track of these queries in Q_{P}, compute Q, and set the winning condition as in G_{11}. It is not hard to see that Proposition 5 directly bounds $\mathsf{Pr}\left[\mathsf{G}_{11}\right] - \mathsf{Pr}\left[\mathsf{G}_{12}\right]$, and thus

$$\mathsf{Adv}^{\mathsf{f\text{-}pred}}_{n+1,n,\overline{S}}(\overline{P}) \leq q^2 \cdot \left(2\alpha + \frac{1}{2^{2n}}\right) + \mathsf{Pr}\left[\mathsf{G}_{12}\right] \;.$$

We are now going to construct a predictor P for S, whose advantage is related to $\mathsf{Pr}[\mathsf{G}_{12}]$. This P will generally be inefficient and (interestingly enough) will in fact not make any usage of the predictor $\overline{P}$. To this end, we first introduce a new predictor $\overline{P}'$ which on input $L, \vec{s}$ outputs a singleton set consisting of the most likely element from $\overline{P}(L, \vec{s})$ to be in Q. We let G_{13} be G_{12} with $\overline{P}'$ instead of $\overline{P}$. Clearly, $\mathsf{Pr}[\mathsf{G}_{12}] \leq p \cdot \mathsf{Pr}[\mathsf{G}_{13}]$.

In the following, in G_{13}, let us consider the conditional distribution $\mathcal{Q}_L$ of Q_{P} given some fixed L. Also, let $p(L)$ be the probability that a particular L is output by S in G_{12}, and let $q(\vec{s})$ be the probability that the seed-vector $\vec{s}$ is chosen, where recall that $\vec{s} = (s_1^{\mathsf{in}}, s_1^{\mathsf{out}}, \ldots, s_r^{\mathsf{in}}, s_r^{\mathsf{out}})$ contains the seeds for P. Finally, we let $Q(\vec{s}, Q_{\mathsf{P}})$ is the set Q computed from Q_{P} using the keys $\vec{s}$. Then, we can re-arrange the execution of G_{13} so that: (1) We first sample L according to the probability distribution $p(\cdot)$, (2) Then sample Q_{P} according to $\mathcal{Q}_L$, (3) Sample $\vec{s}$ according to $q(\cdot)$, and (4) Check whether $\overline{P}'(L, \vec{s}) \in Q(\vec{s}, Q_{\mathsf{P}})$. Then,

$$\mathsf{Pr}[\mathsf{G}_{13}] = \sum_{L,\vec{s}} p(L) \cdot q(\vec{s}) \cdot \mathop{\mathsf{Pr}}_{Q_{\mathsf{P}} \xleftarrow{\$} \mathcal{Q}_L} \left[\overline{P}'(L, \vec{s}) \in Q(\vec{s}, Q_{\mathsf{P}}) \right] .$$

Note now that for any fixed L and $\vec{s}$,

$$\begin{aligned} \mathop{\mathsf{Pr}}_{Q_{\mathsf{P}} \xleftarrow{\$} \mathcal{Q}_L} \left[\overline{P}'(L, \vec{s}) \in Q(\vec{s}, Q_{\mathsf{P}}) \right]^2 &= \mathop{\mathsf{Pr}}_{Q_{\mathsf{P}}, Q'_{\mathsf{P}} \xleftarrow{\$} \mathcal{Q}_L} \left[\overline{P}'(L, \vec{s}) \in Q(\vec{s}, Q_{\mathsf{P}}) \cap Q(\vec{s}, Q'_{\mathsf{P}}) \right] \\ &\leq \mathop{\mathsf{Pr}}_{Q_{\mathsf{P}}, Q'_{\mathsf{P}} \xleftarrow{\$} \mathcal{Q}_L} \left[Q(\vec{s}, Q_{\mathsf{P}}) \cap Q(\vec{s}, Q'_{\mathsf{P}}) \neq \emptyset \right] . \end{aligned}$$

Therefore,

$$\begin{aligned} \mathsf{Pr}[\mathsf{G}_{13}] &\leq \sum_{\vec{s},L} p(L) \cdot q(\vec{s}) \cdot \sqrt{\mathop{\mathsf{Pr}}_{Q_{\mathsf{P}}, Q'_{\mathsf{P}} \xleftarrow{\$} \mathcal{Q}_L} \left[Q(\vec{s}, Q_{\mathsf{P}}) \cap Q(\vec{s}, Q'_{\mathsf{P}}) \neq \emptyset \right]} \\ &\leq \sqrt{\sum_{\vec{s},L} p(L) \cdot q(\vec{s}) \cdot \mathop{\mathsf{Pr}}_{Q_{\mathsf{P}}, Q'_{\mathsf{P}} \xleftarrow{\$} \mathcal{Q}_L} \left[Q(\vec{s}, Q_{\mathsf{P}}) \cap Q(\vec{s}, Q'_{\mathsf{P}}) \neq \emptyset \right]} \end{aligned}$$

where the last inequality follows by Jensen's inequality.

Now, in the following, assume L is fixed such that $p(L) > 0$, and we sample $\vec{s}$ from $q(\cdot)$, as well as $Q_{\mathsf{P}}, Q'_{\mathsf{P}} \xleftarrow{\$} \mathcal{Q}_L$. We also define U_{P} and V_{P} such that $(i, U) \in U_{\mathsf{P}}$ and

$(i, V) \in V_{\mathsf{P}}$ if and only if $(i, U, V) \in Q_{\mathsf{P}}$. Define U'_{P} and V'_{P} analogously with respect to Q'_{P}. Then, let $\mathsf{Good} = \mathsf{Good}(Q_{\mathsf{P}}, Q'_{\mathsf{P}})$ be the event that $U_{\mathsf{P}} \cap U'_{\mathsf{P}} \neq \emptyset$ or $V_{\mathsf{P}} \cap V'_{\mathsf{P}} \neq \emptyset$. Accordingly,

$$\mathsf{Pr}\left[Q(\vec{s}, Q_{\mathsf{P}}) \cap Q(\vec{s}, Q'_{\mathsf{P}}) \neq \emptyset\right] \leq \mathsf{Pr}\left[\mathsf{Good}\right] + \mathsf{Pr}\left[Q(\vec{s}, Q_{\mathsf{P}}) \cap Q(\vec{s}, Q'_{\mathsf{P}}) \neq \emptyset \,\middle|\, \overline{\mathsf{Good}}\right] .$$

The latter probability is the probability that there exist $(i, U) \in U_{\mathsf{P}}$, $(i, U') \in U'_{\mathsf{P}}$ such that $\mathsf{P}_1(s_i^{\mathsf{in}}, U) = \mathsf{P}_1(s_i^{\mathsf{in}}, U')$ and $U \neq U'$, or there exist $(i, V) \in V_{\mathsf{P}}$, $(i, V') \in V'_{\mathsf{P}}$ such that $\mathsf{P}_1(s_i^{\mathsf{out}}, V) = \mathsf{P}_1(s_i^{\mathsf{out}}, V')$ and $V \neq V'$. Note that for any such pairs the probability of such a collision is at most α by α-right universality, and thus the overall probability is $2q^2\alpha$ by the union bound. Therefore, combining this with the above, we get

$$\mathsf{Pr}\left[\mathsf{G}_{13}\right] \leq \sqrt{2q^2\alpha + \sum_L p(L) \cdot \underset{Q_{\mathsf{P}}, Q'_{\mathsf{P}} \xleftarrow{\$} \mathcal{Q}_L}{\mathsf{Pr}}\left[\mathsf{Good}(Q_{\mathsf{P}}, Q'_{\mathsf{P}})\right]} \tag{4.5}$$

To conclude the proof, we consider the predictor P against S which simply, given L, samples Q'_{P} from $\mathcal{Q}_L$, and then compute $U'_{\mathsf{P}}, V'_{\mathsf{P}}$, and outputs them. Since the original queries Q_{P} are also distributed according to $\mathcal{Q}_L$, we can rewrite (4.5) as

$$\mathsf{Pr}\left[\mathsf{G}_{13}\right] \leq \sqrt{2q^2\alpha + \mathsf{Adv}^{\mathsf{p\text{-}pred}}_{2n,S}(P)} , \tag{4.6}$$

which concludes the proof. ■

4.5 The Case of Reset-Secure Sources

Theorem 10's importance stems mostly from the fact that it establishes the equivalence of psPRPs and UCEs for the case of (statistically) unpredictable sources. The question was left open in [38]. Many applications (e.g., instantiating the permutation within sponges, or any other indifferentiable hash construction) however require the stronger notion of reset-security. For this, [38] show that the five-round Feistel construction suf-

fices, using a weaker variant of indifferentiability, and left open the question of whether four-rounds suffice.

We do better here: we prove that the NR construction transforms a UCE for statistically reset-secure sources into a psPRP for the same class of sources. The proof starts as the one of Theorem 10, but then shows that the source $\overline{S}$ built therein is in fact statistically reset-secure whenever S is. This step will resort to a variant of the heavy-query sampling method of Impagliazzo and Rudich [95] to simulate a random oracle from the leakage which captures "relevant correlations" with what is learnt by the source.

Theorem 11 (NR security for reset-secure sources) *Let* P *be a* α*-right universal family of permutations on* $2n$*-bit strings, and let* H *be a function family from* $n+1$ *bits to* n *bits. Then, for all distinguishers* D *and sources* S *making overall* q *queries to their oracle, there exists* $\overline{D}$ *and* $\overline{S}$ *such that*

$$\mathsf{Adv}^{\mathsf{psprp}}_{2n,\mathsf{NR}[\mathsf{P},\mathsf{H}]}(S,D) \leq \mathsf{Adv}^{\mathsf{uce}}_{n+1,n,\mathsf{H}}(\overline{S},\overline{D}) + q^2\left(2\alpha + \frac{1}{2^{2n}}\right) . \tag{4.7}$$

Here, $\overline{D}$ *and* D *are roughly as efficient, and* $\overline{S}$ *and* S *are similarly as efficient. In particular,* $\overline{S}$ *makes* $2q$ *queries. Moreover, for every reset-adversary* $\overline{R}$ *making* p *queries, there exists a reset-adversary* R *such that*

$$\mathsf{Adv}^{\mathsf{f\text{-}reset}}_{n+1,n,\overline{S}}(\overline{R}) \leq 2\mathsf{Adv}^{\mathsf{p\text{-}reset}}_{2n,S}(R) + 4\left(q + \frac{8qp^2}{\varepsilon}\ln(4p/\varepsilon)\right)^2\left(2\alpha + \frac{1}{2^{2n}}\right) , \tag{4.8}$$

where $\varepsilon := \mathsf{Adv}^{\mathsf{f\text{-}reset}}_{n+1,n,\overline{S}}(\overline{R})$*. In particular,* R *makes* $4qp^2/\varepsilon \cdot \ln(4p/\varepsilon)$ *queries to its oracle.*

Asymptotically, (4.8) implies that if $\overline{R}$ exists making $p = \mathsf{poly}(\lambda)$ queries, and achieving non-negligible advantage ε, then R makes also a polynomial number of queries, and achieves non-negligible advantage, as long as α is negligible, and $n = \omega(\log\lambda)$. Thus, reset-security of R yields reset-security of $\overline{R}$.

We also believe that the technique of Barak and Mahmoody [100] can be used to reduce the $8qp^2/\varepsilon$ term to $O(qp)/\varepsilon$. We did not explore this avenue here, as the proof approach of [95] is somewhat easier to adapt to our setting.

Proof: The setup of the proof is identical to that in Theorem 10, in particular the construction of $\overline{S}$ from S (and of $\overline{D}$ from D.) The difference is in relating the reset-security of S and $\overline{S}$. In particular, let

$$\varepsilon := \mathsf{Adv}^{\mathsf{f\text{-}reset}}_{n+1,n,\overline{S}}(\overline{R}) = \mathsf{Pr}\left[\mathsf{f\text{-}Reset}^{\overline{R}}_{n+1,n,\overline{S}} \,\middle|\, b=0\right] - \mathsf{Pr}\left[\neg\mathsf{f\text{-}Reset}^{\overline{R}}_{n+1,n,\overline{S}} \,\middle|\, b=1\right] .$$

The RHS is the difference of the probabilities of $\overline{R}$ outputting 0 in the cases $b = 0$ and $b = 1$ respectively. We are going to build a new adversary R against S which satisfies (4.8). We assume without loss of generality that $\overline{R}$ is deterministic, and makes *exactly* p *distinct* queries to its oracle.

We start the proof with some game transitions that will lead naturally to the definition of the adversary R. Formal descriptions are found in Figures 4.6 and 4.7 in the Appendix – our description here is self-contained.

The initial game G_1 is simply $\mathsf{f\text{-}Reset}^{\overline{R}}_{\overline{S}}$ with the bit $b = 0$, i.e., $\overline{S}$ and $\overline{R}$ access the *same* functions $f_1, \ldots, f_r$ here. Further, G_1 returns true iff $\overline{R}$ returns 0. Thus, $\mathsf{Pr}[\mathsf{G}_1] = \mathsf{Pr}\left[\mathsf{f\text{-}Reset}^{\overline{R}}_{\overline{S}} \,\middle|\, b=0\right]$. Game G_2 slightly changes G_1: It keeps track (in a set Q_{P}) of the triples (i, U, V) describing $\overline{\mathcal{O}}$ queries made by the simulated S within $\overline{S}$; i.e., either S queried $(i, (\mathsf{+}, U))$, and obtained V, or queries $(i, (\mathsf{-}, V))$, and obtained U. After $\overline{S}$ terminates with leakage $(L, \vec{s})$, where $\vec{s} = (s^{\mathsf{in}}_1, s^{\mathsf{out}}_1, \ldots, s^{\mathsf{in}}_r, s^{\mathsf{out}}_r)$, for every $(i, U, V) \in Q_{\mathsf{P}}$ we compute $x_0 \,\|\, x_1 \leftarrow \mathsf{P}(s^{\mathsf{in}}_i, U)$ and $x_3 \,\|\, x_2 \leftarrow \mathsf{P}(s^{\mathsf{out}}_i, V)$, and define table entries

$$T[i, 0 \,\|\, x_1] \leftarrow x_0 \oplus x_2 \ , \quad T[i, 1 \,\|\, x_2] \leftarrow x_1 \oplus x_3 \ .$$

For later reference, we denote by X the set of pairs (i, x) for which we set $T[i, x]$ using Q_{P} and $\vec{s}$. We then run $\overline{R}(L, \vec{s})$, and answer its oracle queries (i, x) using $T[i, x]$. If the

entry is undefined, then we return a random value. (As we assumed all of $\overline{R}$'s queries are distinct, we do not need to remember the output.) As before, G_2 outputs true iff $\overline{R}$ outputs 0.

Note that we always have $T[i,x] = f_i(x)$ for very (i,x) such that $f_i(x)$ was queried by $\overline{S}$, and re-sampling values un-queried by $\overline{S}$ upon $\overline{R}$'s queries does not change the distribution of $\overline{R}$'s output, and hence $\mathsf{Pr}[\mathsf{G}_1] = \mathsf{Pr}[\mathsf{G}_2]$.

The intersection sampler. The game G_3 generates a surrogate for Q_{P}. This is the output of an algorithm Sam which, after $\overline{S}$ terminates with output $(L, \vec{s})$, takes as input the leakage L (crucially, not $\vec{s}$!) and an iteration parameter $\eta = 4p/\varepsilon \ln(4p/\varepsilon)$ (we let also $\tau = p \cdot \eta$). Sam queries the very same $\overline{\mathcal{O}}$ implemented by $\overline{S}$ to answer S's queries (which internally simulates the NR construction using $\overline{S}$'s own oracle), and returns a set $\widetilde{Q}_{\mathsf{P}}$ of 4-tuples (i,U,V,j) such that $j \in [p]$, and (i,U,V) is such that $\overline{\mathcal{O}}(i,(+,U))$ would return V (or equivalently $\overline{\mathcal{O}}(i,(-,V))$ would return U). Internally, Sam will make calls to another (randomized) sub-procedure $\mathcal{Q}$ which takes as input the leakage L, as well as a set Q of tuples (i,U,V,j) consistent with $\overline{\mathcal{O}}$, and returns a set Δ of at most q tuples (i,U,V), which are not necessarily consistent with $\overline{\mathcal{O}}$. We will specify in detail later below what $\mathcal{Q}$ exactly does, as some further game transitions will come handy to set up proper notation. For now, a generic understanding will suffice. In particular, given such $\mathcal{Q}$, Sam operates as in Figure 4.4. As we can see, for each $(i,U,V,j) \in \widetilde{Q}_{\mathsf{P}}$, j indicates the outer iteration in which this query was added to $\widetilde{Q}_{\mathsf{P}}$. Using this information, for every $j \in [p]$, and every 4-tuple (i,U,V,j) we compute $x_0 \,\|\, x_1 \leftarrow \mathsf{P}(s_i^{\mathsf{in}}, U)$ and $x_3 \,\|\, x_2 \leftarrow \mathsf{P}(s_i^{\mathsf{out}}, V)$, define

$$\widetilde{T}[i, 0 \,\|\, x_1] \leftarrow x_0 \oplus x_2 \;, \quad \widetilde{T}[i, 1 \,\|\, x_2] \leftarrow x_1 \oplus x_3 \;,$$

ALGORITHM $\mathsf{Sam}^{\overline{\mathcal{O}}}(L, \tau)$:

$\widetilde{Q}_{\mathsf{P}} \leftarrow \emptyset$

for $j = 1$ to p **do**

  for $k = 1$ to η **do**

    $\Delta_{j,k} \leftarrow \mathcal{Q}(L, \widetilde{Q}_{\mathsf{P}})$

    for all $(i, U, V) \in \Delta_{j,k}$ **do**

      $V' \leftarrow \overline{\mathcal{O}}(i, (+, U))$, $U' \leftarrow \overline{\mathcal{O}}(i, (-, V))$, $\widetilde{Q}_{\mathsf{P}} \xleftarrow{\cup} \{(i, U, V', j), (i, U', V, j)\}$

return $\widetilde{Q}_{\mathsf{P}}$

Figure 4.4: Description of algorithm Sam.

and add $(i, 0 \,\|\, x_1), (i, 1 \,\|\, x_2)$ to the set $\widetilde{X}^j$. A for now irrelevant caveat is that if one of the entries in $\widetilde{T}$ is already set, then we do not overwrite it.[5]

Then, after all of this, G_3 resumes by executing $\overline{R}(L, \vec{s})$. For $\overline{R}$'s j-th query (i, x) we do the following:

1. If $(i, x) \in \widetilde{X}^{j'}$ for some $j' \leq j$, then we respond with $\widetilde{T}[i, x]$.

2. Otherwise, if $(i, x) \in X$, but the first condition was not met, we respond with $T[i, x]$.

3. Finally, if neither of the above is true, we respond randomly.

As before, G_3 outputs true iff $\overline{R}$ outputs 0. For now, all modifications are syntactical. Indeed, up to the point we start $\overline{R}$, we satisfy the invariant that $T[i, x] = f_i(x)$ or $\widetilde{T}[i, x] = f_i(x)$ whenever these are defined, because $\overline{\mathcal{O}}$ behaves according to the NR construction using $\vec{s}$. On the other hand, if during the execution $\overline{R}(L, \vec{s})$ we respond

[5]This does not matter here, as an entry can only be overwritten with the same value; below, we will change the experiment in a way that overwrites may be inconsistent, and we want to ensure we agree to keep the first value.

randomly, we know for sure $f_i(x)$ was not queried by $\overline{S}$, and thus we can re-sample it. Thus, $\mathsf{Pr}[\mathsf{G}_3] = \mathsf{Pr}[\mathsf{G}_2] = \mathsf{Pr}[\mathsf{G}_1]$.

Moving to G_4, we now answer $\overline{\mathcal{O}}$ queries by S (within $\overline{S}$) and by Sam using random permutations $\pi_1, \ldots \pi_r$, instead of simulating the NR construction using $f_1, \ldots, f_r$, i.e., $\overline{\mathcal{O}}(i, (+, U)) = \pi_i(U)$ and $\overline{\mathcal{O}}(i, (-, V)) = \pi_i^{-1}(V)$. The seeds $\vec{s}$ are now independent of $\overline{\mathcal{O}}$. We do not change anything else. We note that the indistinguishability of G_3 and G_4 directly reduces to a suitable distinguisher for Proposition 5, as only Sam and S (within $\overline{S}$) make queries to $\overline{\mathcal{O}}$, but they do not get the keys $\vec{s}$, which are used only after all queries to $\overline{O}$ have been made to define X and $\widetilde{X}$. Therefore,

$$\mathsf{Pr}[\mathsf{G}_1] = \mathsf{Pr}[\mathsf{G}_3] \leq \mathsf{Pr}[\mathsf{G}_4] + (q + 2q\tau)^2 \left(2\alpha + \frac{1}{2^{2n}}\right) , \tag{4.9}$$

where we have used the fact that Sam makes $2q\tau = 2pq\eta$ queries.

The final game is G_5 is identical to G_4, *except* that in the process of answering $\overline{R}$'s queries, if case 2 happens, we also set answer randomly. However, should such situation occur, a bad flag is set in G_5, and since up to the point this flag is set, the behavior of G_4 and G_5 is identical,

$$\mathsf{Pr}[\mathsf{G}_4] - \mathsf{Pr}[\mathsf{G}_5] \leq \mathsf{Pr}[\mathsf{G}_5 \text{ sets } \mathsf{bad}] \ .$$

To analyze the probability on the RHS, we need to specify $\mathcal{Q}(L, \widetilde{Q})$ used by Sam here. (Note all statements so far were independent of it.) For a given L which appears with positive probability in G_5, consider the distribution of the input-output pairs Q_{P} defined by the interaction of S with $\overline{\mathcal{O}}$, conditioned on the leakage being L, and $\pi_1, \ldots, \pi_r$ being consistent with the triples defined by $\widetilde{Q}$. Then, $\mathcal{Q}(L, \widetilde{Q})$ outputs a sample of Q_{P} according to this distribution. Using this, we prove the following lemma in Appendix 4.5.1, which uses ideas similar to those from [95], with some modifications due to the setting (and the fact that $\overline{R}$ makes p queries).

ADVERSARY $R^{\mathcal{O}}(1^r, L)$:

$c \leftarrow 0$, $\widetilde{X} \leftarrow \emptyset$

$\vec{s} = (s_1^{\mathsf{in}}, s_1^{\mathsf{out}}, \ldots, s_r^{\mathsf{in}}, s_r^{\mathsf{out}}) \xleftarrow{\$} \mathsf{P.Kg}$

$\widetilde{Q}_{\mathsf{P}} \xleftarrow{\$} \mathsf{Sam}^{\mathcal{O}}(L)$

for $j = 1$ to p **do**

 for all $(i, U, V, j) \in \widetilde{Q}$ **do**

 $x_0 \,\|\, x_1 \leftarrow \mathsf{P}(s_i^{\mathsf{in}}, U)$, $x_3 \,\|\, x_2 \leftarrow \mathsf{P}(s_i^{\mathsf{out}}, V)$

 $\widetilde{X}^j \leftarrow \widetilde{X}^j \cup \{(i, 0 \,\|\, x_1), (i, 1 \,\|\, x_2)\}$

 if $\widetilde{T}[i, 0 \,\|\, x_1] = \perp$ **then** $\widetilde{T}[i, 0 \,\|\, x_1] \leftarrow x_0 \oplus x_2$

 if $\widetilde{T}[i, 1 \,\|\, x_2] = \perp$ **then** $\widetilde{T}[i, 1 \,\|\, x_2] \leftarrow x_1 \oplus x_3$

$b' \leftarrow \overline{R}^{\mathcal{O}'}(L, \vec{s})$

return b'

Proc. $\mathcal{O}'(i, x)$:

$c \leftarrow c + 1$, $\widetilde{X} \leftarrow \widetilde{X} \cup \widetilde{X}^c$

if $T[i, x] = \perp$ **then**

 if $(i, x) \in \widetilde{X}$ **then**

 $T[i, x] \leftarrow \widetilde{T}[i, x]$

 else $T[i, x] \xleftarrow{\$} \{0,1\}^n$

return $T[i, x]$

Figure 4.5: Adversary R in the proof of Theorem 11.

Lemma 31 $\mathsf{Pr}\left[\mathsf{G}_5 \text{ sets } \mathsf{bad}\right] \leq \varepsilon/2$

Given this, we are now ready to give our adversary R, which we build from $\overline{R}$ and Sam as described in Figure 4.5. By a purely syntactical argument,

$$\mathsf{Pr}\left[\mathsf{G}_5\right] = \mathsf{Pr}\left[\mathsf{p\text{-}Reset}_S^R \,\middle|\, b = 0\right] \ , \tag{4.10}$$

recalling that the case $b = 0$ is the one where both S and R access the same permutations $\pi_1, \ldots, \pi_r$. Therefore, we have established, combining (4.10), (4.9), Lemma 31,

$$\mathsf{Pr}\left[\mathsf{p\text{-}Reset}_S^R \,\middle|\, b = 0\right] \geq \mathsf{Pr}\left[\mathsf{f\text{-}Reset}_{\overline{S}}^{\overline{R}} \,\middle|\, b = 0\right] - \frac{\varepsilon}{2} - (q + 2q\tau)^2 \left(2\alpha + \frac{1}{2^{2n}}\right) \ . \tag{4.11}$$

In Appendix 4.5.2 we also prove formally that in the case $b = 1$, R in the game $\mathsf{p\text{-}Reset}_S^R$ almost perfectly simulates an execution of $\mathsf{f\text{-}Reset}_{\overline{S}}^{\overline{R}}$, or more formally,

$$\mathsf{Pr}\left[\neg\mathsf{p\text{-}Reset}_S^R \,\middle|\, b = 1\right] \leq \mathsf{Pr}\left[\neg\mathsf{f\text{-}Reset}_{\overline{S}}^{\overline{R}} \,\middle|\, b = 1\right] + (q + 2q\tau)^2 \left(2\alpha + \frac{1}{2^{2n}}\right) \ . \tag{4.12}$$

We can combine (4.12) and (4.11) to obtain, with $\Delta = 2(q+2q\tau)^2\left(2\alpha + \frac{1}{2^{2n}}\right)$,

$$\begin{aligned}\mathsf{Adv}^{\mathsf{p\text{-}reset}}_{2n,S}(R) &= \mathsf{Pr}\left[\mathsf{p\text{-}Reset}^{R}_{S} \,\middle|\, b=0\right] - \mathsf{Pr}\left[\neg\mathsf{p\text{-}Reset}^{R}_{S} \,\middle|\, b=1\right] \\ &\geq \mathsf{Pr}\left[\mathsf{f\text{-}Reset}^{\overline{R}}_{\overline{S}} \,\middle|\, b=0\right] - \mathsf{Pr}\left[\neg\mathsf{f\text{-}Reset}^{\overline{R}}_{\overline{S}}) \,\middle|\, b=1\right] - \frac{\varepsilon}{2} - \Delta \\ &\geq \varepsilon/2 - \Delta\ .\end{aligned}$$

This concludes the proof.

■

4.5.1 Proof of Lemma 31

Proof: [Of Lemma 31] Recall that $\overline{R}$ makes exactly p distinct queries, and is deterministic. Also, recall that Sam goes through p outer iterations, and $\eta = 4p/\varepsilon \ln(4p/\varepsilon)$ inner iterations, and consequently we denote an iteration of Sam with a pair (j,k), where $j \in [p]$ and $k \in [\eta]$.

Also, in the following, let us *fix* the leakage L, the keys $\vec{s}$ for P, and the values $Y_1, \ldots, Y_p$, where Y_i is the uniformly sampled value used to answer $\overline{R}$'s i-th query in G_5 if the query is not in $\widetilde{X}$. (We can think of the values $Y_1, \ldots, Y_p$ as being pre-sampled, and then used as needed.) We will prove an upper bound on bad being set, even given these are fixed. The final bound then follows by averaging.

Some more notation. For every $(j,k) \in [p] \times [\eta]$, we denote by $\widetilde{Q}_{\mathsf{P},j,k}$ the contents of $\widetilde{Q}_{\mathsf{P}}$ within Sam at the end of iteration (j,k). Further, we let $\widetilde{Q}_{\mathsf{P},1,0} = \emptyset$, and $\widetilde{Q}_{\mathsf{P},j,0} = \widetilde{Q}_{\mathsf{P},j-1,\eta}$ for all $j \geq 2$. Note that the final output is $\widetilde{Q}_{\mathsf{P}} = \widetilde{Q}_{\mathsf{P},p,\eta}$. Also, for $\vec{s}$ defined as in the games, and any set Q of triples (i,U,V), we define for convenience $X(\vec{s},Q)$ as the set

$$\begin{aligned}X(\vec{s},Q) = \left\{(i,0\,\|\,x_1) \,:\, \exists (i,U,V,j) \in Q :\ \mathsf{P}_1(s_i^{\mathsf{in}},U) = x_1\right\} \\ \cup \left\{(i,1\,\|\,x_2) \,:\, \exists (i,U,V,j) \in Q :\ \mathsf{P}_1(s_i^{\mathsf{out}},V) = x_2\right\}\ .\end{aligned}$$

MAIN G_1:

$Q_{\mathsf{P}} \leftarrow \emptyset$, $(r,t) \xleftarrow{\$} S(\varepsilon)$

$f_1, \ldots, f_r \xleftarrow{\$} \mathsf{Funcs}(n+1, n)$

$\vec{s} = (s_1^{\mathsf{in}}, s_1^{\mathsf{out}}, \ldots, s_r^{\mathsf{in}}, s_r^{\mathsf{out}}) \xleftarrow{\$} \mathsf{P.Kg}$

$L \xleftarrow{\$} S^{\overline{\mathcal{O}}}(t)$

$b' \leftarrow \overline{R}^{\mathcal{O}'}(L, \vec{s})$

return $(b' = 0)$

MAIN G_2:

$Q_{\mathsf{P}} \leftarrow \emptyset$, $(r,t) \xleftarrow{\$} S(\varepsilon)$

$f_1, \ldots, f_r \xleftarrow{\$} \mathsf{Funcs}(n+1, n)$

$\vec{s} = (s_1^{\mathsf{in}}, s_1^{\mathsf{out}}, \ldots, s_r^{\mathsf{in}}, s_r^{\mathsf{out}}) \xleftarrow{\$} \mathsf{P.Kg}$

$L \xleftarrow{\$} S^{\overline{\mathcal{O}}}(t)$

for all $(i, U, V) \in Q_{\mathsf{P}}$ **do**

  $x_0 \,\|\, x_1 \leftarrow \mathsf{P}(s_i^{\mathsf{in}}, U)$

  $x_2 \,\|\, x_2 \leftarrow \mathsf{P}(s_i^{\mathsf{out}}, V)$

  $T[i, 0 \,\|\, x_1] \leftarrow x_0 \oplus x_2$

  $T[i, 1 \,\|\, x_2] \leftarrow x_1 \oplus x_3$

  $X \xleftarrow{\cup} \{(i, 0 \,\|\, x_1), (i, 1 \,\|\, x_2)\}$

$b' \leftarrow \overline{R}^{\mathcal{O}'}(L, \vec{s})$

return $(b' = 0)$

Proc. $\overline{\mathcal{O}}(i, (\sigma, U))$: // G_1, G_2, G_3

if $\sigma =$ **+ then**

  $x_0 \,\|\, x_1 \leftarrow \mathsf{P}(s_i^{\mathsf{in}}, U)$

  $x_2 \leftarrow f_i(0 \,\|\, x_1) \oplus x_0$

  $x_3 \leftarrow f_i(1 \,\|\, x_2) \oplus x_1$

  $V \leftarrow \mathsf{P}^{-1}(s_i^{\mathsf{out}}, x_3 \,\|\, x_2)$

  $Q_{\mathsf{P}} \leftarrow Q_{\mathsf{P}} \cup \{(i, U, V)\}$

else

  $x_3 \,\|\, x_2 \leftarrow \mathsf{P}(s_i^{\mathsf{out}}, U)$

  $x_1 \leftarrow f_i(1 \,\|\, x_2) \oplus x_3$

  $x_0 \leftarrow f_i(0 \,\|\, x_1) \oplus x_2$

  $V \leftarrow \mathsf{P}^{-1}(s_i^{\mathsf{in}}, x_0 \,\|\, x_1)$

  $Q_{\mathsf{P}} \leftarrow Q_{\mathsf{P}} \cup \{(i, V, U)\}$

return V

Proc. $\mathcal{O}'(i, x)$: // G_2

if $T[i, x] \neq \perp$ **then** $y \leftarrow T[i, x]$

else $y \xleftarrow{\$} \{0,1\}^n$

return y

Proc. $\mathcal{O}'(i, x)$: // G_1

return $f_i(X)$.

Figure 4.6: Games to prove reset-security of the source $\overline{S}$. Here, $\overline{R}$ is deterministic and assumed to make p distinct queries, and thus it is not necessary for the oracle $\mathcal{O}'$ to keep state.

In particular, $X(\vec{s}, \widetilde{Q}_{\mathsf{P},j,\eta}) = \widetilde{X}^1 \cup \cdots \cup \widetilde{X}^j$. For this reason, we note in particular that for every $j \in [p]$, the j-th query q_j of $\overline{R}$ is uniquely determined by L, $\vec{s}$, $Y_1, \ldots, Y_{j-1}$ and $\widetilde{Q}_{\mathsf{P},j,0}$. This fact relies on $\overline{R}$ being deterministic, and somewhat more subtly, the fact that the values $\widetilde{T}$ are never overwritten.

MAIN $\mathsf{G}_3, \mathsf{G}_4, \mathsf{G}_5$:

$c \leftarrow 0$

$Q_{\mathsf{P}} \leftarrow \emptyset$, $(r,t) \xleftarrow{\$} S(\varepsilon)$

$\pi_1, \ldots, \pi_r \xleftarrow{\$} \mathsf{Perms}(2n)$

$f_1, \ldots, f_r \xleftarrow{\$} \mathsf{Funcs}(n+1, n)$

$\vec{s} = (s_1^{\mathsf{in}}, s_1^{\mathsf{out}}, \ldots, s_r^{\mathsf{in}}, s_r^{\mathsf{out}}) \xleftarrow{\$} \mathsf{P.Kg}$

$L \xleftarrow{\$} S^{\overline{\mathcal{O}}}(t)$

$\widetilde{Q}_{\mathsf{P}} \xleftarrow{\$} \mathsf{Sam}^{\overline{\mathcal{O}}}(L)$

for all $(i, U, V) \in Q_{\mathsf{P}}$ **do**

  $x_0 \,\|\, x_1 \leftarrow \mathsf{P}(s_i^{\mathsf{in}}, U)$

  $x_2 \,\|\, x_2 \leftarrow \mathsf{P}(s_i^{\mathsf{out}}, V)$

  $T[i, 0 \,\|\, x_1] \leftarrow x_0 \oplus x_2$

  $T[i, 1 \,\|\, x_2] \leftarrow x_1 \oplus x_3$

  $X \xleftarrow{\cup} \{(i, 0 \,\|\, x_1), (i, 1 \,\|\, x_2)\}$

for $j = 1$ to p **do**

  for all $(i, U, V, j) \in \widetilde{Q}_{\mathsf{P}}$ **do**

    $x_0 \,\|\, x_1 \leftarrow \mathsf{P}(s_i^{\mathsf{in}}, U)$

    $x_3 \,\|\, x_2 \leftarrow \mathsf{P}(s_i^{\mathsf{out}}, V)$

    $\widetilde{X}^j \xleftarrow{\cup} \widetilde{X}^j \cup \{(i, 0 \,\|\, x_1), (i, 1 \,\|\, x_2)\}$

    if $\widetilde{T}[i, 0 \,\|\, x_1] = \perp$ **then**

      $\widetilde{T}[i, 0 \,\|\, x_1] \leftarrow x_0 \oplus x_2$

    if $\widetilde{T}[i, 1 \,\|\, x_2] = \perp$ **then**

      $\widetilde{T}[i, 1 \,\|\, x_2] \leftarrow x_1 \oplus x_3$

$b' \leftarrow \overline{R}^{\mathcal{O}'}(L, \vec{s})$

return $(b' = 0)$

Proc. $\mathcal{O}'(i, x)$: // G_3, G_4

$c \leftarrow c + 1$, $\widetilde{X} \xleftarrow{\cup} \widetilde{X}^c$

if $(i, x) \in \widetilde{X}$ **then** $y \leftarrow \widetilde{T}[i, x]$

elseif $(i, x) \in X$ **then**

  $\mathsf{bad} \leftarrow \mathsf{true}$, $y \leftarrow T[i, X]$

elseif $y \xleftarrow{\$} \{0,1\}^n$

return y

Proc. $\mathcal{O}'(i, x)$: // G_5

$c \leftarrow c + 1$, $\widetilde{X} \xleftarrow{\cup} \widetilde{X}^c$

if $(i, x) \in \widetilde{X}$ **then** $y \leftarrow \widetilde{T}[i, x]$

elseif $(i, x) \in X$ **then**

  $\mathsf{bad} \leftarrow \mathsf{true}$, $y \xleftarrow{\$} \{0,1\}^n$

elseif $y \xleftarrow{\$} \{0,1\}^n$

return y

Proc. $\overline{\mathcal{O}}(i, (\sigma, U))$: // $\mathsf{G}_4, \mathsf{G}_5$

if $\sigma =$ **+ then**

  $V \xleftarrow{\$} \pi_i(U)$

  $Q_{\mathsf{P}} \leftarrow Q_{\mathsf{P}} \cup \{(i, U, V)\}$

else

  $V \xleftarrow{\$} \pi_i^{-1}(U)$

  $Q_{\mathsf{P}} \leftarrow Q_{\mathsf{P}} \cup \{(i, V, U)\}$

return V

Figure 4.7: Games to prove reset-security of the source $\overline{S}$. (Continued) Here, $\overline{R}$ is deterministic and assumed to make p distinct queries, and thus it is not necessary for the oracle $\mathcal{O}'$ to keep state.

The bad events. For $j \in [p]$, we also define the event bad_j that for the j-th query of $\overline{R}$ we have $\mathsf{q}_j \notin \widetilde{X}^1 \cup \cdots \cup \widetilde{X}^j$, and $\mathsf{q}_j \in X_{\mathsf{P}}$, which is exactly the event that bad is set when answering the j-th query of $\overline{R}$. We will show that $\mathsf{Pr}\left[\mathsf{bad}_j\right] \leq \varepsilon/2p$, and consequently

$$\mathsf{Pr}\left[\mathsf{bad}\right] \leq \sum_{j=1}^{p} \mathsf{Pr}\left[\mathsf{bad}_j\right] \leq p \cdot \varepsilon/2p = \varepsilon/2 \ . \tag{4.13}$$

To bound $\mathsf{Pr}\left[\mathsf{bad}_j\right]$ for any fixed $j \in [p]$, assume that we further fix $\widetilde{Q}_{\mathsf{P},j,0}$ to some possible value which can occur with positive probability. This also (by the above argument) uniquely fixed the j-th query q_j.

Now, if $\mathsf{q}_j \in X(\vec{s}, \widetilde{Q}_{\mathsf{P},j,0})$, then $\mathsf{Pr}\left[\mathsf{bad}_j\right]$ is 0 conditioned on this. Therefore, let us assume that $\mathsf{q}_j \notin X(\vec{s}, \widetilde{Q}_{\mathsf{P},j,0})$. Then, define for every $k \in [\eta]$,

$$\alpha_{j,k} = \mathsf{Pr}\left[\Delta_j \xleftarrow{\$} \mathcal{Q}(L, \widetilde{Q}_{\mathsf{P},j,k-1}) \ : \ \mathsf{q}_j \in X(\vec{s}, \Delta_j)\right] \ .$$

Now, either there exists $k \in [\eta]$ such that $\alpha_{j,k} \leq \varepsilon/4p$, in which case the probability that $\mathsf{q}_j \in X$ is at most $\varepsilon/4p$, and thus the probability of bad_j conditioned on this is at most $\varepsilon/4p$. This follows from the following fact (made explicit in [95]): $\alpha_{j,k}$ is exactly the conditional probability, given the acquired information so far, that $\mathsf{q}_j \in X$ in the actual execution, and thus, conditioned on $\alpha_{j,k} = p$, the probability that $\mathsf{q}_j \in X$ is indeed p.

The other case is that no such k exists, i.e., $\alpha_{j,k} > \frac{\varepsilon}{4p}$ for all $k \in [\eta]$. In this case, we are going to necessarily have $\mathsf{q}_j \in X(\widetilde{Q}_{\mathsf{P},j,\eta})$, except with probability

$$(1 - \varepsilon/4p)^{\eta} \leq e^{-(\varepsilon/4p)\eta} = \varepsilon/4p$$

using the fact that $\eta = 4p/\varepsilon \ln(4p/\varepsilon)$. Therefore, the probability that bad_j is set is at most $2 \times \varepsilon/4p = \varepsilon/2p$.

This concludes the proof of Lemma 31. ■

4.5.2 Analysis of the $b = 1$ case

To conclude the proof, we still need to analyze the $b = 1$ case, i.e., the case where the source and the reset adversary are given independent oracles. To this end, we introduce two games $\mathsf{G}_{11}, \mathsf{G}_{12}$ which simulate an execution of R against S where in the former game, R and S access independent sets of permutations, whereas in G_{12} these permutations are instantiated using the NR construction, sharing however the underlying keys $\vec{s}$ for P. Clearly,

$$\mathsf{Pr}\left[\neg\mathsf{p\text{-}Reset}_S^R \,\middle|\, b = 1\right] = \mathsf{Pr}\left[\mathsf{G}_{11}\right] \ . \tag{4.14}$$

To transition from G_{11} to G_{12}, we can use Proposition 5 *twice*, first with respect to S's q queries to replace $\pi_1^0, \pi_2^0, \ldots$ with their counterparts obtained from the NR construction, and then another time to do the same $\pi_1^1, \pi_1^1, \ldots$ (this time there are $2q\tau$ queries being made by Sam to take into account). So overall,

$$\begin{aligned}\mathsf{Pr}\left[\mathsf{G}_{11}\right] &\leq \mathsf{Pr}\left[\mathsf{G}_{12}\right] + (q^2 + (2q\tau)^2)\left(2\alpha + \frac{1}{2^{2n}}\right) \\ &\leq \mathsf{Pr}\left[\mathsf{G}_{12}\right] + (q + 2q\tau)^2\left(2\alpha + \frac{1}{2^{2n}}\right) \ .\end{aligned} \tag{4.15}$$

Now, looking at G_{12}, we observe that $\overline{T}[i, x] = f_i^1(x)$ always holds by construction. Therefore, when running $\overline{R}$, the values returned by $\mathcal{O}'(i, x)$ are always either equal $f_i^b(x)$ *or* freshly random, and thus in both case random and independent than the values of $f_1^0, \ldots, f_r^0$. In other words, this perfectly simulates an execution of $\overline{R}$ against $\overline{S}$ in the $b = 1$ case, i.e.

$$\mathsf{Pr}\left[\mathsf{G}_{12}\right] = \mathsf{Pr}\left[\neg\mathsf{f\text{-}Reset}_{\overline{S}}^{\overline{R}} \,\middle|\, b = 1\right] \ , \tag{4.16}$$

and thus combining (4.14), (4.15), and (4.16), we get

$$\mathsf{Pr}\left[\neg\mathsf{p\text{-}Reset}_S^R \,\middle|\, b = 1\right] \leq \mathsf{Pr}\left[\neg\mathsf{f\text{-}Reset}_{\overline{S}}^{\overline{R}} \,\middle|\, b = 1\right] + (q + 2q\tau)^2\left(2\alpha + \frac{1}{2^{2n}}\right) \ , \tag{4.17}$$

as we wanted to show.

MAIN $\mathsf{G}_{11}, \mathsf{G}_{12}$:

$c \leftarrow 0$

$Q_{\mathsf{P}} \leftarrow \emptyset$, $(r,t) \xleftarrow{\$} S(\varepsilon)$

$\pi_1^0, \pi_1^1, , \ldots, \pi_r^0, \pi_r^1 \xleftarrow{\$} \mathsf{Perms}(2n)$

$f_1^0, f_1^1, \ldots, f_r^0, f_r^1 \xleftarrow{\$} \mathsf{Funcs}(n+1, n)$

$\vec{s} = (s_1^{\mathsf{in}}, s_1^{\mathsf{out}}, \ldots, s_r^{\mathsf{in}}, s_r^{\mathsf{out}}) \xleftarrow{\$} \mathsf{P.Kg}$

$L \xleftarrow{\$} S^{\overline{\mathcal{O}}^0}(t)$

$\widetilde{Q}_{\mathsf{P}} \xleftarrow{\$} \mathsf{Sam}^{\overline{\mathcal{O}}^1}(L)$

for $j = 1$ to p **do**

  for all $(i, U, V, j) \in \widetilde{Q}_{\mathsf{P}}$ **do**

    $x_0 \,\|\, x_1 \leftarrow \mathsf{P}(s_i^{\mathsf{in}}, U)$

    $x_3 \,\|\, x_2 \leftarrow \mathsf{P}(s_i^{\mathsf{out}}, V)$

    $\widetilde{X}^j \xleftarrow{\cup} \widetilde{X}^j \cup \{(i, 0 \,\|\, x_1), (i, 1 \,\|\, x_2)\}$

    if $\widetilde{T}[i, 0 \,\|\, x_1] = \perp$ **then**

      $\widetilde{T}[i, 0 \,\|\, x_1] \leftarrow x_0 \oplus x_2$

    if $\widetilde{T}[i, 1 \,\|\, x_2] = \perp$ **then**

      $\widetilde{T}[i, 1 \,\|\, x_2] \leftarrow x_1 \oplus x_3$

$b' \leftarrow \overline{R}^{\mathcal{O}'}(L, \vec{s})$

return $(b' = 0)$

Proc. $\overline{\mathcal{O}}^b(i, (\sigma, U))$: // G_{11}

if $\sigma =$ + **then** $V \xleftarrow{\$} \pi_i(U)$

else $V \xleftarrow{\$} \pi_i^{-1}(U)$

return V

Proc. $\overline{\mathcal{O}}^b(i, (\sigma, U))$: // G_{12}

if $\sigma =$ + **then**

  $x_0 \,\|\, x_1 \leftarrow \mathsf{P}(s_i^{\mathsf{in}}, U)$

  $x_2 \leftarrow f_i^b(0 \,\|\, x_1) \oplus x_0$

  $x_3 \leftarrow f_i^b(1 \,\|\, x_2) \oplus x_1$

  $V \leftarrow \mathsf{P}^{-1}(s_i^{\mathsf{out}}, x_3 \,\|\, x_2)$

else

  $x_3 \,\|\, x_2 \leftarrow \mathsf{P}(s_i^{\mathsf{out}}, U)$

  $x_1 \leftarrow f_i^b(1 \,\|\, x_2) \oplus x_3$

  $x_0 \leftarrow f_i^b(0 \,\|\, x_1) \oplus x_2$

  $V \leftarrow \mathsf{P}^{-1}(s_i^{\mathsf{in}}, x_0 \,\|\, x_1)$

return V

Proc. $\mathcal{O}'(i, x)$: // G_{11}, G_{12}

$c \leftarrow c + 1$, $\widetilde{X} \xleftarrow{\cup} \widetilde{X}^c$

if $(i, x) \in \widetilde{X}$ **then** $y \leftarrow \widetilde{T}[i, x]$

elseif $y \xleftarrow{\$} \{0,1\}^n$

return y

Figure 4.8: Games to prove reset-security of the source $\overline{S}$. (Continued) Case $b = 1$. Here, $\overline{R}$ is deterministic and assumed to make p distinct queries, and thus it is not necessary for the oracle $\mathcal{O}'$ to keep state.

Chapter 5

Two-round and Non-interactive Non-malleable Commitments

The main result of this chapter is a construction of two-round and non-interactive non-malleable commitments from sub-exponential-time well-studied assumptions. Towards this, we provide a detailed overview of our techniques in Section 5.1. In Section 5.2, we provide preliminaries and definitions. Section 5.3 presents a family of basic commitment schemes that are mutually harder than each other at different axis, we call them size-robust, depth-robust and size-and-depth robust commitments. Using these basic commitment schemes, in Section 5.4, we construct a commitment scheme for short identities that satisfy a weaker notion of non-malleability that we formalize as non-malleability w.r.t. extraction. In Section 5.5, we present a non-malleability strengthening technique that increases the length of identities exponentially, and lifts non-malleability w.r.t. extraction in the stand-alone setting to both non-malleability w.r.t. extraction and standard non-malleability in the concurrent setting. In Section 5.7, we construct 2-round non-malleable commitment scheme for n-bit identities, by iteratively applying the amplification technique in Section 5.5 to the basic scheme in Section 5.4. Then in Section 5.8 we discuss

the robust CCA-security of the 2-round non-malleable commitment scheme described in Section 5.7. Finally in Section 5.9, we show how to remove the first-message in our 2-round non-malleable and robust-CCA secure commitment from Section 5.7 when the attackers are restricted to be uniform Turing machines.

5.1 Overview

Every secure statistically binding commitment scheme is *hiding* against polynomial-sized circuits, while *extractable* by some exponential-sized circuit (such an extractor is guaranteed to exist since one can always find the committed value by brute force). In this work, we pay special attention to the *gap* between the "resources" of attackers and that of extractors. Moreover, we crucially rely on the synergy between different resources — in particular, *circuit-size* and *circuit-depth*, which are captured by the following two basic types of commitment schemes:

Size-Robust Commitments are parametrized versions of classical commitments: An (S, S')-*size-robust commitment* is hiding against any size-$\mathsf{poly}(S)$ attackers, and extractable by some size-S' extractor, for an $S' = S^{\omega(1)}$ denoted as $S' >> S$, of *shallow* polynomial depth where S and S' are some function of the security parameter. For instance, such extractors can be implemented using the naïve brute force strategy of enumerating all possible decommitments, which is a time-consuming but highly-parallelizable task.

Depth-Robust Commitments are natural analogues of size-robust commitments, but with respect to the resource of circuit-depth. A (D, D')-*depth-robust commitment* is hiding against any depth-$\mathsf{poly}(D)$ circuits with size up to a large upper bound B, and extractable by some *size-D'* extractor for $B > D' >> D$ that necessarily has a

depth super-polynomially larger than D. In this work, we consider a subexponential size upper bound $B = 2^{n^\varepsilon}$ for some constant $\varepsilon > 0$; for simplicity of exposition, we ignore this upper bound in the rest of this overview (see Section 5.3 for more detail).

Size-Robust Commitments from Subexponential Injective OWFs. The size-robust commitments we need (for specific relations between S and S') can essentially be instantiated using any off-the-shelf commitment schemes that are subexponentially secure, by appropriately scaling the security parameter to control the levels of security and hardness for extraction. Take the standard non-interactive commitment scheme from any injective one-way function f as an example: A commitment to a bit b is of the form $(f(r), h(r) \oplus b)$, consisting of the image $f(r)$ of a random string r of length n, and the committed bit b XORed with the hard-core bit $h(r)$. Assuming that f is subexponentially hard to invert, the commitment is hiding against all size-2^{n^ε} circuits for some constant $\varepsilon > 0$, while extractable in size 2^n (ignoring polynomial factors in n) and polynomial depth. By setting the security parameter n to $(\log S)^{1/\varepsilon}$, we immediately obtain a (S, S')-size robust commitment for $S' = 2^{\log S^{1/\varepsilon}}$.

Depth-Robust Commitments from Time-Lock Puzzles. Depth-robust commitments are naturally connected with cryptographic objects that consider parallel-time complexity, which corresponds to circuit-depth. When replacing subexponentially-hard one-way functions in the above construction with time-lock puzzles, we immediately obtain depth-robust commitments:

- To commit to a bit b, generate a puzzle puzz with a random solution s and a designated level of hardness t, and hide b using the Goldreich-Levin hard-core bit, producing $C = (\mathsf{puzz}, r, \langle r, s\rangle \oplus b)$ as the commitment.

- To decommit, the committer can simply reveal the puzzle solution s together with the random coins ρ used for generating the puzzle. The receiver verifies that the puzzle is honestly generated with solution s, and uses s to recover the committed bit b.

Since the time-lock puzzle solution s is hidden against adversaries in parallel-time $T(t)$ (and overall time $B(n)$), the commitments are hiding against depth-$T(t)$ adversaries (with size up to $B(n)$). Moreover, since the puzzles can be "forcefully" solved in time 2^t, the committed values can be extracted in size 2^t. This gives a $(T, 2^t)$-depth-robust commitment.[1]

Next, we show how to compose the basic size-robust and depth-robust commitment schemes to overcome Pass's impossibility result on 2-round non-malleable commitments.

5.1.1 Towards Overcoming the Impossibility Result

One-Sided Non-Malleability via Complexity Leveraging. It is well known that *one-sided non-malleability* can be achieved easily via complexity leveraging. One-sided non-malleability only prevents mauling attacks when the tag of the left commitment is "larger than" the tag of the right commitment.[2] In the simple case of 1-bit tags, this requires the commitment for tag 1 (on the left) to be non-malleable w.r.t. the commitment for tag 0 (on the right), which holds if the tag-1 commitment is "harder" than the tag-0 commitment. For example, if the tag-1 commitment is (S_1, S_1')-size-robust while the tag-0 commitment is (S_0, S_0')-size-robust for some $S_0 << S_0' << S_1 << S_1'$, then one can extract the right committed value using a size-S_0' extractor, while the left committed value still remains hidden. Therefore, the right committed value must be (computationally)

[1] Binding follows from the injectivity of time-lock puzzles.

[2] The choice that the left tag is smaller than the right tag is not important. One could also require the opposite, that is, the left tag is larger than the right tag. The limitation is that the design of the commitments depends on this arbitrary decision.

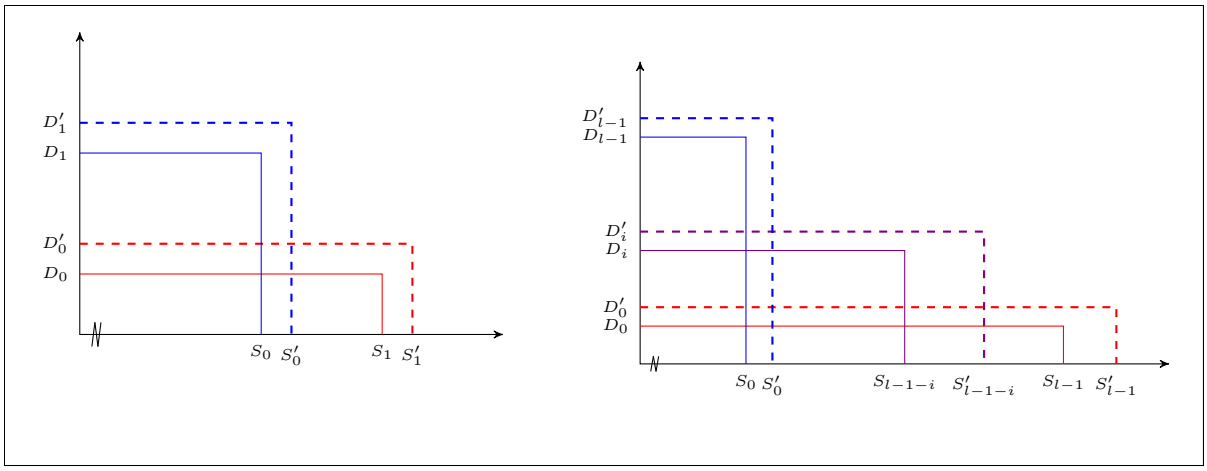


Figure 5.1: (left) A 1-bit tag based commitment scheme: The tag-0 (resp., tag-1) commitment scheme is hiding for circuits of depth below D_0 (resp., D_1) *OR* size below S_1 (resp., S_0), represented by the solid line joining D_0 (resp., D_1) and S_1 (resp., S_0). The tag-0 (resp., tag-1) commitment scheme admits an extractor of depth at most D'_0 (resp., D'_1) and size at most S'_1 (resp., S'_0). (right) This is a generalization of the 1-bit tag commitment scheme to $\log l$-bits tags, where for tag-i the commitment scheme is hiding for circuits of depth below D_i *OR* size below S_{l-1-i} and exhibits an extractor of depth at most D'_i and size at most S'_{l-1-i}.

independent of the left. Similarly, we can also achieve one-sided non-malleability using depth-robust commitments, by using a (D_1, D'_1)-depth robust commitment scheme for tag 1 and a (D_0, D'_0)-depth robust commitment scheme for tag 0, for some $D_0 << D'_0 << D_1 << D'_1$.

However, simple complexity leveraging is inherently limited to one-sided non-malleability, since when only one resource is considered, the tag-1 commitment cannot be both harder and easier than the tag-0 commitment.

Two Resources for (Two-Sided) Non-Malleability. Therefore, our key idea is using two resources to create two "axes", such that, the tag-1 commitment and tag-0 commitment are simultaneously "harder" than the other, but, with respect to differ-

ent resources. This is enough to circumvent the lowerbound due to Pass [67]. This is achieved by combining the basic size-robust and depth-robust commitment schemes in the following simple way.

Basic 1-bit Tag Non-Malleable Commitment:

For some $D_0 << D'_0 << D_1 << D'_1 << S_0 << S'_0 << S_1 << S'_1$,

- a tag-0 commitment to a value v consists of commitments to two random, xor secret shares α, β of v, such that, $v = \alpha + \beta$, where the first share is committed under a (D_0, D'_0)-depth-robust commitment scheme and the second under a (S_1, S'_1)-size-robust commitment scheme, and

- a tag-1 commitment to v, on the other hand, uses a (D_1, D'_1)-depth-robust commitment scheme to commit to the first share and a (S_0, S'_0)-size-robust commitment scheme to commit to the second share.

Thus, the tag-1 commitment is harder w.r.t. circuit-depth, while the tag-0 commitment is harder w.r.t. circuit-size. Leveraging this difference, one can extract from a tag-0 commitment (on the right) without violating the hiding property of a tag-1 commitment (on the left), and vice versa — leading to two-sided non-malleability. More specifically, the committed values in a tag-0 commitment can be extracted in depth D'_0 and size S'_1 by extracting both secret shares from the size- and depth-robust commitments contained in it. Yet, adversaries with such depth and size cannot break the (D_1, D'_1)-depth-robust commitment contained in a tag-1 commitment; thus, the value committed to in the tag-1 commitment remains hidden. On the flip side, the committed value in a tag-1 commitment can be extracted in depth D'_1 and size S'_0, and, similarly, adversaries with such depth and size do not violate the hiding of a tag-0 commitment, due to the fact that the size-robust commitment contained in it is hiding against size-S_1 adversaries.

In summary, combining the two types of commitment schemes gives us depth-and-size robust commitment schemes: A $(D \vee S, D' \wedge S')$-robust commitment is hiding against circuits with depth below D *or* size below S, while extractable by some circuit with depth D' *and* size S', as illustrated in Figure 5.1 (left). In this language, a tag-0 commitment is $(D_0 \vee S_1, D'_0 \wedge S'_1)$-robust while a tag-1 commitment is $(D_1 \vee S_0, D'_1 \wedge S'_0)$-robust. They are mutually non-malleable, because the extractor for one falls into the class of adversaries that the other is hiding against.

The Subtle Issue of Over-Extraction. The above argument captures our key idea, but is overly-simplified. It implicitly assumes that the size- and depth-robust commitments are extractable in the perfect manner: 1) Whenever a commitment is valid, in the sense that there exists an accepting decommitment, the extractor outputs exactly the committed value, otherwise, 2) when the commitment is invalid, it outputs $\perp$. Such strong extractability ensures that to show non-malleability – the right *committed* value is independent of the left committed value, it suffices to show that the right *extracted* value is independent of the left committed value, as argued above. On the other hand, suppose that property 2) does not hold, that is, when the commitment is invalid, the extractor may output arbitrary values – this is known as *over-extraction*. In this case, we can no longer argue the independence of the right committed value based on the independence of the right extracted value. For instance, the extracted value $\tilde{v}$ may not change as the left committed value changes, but the right committed value may have switched from $\tilde{v}$ to $\perp$.

However, our depth-robust commitments from time-lock puzzles do not satisfy such strong extractability.[3] In particular, they are subject to over-extraction. Over-extraction traces back to the fact that only *honestly generated* time-lock puzzles (*i.e.*, in the domain

[3]Our size-robust commitments from injective one-way functions do satisfy such strong extractability.

of the puzzle generation algorithm) are guaranteed to be solvable in certain time. There is no guarantee for ill-generated puzzles, and no efficient procedure for deciding whether a puzzle is honestly generated or not. Observe that this is the case for the time-lock puzzles proposed by Rivest, Shamir, and Wagner [70], since given a puzzle $(s + g^{2^{2^t}} \bmod N,\ N)$ one can extract s using 2^t squaring modular N, but cannot obtain a proof that N is a valid RSA-modulus; this is also the case for the other puzzle construction [76]. As a result, the extractor of our depth-robust commitments that extracts committed values via solving time-lock puzzles, provides no guarantees when commitments are invalid.

This means that our basic 1-bit tag commitment scheme is over-extractable, and the argument above that reasons about the right extracted value fails to establish non-malleability. Nevertheless, the basic scheme does satisfy a variant of non-malleability that we call *non-malleability w.r.t. extraction*, which ensures that the value *extracted* from the right commitment is independent of the left committed value.[4] When a commitment scheme is perfectly-extractable, this new notion is equivalent to standard non-malleability (w.r.t. commitment), but with over-extraction, it becomes incomparable. The issue of over-extraction has appeared in the literature (*e.g.*, [59, 84]), standard methods for dealing with over-extraction requires the committer to additionally prove the validity of the commitment it sends, using for instance zero-knowledge protocols or cut-and-choose techniques. However, these methods take more than 2 rounds of interaction, and do not apply here.

[4]Our notion of non-malleability w.r.t. extraction is inspired from the notion of non-malleability w.r.t. extraction defined by Wee [59]. Furthermore, our notion can be viewed as a special case of the notion of non-malleability w.r.t. replacement defined by Goyal [60], in the sense that the replacer in Goyal's definition is fixed to the over-extractor of the commitment scheme. The benefit of doing so is that we know exactly the complexity of the extractor, which is useful in the rest of the construction.

5.1.2 Full-Fledged Non-Malleable Commitments

At this point, we face two challenges towards constructing full-fledged non-malleable commitments:

- *Challenge 1:* We need to go from non-malleability w.r.t. extraction to non-malleability w.r.t. commitment in 2 rounds. Resolving this challenge would give a 2-round 1-bit tag non-malleable commitment scheme.

- *Challenge 2:* The next challenge is going beyond two tags, towards supporting an exponential 2^n number of tags.

  It is easy to generalize our basic 1-bit tag commitment scheme to handle arbitrary l tags, if there exists a "ladder" of l commitment schemes with increasing levels of depth-robustness, and another "ladder" of l schemes with increasing levels of size-robustness. Concretely, the i'th schemes are respectively (D_i, D'_i)-depth robust and (S_i, S'_i)-size robust, for some

  $$\cdots << D_i << D'_i << \cdots << D_{l-1} << D'_{l-1}$$
  $$<< S_0 << S'_0 << \cdots << S_i << S'_i << \cdots \ .$$

  A commitment with tag $i \in \{0, \cdots, l-1\}$ combines the i'th (D_i, D'_i)-depth-robust scheme and the $(l-i-1)$'th (S_{l-i-1}, S'_{l-i-1})-size-robust scheme to commit to a pair of secret shares of the committed value. This gives a family of l mutually non-malleable commitment schemes, as illustrated in Figure 5.1 (right).

  To directly obtain full-fledged non-malleable commitments, we need an exponential number of levels $l = 2^n$ of depth- and size-robustness, which is, however, impossible from the underlying assumptions. From generic subexponentially hard, say 2^{n^ε} hard, injective one-way functions, we can instantiate at most $O(\log \log n)$ levels of size-robustness. (This is because if we instantiate the i'th size-robust commitment

using the one-way functions with security parameter n_i, the commitment is hiding for adversaries of size $S_i = \mathsf{poly}(2^{n_i^{\varepsilon}})$, and can be broken by adversaries of size $S'_i = \mathsf{poly}(2^{n_i})$. Then, ensuring $S'_{i-1} << S_i$ entails that $n_{i-1}^{1/\varepsilon} < n_i$, and hence $n_0^{1/\varepsilon^i} < n_i$. Since n_i also needs to be polynomial in the global securty parameter n, we have $i = O(\log \log n)$.) Similarly, from subexponentially parallel-time hard time-lock puzzles, we can instantiate $O(\log \log n)$ levels of depth-robustness. Therefore, we need to amplify the number of tags.

We address both challenges using a single transformation.

2-Round Tag Amplification Technique: We present a transformation that converts a 2-round l-tag commitment scheme that is non-malleable w.r.t. extraction, into a 2-round 2^{l-1}-tag commitment scheme that is both non-malleable w.r.t. extraction and w.r.t. commitment. The output protocol can be further transformed to achieve concurrent non-malleability.

With the above transformation, we can now construct full-fledged non-malleable commitment. Start from our basic scheme for a constant $l_0 = O(1)$ number of tags that is non-malleable w.r.t. extraction; apply the tag-amplification technique *iteratively for* $m = O(\log^* n)$ *times* to obtain a scheme for $l_m = 2^n$ tags that is both non-malleable w.r.t. extraction and w.r.t. commitment.

Previously, similar tag-amplification techniques were presented by Lin and Pass in [56] and by Wee in [59]. Our transformation follows the same blueprint, but differ at two important aspects. First, our transformation starts with and preserves non-malleability w.r.t. extraction, which is not considered in their work. Second, their amplification techniques incur a constant additive overhead in the round complexity of the protocol, whereas our transformation keeps the number of rounds invariant at 2. To do so, our amplification step combines ideas from previous works with the new idea of using our

depth-and-size robust commitments to create different 2-round sub-protocols that are mutually "non-malleable" when executed in parallel, in the sense that the security of one sub-protocol remains intact even when the security of another is violated by force.

Our 2-Round Tag-Amplification Technique in More Detail. Similar to [56, 59], the transformation proceeds in two steps:

- First, amplify the security of a scheme from (*one-one*) non-malleability w.r.t. extraction to *one-many* non-malleability w.r.t. extraction and commitment, which, following a proof in [55], implies *concurrent* (or many-many) non-malleability w.r.t. extraction and commitment. (This is why our final protocol can be made concurrently non-malleable.) Here, one-many and concurrent non-malleability w.r.t. extraction or commitment naturally generalize standard non-malleability to the setting where the man-in-the-middle concurrently receives one or many commitments on the left and gives many commitments on the right, and ensures that the joint distribution of the values extracted from or committed in right commitments is independent of the value(s) committed in the left commitments.

- Next, apply the "log-n trick" by Dolev, Dwork and Naor [69] to amplify the number of tags supported from l to 2^{l-1} at the price of losing concurrent security, yielding a protocol that is (*one-one*) non-malleable w.r.t. extraction and commitment.

The main technical challenges lie in the first step. We briefly review the LP [56] approach. At a high-level, they construct one-many non-malleable commitment following the Fiege-Lapidot-Shamir paradigm [103]: The receiver starts by setting up a *hidden* "trapdoor" t. The sender commits to a value v using an arbitrary (potentially malleable) 2-message commitment scheme, followed by committing to 0^n using a (one-one) non-malleable commitment and proving using *many* witness-indistinguishable proofs of

knowledge (WIPOK) that either it knows a decommitment to v *or* it knows a decommitment of the non-malleable commitment to the trapdoor t; the former, called the honest witness, is used by the honest committer, while the latter, called the fake witness, is used for simulation.

The LP protocol arranges all components — the trapdoor-setup, commitment to v, non-malleable commitment (for trapdoor), and every WIPOK — *sequentially*. To compress the protocol into 2 rounds, we run all components in *parallel*, and replace multiple WIPOK proofs with a single 2-round ZAP proof.

Unfortunately, arranging all components in parallel renders the proof of one-many non-malleability in LP invalid. They designed a sequence of hybrids in which different components in the (single) left interaction are gradually switched from being honestly generated to simulated, while maintaining two invariants regarding the (many) right interactions. First, the *soundness* condition states that the man-in-the-middle never commits to a trapdoor in any right interaction. Second, in every right interaction, there is always a WIPOK that can be rewound to extract the value committed to in this interaction, without rewinding the left component being changed; the value extracted must be a valid decommitment since the fake witness does not exist by the soundness invariant — this establishes *strong extractability*. The second invariant is true because the LP protocol contains sufficiently many sequential WIPOKs so that there is always a proof that does not interleave with the left-component being changed. The first invariant, on the other hand, relies not only on the non-malleability of the input commitment scheme, but also on its "robustness" to other components that have a small fixed k number of rounds (such as 2-message commitment and WIPOK). The robustness captures "non-malleability" w.r.t. other protocols, and is achieved by embedding more than k rewinding slots in the input commitment scheme.

In our 2-round protocol, we cannot afford to have many rewinding slots for extraction,

nor for establishing non-malleability between different components. Naturally, we resort to our size-and-depth robust commitments, which can be made mutually non-malleable w.r.t. extraction by setting the appropriate profiles of size-and-depth robustness. We embed a family of 4 such commitments in different components of the protocol, and mimic the LP proof in the following (overly-simplified) manner: In every hybrid, in the left interaction, either a size-and-depth robust commitment or the non-malleable commitment is changed, while on the right, committed values are extracted from a *different* size-and-depth robust commitment or from the non-malleable commitment. (Note that since we now extract values from commitments instead of from WI proofs, we no longer need many WIPOKs and a single ZAP suffices.)

To show that the left interaction remains indistinguishable despite the extraction, we rely on the mutual non-malleability of the size-and-depth robust schemes, but also need the non-malleable commitment and the size-and-depth robust commitments to be mutually non-malleable, which unfortunately does not hold.

Let us explain. It turns out that our basic non-malleable commitment schemes for short tags, and all intermediate schemes produced by the tag-amplification technique are only secure against circuits with *both* bounded-size *and* bounded-depth. In contrast, the depth-and-size robust commitments are secure against circuits with *either* bounded-size *or* bounded-depth. This qualitative difference in adversarial circuit classes prevents them from being mutually non-malleable. To get around this, we instead rely on a "cycle of non-malleability" that consists of the non-malleable commitment scheme and two depth-and-size robust commitment schemes, satisfying that the first scheme is non-malleable to the second, the second non-malleable to the third, and the third to the first. Such a cycle turns out to be sufficient for our proof to go through.

One final technicality is that in order to create the cycle of non-malleability, the hardness of the two size-and-depth robust commitments must be set appropriately according

to that of the non-malleable commitment scheme. Furthermore, the non-malleable commitment scheme produced by the above transformation has weaker security than the input scheme. As a result, to iteratively apply the tag-amplification technique for $O(\log^* n)$ times, we need $O(\log^* n)$ levels of depth- and size-robustness. This can be easily instantiated using subexponentially secure non-interactive commitment schemes and time-lock puzzles as stated in Theorem 2-rnd in Section 1.3. See Section 5.5 for more details on our tag amplification and its security proof.

5.1.3 Extensions

Finally, we briefly mention two extensions. First, our two-round non-malleable commitment scheme can be made non-interactive, at the price of becoming only concurrent non-malleable against attackers that are uniform Turing machines. Second, we show that our two-round non-malleable commitment scheme (and its non-interactive version resp.) in fact satisfies the stronger notion of Chosen Commitment Attack (CCA) security (against uniform Turing machines resp.).

Non-Interactive Non-Malleable Commitments w.r.t. Uniform Attackers. For the first extension, observe that the only step in our construction that requires 2 rounds is the non-malleability strengthening step in the tag-amplification technique. (The basic non-malleable scheme for a constant number of tags are non-interactive and the log-n trick in the tag-amplification technique is round-preserving.) The non-malleability strengthening step produces 2-round protocols, where the first message is from the receiver and consists of i) the first message of a 2-round WI proof, ii) a randomly sampled function from a family of collision resistant hash functions secure against non-uniform attackers, and iii) the first message of the input (one-one) non-malleable commitment scheme if it has 2 rounds. To remove the first message we can simply replace 2-round WI proofs with

non-interactive WI proofs (NIWIs), and fix a single hash function (instead of a family). However, since a single hash function can only be collision resistant to attackers that are uniform Turing machines, the resulting non-interactive commitment scheme is only concurrent non-malleable against uniform adversaries. See Section 5.9 for more details.

CCA-secure Commitments. CCA-security strengthens the notion of concurrent non-malleability in ways similar to how Chosen Ciphertext Attack secure encryption strengthens non-malleable encryption. Roughly speaking, CCA-security requires that no man-in-the-middle attacker can distinguish commitments to different values on the left, even if it has access to a committed-value oracle, which breaks every commitment the attacker sends on the right (except the left commitment), and returns the unique committed value as soon as the right interaction ends. Our 2-round concurrent non-malleable commitments are in fact CCA-secure. To see this, it suffices to argue that the non-malleability strengthening step in the tag-amplification technique produces CCA-secure commitments, as the final 2-round protocol is produced by this procedure. Recall that to show the concurrent non-malleability of the resulting 2-round protocol, we built a sequence of hybrids, where different components in the left commitment are changed one by one, while the right committed values are extracted by breaking different components in right commitments. The indistinguishability of neighboring hybrids follows from the mutual non-malleability of the component being broken on the right, and the component being changed on the left. We observe that this argument can be easily changed to prove CCA security. The only modification to the hybrids is simulating the committed-value oracle for the attacker by sending it the values extracted from the right commitments. The mutual non-malleability of different components still guarantees the indistinguishability of the hybrids, now with committed-value oracles. There are still some subtleties in the proof; see Section 5.8 for more details.

5.1.4 Concurrent and Independent Work

A concurrent and independent, beautiful, work by Khurana and Sahai (KS) [104] also presents a construction of 2-round non-malleable commitments from subexponential hardness assumptions. The results, however, are incomparable, both in terms of assumptions, and also in terms of the achieved results (and use significantly different techniques).

In terms of results, our protocols satisfy *full* concurrent non-malleability, whereas the KS protocol only satisfies "bounded-concurrent" non-malleability—which is a weaker notion of concurrent non-malleability where the number of sessions is *a-priori bounded* by some pre-determined polynomial in the security parameter; in particular, the communication complexity of their protocol grows super linearly with the bound on the number of sessions, and the complexity assumptions they rely on need to be parametrized by it. Additionally, we also present a fully non-interactive protocol, whereas their technique appears to be inherently limited to two-round protocols.

In terms of assumptions, the key difference is that KS does not rely on time-lock puzzles but rather on the existence of certain 2-round secure two-party computation protocols (with super-polynomial-time simulation security); they also claim that such protocols can be constructed based on the subexponential DDH assumption, or the subexponential QR assumption. These assumptions are incomparable to the subexponential repeated squaring assumption, which as we mentioned above is also a very natural computational problem that has been extensively studied over the years. On a qualitative level, it is also a search assumption (and thus our construction of non-malleable commitments can be based on search assumptions), whereas the KS construction (due to the above DDH, or QR, assumption) relies on "decisional assumptions".

5.1.5 A Perspective: Non-Malleability from Hardness in Different Axes

In this work, our foremost idea is deriving non-malleability from hardness in different axes. While our particular instantiation uses commitments hard in the axis of circuit-size (or time) and commitments hard in the axis of circuit-depth (or parallel time), these are many other types of resources one can consider. For instance, the concurrent work by Khurana and Sahai [104] uses commitments extractable in certain time without rewinding, and rewinding does not help extraction (e.g. any non-interative commitments), and commitments extractable using rewinding, and is extremely hard to break without rewinding (they constructed such commitments using special 2-round two-party computation protocols). We can view the hardness axes involved in their work as 1) time for extraction without rewinding, and 2) time for extraction with rewinding. In a follow-up work by Bitansky and Lin [105] on constructing one-message zero-knowlege arguments and non-malleable commitments from keyless multi-collision resistant hash functions and other assumptions, they considered two axes: 1) time for extraction with probability 1 and 2) the probability of successful extraction in polynomial time. More precisely, they build $\mathsf{Com}_1, \mathsf{Com}_2$ such that the values committed using Com_2 can be extracted with probability 1 in time T, while Com_1 remains hiding in time T, whereas the probability that a polynomial-time extractor succeeds in extracting values from Com_2 is much smaller than that from Com_1. In another follow-up work [106] on constructing non-malleable codes against bounded polynomial time tampering, they considered the axis of "BP-time" corresponding to time for extraction by probablistic Turing machine, and the axis of non-deterministic "(ND)-size" corresponding to time for extraction by NP circuits. We believe that there are more hardness axes and considering their synergy may lead to new applications.

5.2 Preliminaries

5.2.1 Basic Notation

We denote the security parameter by n. For $n \in \mathbb{N}$, by $[n]$ we denote the set $\{1, \ldots, n\}$. If v is a binary string then $|v|$ denotes the length of the string and $v[i]$ is the ith bit of v, for $0 \leq i \leq |v| - 1$. We use $||$ as the string concatenation operator. We identify strings $p \in \{0,1\}^t$ with an index in $[2^t]$. For any probability distribution D, $x \leftarrow D$ denotes sampling an element from the distribution D and assigning it to x. However, for a finite set Q, $x \leftarrow Q$ denotes sampling an element from the set Q uniformly and randomly, and assigning it to x. We model algorithms as uniform TMs. We use the abbreviation PPT to denote probabilistic-polynomial time. $\mathcal{P}/\text{poly}$ is the set of all non-uniform polynomial size circuits. We say that a function $\nu : \mathbb{N} \to \mathbb{R}$ is negligible, if for every constant $c > 0$ and for all sufficiently large $n \in \mathbb{N}$ we have $\nu(n) < n^{-c}$. For functions d, S defined over $\mathbb{N}$, we say that $d < S$ (resp. $d \leq S$) if for all sufficiently large $n \in \mathbb{N}$, $d(n) < S(n)$ (resp. $d(n) \leq S(n)$). Furthermore, we say that $d << S$ if for every polynomial poly, $\mathsf{poly}(d) < S$.

5.2.2 Circuit Classes

We define the following circuit classes which are going to be used throughout this work. For the following definitions, consider $n \in \mathbb{N}$ and let d, S and S^* be some non-decreasing functions defined on $\mathbb{N}$ such that $d \leq S << S^*$.

Definition 14 (Depth $\wedge$ size-restricted circuits) *$\mathcal{C}^{\wedge}_{d,S}$ is the set of all non-uniform circuits $C = \{C_n\}_{n \in \mathbb{N}}$ such that there exists a polynomial* poly *such that for all sufficiently*

large $n \in \mathbb{N}$,

$$\mathsf{dep}(C_n) < \mathsf{poly}(d(n))$$
$$\textit{and} \quad \mathsf{size}(C_n) < \mathsf{poly}(S(n)) \ ,$$

where $\mathsf{dep}(C_n)$ *and* $\mathsf{size}(C_n)$ *denote the depth and the size of the circuit* C_n *respectively.*

Throughout this work, we only consider circuits of sub-exponential size. In particular, all such circuits have size significantly lesser than 2^{n^ε} for some $0 < \varepsilon < 1$. For generality, we let S^* to denote some pre-defined upper bound on the size of any circuits considered in this work. Furthermore, when we are only concerned with restricting the depth of the circuits, whose size can be as large as the upperbound $\mathsf{poly}(S^*)$ for any polynomial poly, we simply refer to the circuit class $\mathcal{C}^{\wedge}_{d,S^*}$ as $\mathcal{C}_d$.

Definition 15 (Depth-restricted circuits) $\mathcal{C}_d$ *is the set of all non-uniform circuits* $C = \{C_n\}_{n\in\mathbb{N}}$ *such that there exists a polynomial* poly *such that for all sufficiently large* $n \in \mathbb{N}$,

$$\mathsf{dep}(C_n) < \mathsf{poly}(d(n))$$
$$\textit{and} \quad \mathsf{size}(C_n) < \mathsf{poly}(S^*(n)) \ .$$

Definition 16 (Depth $\vee$ size-restricted circuits) $\mathcal{C}^{\vee}_{d,S}$ *is the set of all non-uniform circuits* $C = \{C_n\}_{n\in\mathbb{N}}$ *such that either* $C \in \mathcal{C}_d$ *or* $C \in \mathcal{C}_S$.

Remark 7 *The classes of circuits* $\mathcal{C}$ *(namely,* $\mathcal{C}_d$,$\mathcal{C}^{\vee}_{d,S}$ *and* $\mathcal{C}^{\wedge}_{d,S}$*) considered in this work are such that* $S \geq d >> n$*, that is, all* d*'s and* S*'s are super-polynomials. For such classes* $\mathcal{C}$*, composing any circuit* $C \in \mathcal{C}$ *with a circuit* $P \in \mathcal{P}/\mathrm{poly}$ *results in a circuit* C' *which is also in the class* $\mathcal{C}$*. Therefore, we say that the circuit class* $\mathcal{C}$ *is closed under composition with* $\mathcal{P}/\mathrm{poly}$*. This fact is going to be important in the rest of this work.*

Below, we define standard cryptographic primitives w.r.t. a general circuit class $\mathcal{C}$, requiring that any adversary in $\mathcal{C}$ has negligible advantage in breaking the security of the primitive. When $\mathcal{C} = \mathcal{P}/\text{poly}$, we say that the primitive is computationally secure and when $\mathcal{C}$ is the set of non-uniform circuits whose size is bounded by 2^{n^ε} for some constant $\varepsilon < 1$, we say that the primitive is subexponentially secure.

5.2.3 Indistinguishability and One-wayness

Definition 17 ($\mathcal{C}$-indistinguishability) *Two distribution ensembles $\{A_n\}_{n\in\mathbb{N}}$ and $\{B_n\}_{n\in\mathbb{N}}$ are said to be $\mathcal{C}$-indistinguishable, if for every non-uniform circuit $D = \{D_n\}_{n\in\mathbb{N}} \in \mathcal{C}$, there exists a negligible function $\nu(\cdot)$ such that for every $n \in \mathbb{N}$:*

$$|\mathsf{Pr}\left[a \leftarrow A_n : D_n(a) = 1\right] - \mathsf{Pr}\left[b \leftarrow B_n : D_n(b) = 1\right]| \leq \nu(n) \ .$$

Definition 18 ($\mathcal{C}$-unpredictability) *Let $X = \{X_n\}_{n\in\mathbb{N}}$ and $Y = \{Y_n\}_{n\in\mathbb{N}}$ be two ensembles of countable sets. Let $D = \{D_n\}_{n\in\mathbb{N}}$ be a distribution ensemble such that for every $n \in \mathbb{N}$, D_n is a distribution over pairs $(x, y) \in X_n \times Y_n$. We say that D is $\mathcal{C}$-unpredictable w.r.t. (X, Y) if for every non-uniform circuit $A = \{A_n\}_{n\in\mathbb{N}} \in \mathcal{C}$ there exists a negligible function $\nu(\cdot)$ such that for every $n \in \mathbb{N}$,*

$$\mathsf{Pr}[(x, y) \xleftarrow{\$} D_n, x' \leftarrow A_n(y) : x = x'] \leq \nu(n) \ .$$

Definition 19 (One-way functions) *A function $f : \{0,1\}^* \to \{0,1\}^*$ is called a $\mathcal{C}^{\wedge}_{S,S}$-secure one-way function (OWF) if the following hold:*

1. *There exists a deterministic polynomial-time algorithm that on input s in the domain of f outputs $f(s)$.*

2. *For every $A = \{A_n\}_{n\in\mathbb{N}} \in \mathcal{C}^{\wedge}_{S,S}$ there exists a negligible function $\nu(\cdot)$ such that for every $n \in \mathbb{N}$,*

$$\mathsf{Pr}\left[s \leftarrow \{0,1\}^n, s' \leftarrow A_n(f(s)) : f(s') = f(s)\right] \leq \nu(S(n)) \ .$$

As a short-hand, we will sometimes refer to $\mathcal{C}^{\wedge}_{S,S}$-secure one-way function as S-secure one-way function. In this work, we will use a one-way function that is injective and is subexponentially secure. That is, we assume the existence of a $\mathcal{C}^{\wedge}_{S,S}$-secure injective one-way function where $S = 2^{n^{\varepsilon}}$ for some $0 < \varepsilon < 1$.

Definition 20 (Hardcore functions) *Let D be a distribution ensemble over pair (X, Y) of ensembles of countable sets. A function $h : X \to \{0,1\}$ is a $\mathcal{C}$-hardcore predicate of D if the following hold:*

1. *There exists a deterministic polynomial-time algorithm that on input $x \in X$ outputs $h(x)$.*
2. *For every $A = \{A_n\}_{n\in\mathbb{N}} \in \mathcal{C}$ there exists a negligible function $\nu(\cdot)$ such that for every $n \in \mathbb{N}$,*

$$\mathsf{Pr}\left[(x,y) \leftarrow D_n, b \leftarrow A_n(y) : b = h(x)\right] \leq \frac{1}{2} + \nu(n) \ .$$

Theorem 12 (Golreich-Levin Hardcore Bit) *Let D be a $\mathcal{C}$-unpredictable distribution ensemble over (X, Y) such that there exists a polynomially bounded function r such that for every $n \in \mathbb{N}$, $X_n \subseteq \{0,1\}^{r(n)}$. Let D' be the following distribution ensemble,*

$$\{((x,z),(y,z)) : (x,y) \leftarrow D_n, z \leftarrow \{0,1\}^{r(n)}\}_{n\in\mathbb{N}} \ .$$

And let $h : X \times \{0,1\}^r \to \{0,1\}$ be the function such that for every $(x,z) \in X \times \{0,1\}^r$, $h((x,z)) = \langle x \cdot z \rangle$. Then, D' is $\mathcal{C}$-unpredictable over $(X \times \{0,1\}^r, Y \times \{0,1\}^r)$ and h is a $\mathcal{C}$-hardcore predicate of D'.

Remark 8 *Goldreich and Levin [107] show that for any adversary $A \in \mathcal{C}$ that breaks the hardcoreness of h w.r.t. D' with probability $1/2 + \varepsilon(n)$ there exists an adversary B that breaks unpredictability of D where*

$$\mathsf{size}(B) \leq \mathsf{poly}(n/\varepsilon^2) \cdot \mathsf{size}(A) \ ; \ \mathsf{dep}(B) \leq \mathsf{poly}(n/\varepsilon^2) \cdot \mathsf{dep}(A) \ .$$

Since, $\varepsilon = 1/p(n)$ for some polynomial p the reduction blows up the size/depth of B over size/depth of A by only a $\mathsf{poly}(n)$ factor. Therefore, if $A \in \mathcal{C}$ then $B \in \mathcal{C}$ which then contradicts the $\mathcal{C}$-unpredictability of D.

5.2.4 Witness Relation, ZAP and NIWI

Definition 21 (Witness Relation) *A witness relation or relation (for short) for a language $L \in \mathcal{NP}$ is a binary relation $\mathcal{R}_L$ that is polynomially bounded, polynomial time recognizable and characterizes L by $L = \{x : \exists w \text{ s.t. } (x, w) \in \mathcal{R}_L\}$.*

We say that w is a witness for the membership of $x \in L$ if $(x, w) \in \mathcal{R}_L$. We will also let $\mathcal{R}_L(x)$ denote the set of witnesses for the membership of $x \in L$; that is, $\mathcal{R}_L(x) = \{w : (x, w) \in \mathcal{R}_L\}$.

ZAPs are two-message public coin witness indistinguishable proofs defined as follows.

Definition 22 (ZAP [69]) *A pair of algorithms $(\mathcal{P}, \mathcal{V})$, where $\mathcal{P}$ is PPT and $\mathcal{V}$ is (deterministic) polytime, is a $\mathcal{C}$-ZAP for an $\mathcal{NP}$ relation $\mathcal{R}_L$ if it satisfies:*

1. *Completeness: There exists a polynomial $l(\cdot)$ such that for every $(x, w) \in \mathcal{R}_L$,*

$$\mathsf{Pr}\left[r \leftarrow \{0,1\}^{l(|x|)}, \pi \leftarrow \mathcal{P}(x, w, r) : \mathcal{V}(x, \pi, r) = 1\right] = 1 \ .$$

2. *Adaptive soundness: There exists a negligible function $\nu(\cdot)$ such that for every malicious (potentially unbounded) prover $\mathcal{P}^*$ and every $n \in \mathbb{N}$,*

$$\mathsf{Pr}\left[r \leftarrow \{0,1\}^{l(n)}, (x, \pi) \leftarrow \mathcal{P}^*(r) : x \in \{0,1\}^n \setminus L \wedge \mathcal{V}(x, \pi, r) = 1\right] \leq \nu(n).$$

3. *$\mathcal{C}$-witness indistinguishability: For any sequence $\{(x_n, w_n^1, w_n^2, r_n)\}_{n\in\mathbb{N}}$ such that for every $n \in \mathbb{N}$, $x_n \in L \cap \{0,1\}^n$, $w_n^1, w_n^2 \in \mathcal{R}_L(x_n)$ and $r_n \in \{0,1\}^{l(n)}$, the following*

ensembles are $\mathcal{C}$-indistinguishable:

$$\{\pi_1 \leftarrow \mathcal{P}(x_n, w_n^1, r_n) : (x_n, w_n^1, w_n^2, \pi_1, r_n)\}_{n\in\mathbb{N}} ,$$
$$\{\pi_2 \leftarrow \mathcal{P}(x_n, w_n^2, r_n) : (x_n, w_n^1, w_n^2, \pi_2, r_n)\}_{n\in\mathbb{N}} .$$

Throughout this work, we will refer to the first message r of ZAP as a_{ZAP} and the second message together with the statement (π, x) as b_{ZAP}.

Dwork and Naor [69] were the first to construct a ZAP from certified trapdoor permutations [108]. They also showed that ZAP for $L \in \mathcal{NP}$ can be based on the weaker assumption of the existence of NIZKs for L.

Theorem 13 *If there exists a $\mathcal{C}$-secure family of certified trapdoor permutations then there exists a $\mathcal{C}$-ZAP.*

Furthermore, Bitansky and Paneth [79] construct ZAP based on the existence of indistinguishability obfuscation (iO) for a certain family of polysize circuits and one-way functions.

NIWIs are non-interactive witness-indistinguishable proofs.

Definition 23 (NIWI [77]) *A pair of algorithms $(\mathcal{P}, \mathcal{V})$ where $\mathcal{P}$ is PPT and $\mathcal{V}$ is (deterministic) polytime, is a $\mathcal{C}$-NIWI for an $\mathcal{NP}$ relation $\mathcal{R}_L$ if it satisfies:*

1. *Completeness: For every $(x, w) \in \mathcal{R}_L$,*

$$\mathsf{Pr}\left[\pi \leftarrow \mathcal{P}(x, w) : \mathcal{V}(x, \pi) = 1\right] = 1 .$$

2. *Soundness: For every $x \notin L$ and $\pi \in \{0,1\}^{\mathsf{poly}(|x|)}$:*

$$\mathsf{Pr}\left[\mathcal{V}(x, \pi) = 1\right] = 0 .$$

3. *$\mathcal{C}$-witness indistinguishability:* *For any sequence $\{(x_n, w_n^1, w_n^2)\}_{n\in\mathbb{N}}$ such that for every $n \in \mathbb{N}$, $x_n \in L \cap \{0,1\}^n$, $w_n^1, w_n^2 \in \mathcal{R}_L(x_n)$, the following ensembles are $\mathcal{C}$-indistinguishable:*

$$\{\pi_1 \leftarrow \mathcal{P}(x_n, w_n^1) : (x_n, w_n^1, w_n^2, \pi_1)\}_{n\in\mathbb{N}} ,$$

$$\{\pi_2 \leftarrow \mathcal{P}(x_n, w_n^2) : (x_n, w_n^1, w_n^2, \pi_2)\}_{n\in\mathbb{N}} .$$

Dwork and Naor [69] showed the existence of a non-uniform non-constructive NIWI which can be based on their ZAP construction by fixing the first message non-uniformly. Building on their work, Barak, Ong and Vadhan [77] de-randomize the ZAP verifier in [69] to give the first NIWI construction. They base their de-randomization technique on the existence of a function in $Dtime(2^{O(n)})$ with non-deterministic circuit complexity $2^{\Omega(n)}$. The ZAP construction from [79] can also be de-randomized under the same assumption. Furthermore, Groth, Ostrovsky and Sahai [78] construct a NIWI based on the decisional linear assumption for bilinear groups.

Theorem 14 *We base the existence of* NIWI *on either of the following assumptions:*

1. *If decisional linear assumption holds for the elliptic curve based bilinear groups in [109] against all circuits in class $\mathcal{C}$ then there exists a $\mathcal{C}$-*NIWI*.*

2. *If $\mathcal{C}$-ZAPs exist and there exists a function in the class $Dtime(2^{O(n)})$ with non-deterministic circuit complexity $2^{\Omega(n)}$ then there exists a $\mathcal{C}$-*NIWI*.*

5.2.5 Commitment Schemes

Definition 24 (Commitment scheme) *A commitment scheme $\langle C, R\rangle$ consists of a pair of interactive* PPT *TMs C and R with the following properties:*

1. *The commitment scheme has two stages: a commit stage and a reveal stage. In both stages, C and R receive as common inputs 1^n and $1^{\alpha(n)}$ and C additionally*

receives a private input $v \in \{0,1\}^{\alpha(n)}$ *where* $n \in \mathbb{N}$ *is the security parameter and* $\alpha(\cdot)$ *is some polynomially bounded function.*[5]

2. *The commit stage results in a joint output* c*, called the commitment, a private output for* C*,* d*, called the decommitment string. Without loss of generality,* c *can be the full transcript of the interaction between* C *and* R *including the common input* 1^n *and* $1^{\alpha(n)}$*. Let* $n_c = n_c(n, \alpha(n))$ *denote the maximal length of the commitment generated by* $\langle C, R\rangle$ *while committing to an* $\alpha(n)$*-bit value on security parameter* n*.*

3. *In the reveal stage, committer* C *sends the pair* (v, d) *to the receiver* R*, and* R *decides to accept or reject the decommitment* (v, d) *deterministically according to an efficiently computable function* Open*; that is,* R *accepts iff* $\mathsf{Open}(c, v, d) = 1$*.*

4. *If* C *and* R *do not deviate from the protocol, then* R *should accept with probability 1 in the reveal stage.*

Furthermore, we say that a commitment c is valid, if there exists a string $v \in \{0,1\}^{\alpha(n)}$ and a decommitment string d such that $\mathsf{Open}(c, v, d) = 1$.

Next we define the binding and hiding property of a commitment scheme.

Definition 25 (Statistical binding) *A commitment scheme* $\langle C, R\rangle$ *is statistically binding if for every polynomially bounded function* $\alpha(\cdot)$ *and for any committer* C^* *possibly unbounded, there exists a negligible function* $\nu(\cdot)$ *such that* C^* *succeeds in the following game with probability at most* $\nu(n)$*:*

On security parameter 1^n*,* C^* *first interacts with* R *in the commit stage to produce a commitment* c*. Then* C^* *outputs two decommitments* (v_0, d_0) *and* (v_1, d_1)*, and succeeds if* $v_0, v_1 \in \{0,1\}^{\alpha(n)}$*,* $v_0 \neq v_1$ *and* R *accepts both as decommitments of* c*.*

[5]For notational convenience we will usually drop the length of the value v being committer, that is, $1^{\alpha(n)}$ from the common input.

Furthermore, a commitment scheme is perfectly binding if the probability that C^* succeeds in the above game is 0.

We define the value of any commitment through a function val, that takes as input an arbitrary commitment c and outputs v if c is valid and there exists exactly one value $v \in \{0,1\}^\alpha$ such that $\mathsf{Open}(c, v, d) = 1$ for some d, otherwise it outputs $\perp$. Note that such a function val may not be efficiently computable.

Definition 26 ($\mathcal{C}$-hiding) *A commitment scheme $\langle C, R\rangle$ is $\mathcal{C}$-hiding if for every polynomially bounded function $\alpha(\cdot)$ and for every non-uniform circuit $A = \{A_n\}_{n\in\mathbb{N}} \in \mathcal{C}$ there exists a negligible function $\nu(\cdot)$ such that A succeeds in the following game with probability at most $\frac{1}{2} + \nu(n)$:*

For security parameter 1^n, A_n outputs a pair of values $v_0, v_1 \in \{0,1\}^{\alpha(n)}$. C on input v_b, where b is a randomly chosen bit, interacts with A_n to produce a commitment of v_b. A_n outputs a bit b' and wins the game if $b' = b$.

Additionally, we consider commitment schemes that are "tag-based".

Definition 27 (Tag-based commitment scheme) *A commitment scheme $\langle C, R\rangle$ is a tag-based scheme with $t(n)$-bit identities if, in addition to the security parameter 1^n, the committer and receiver also receive a "tag" – a.k.a. identity– $\mathsf{id} \in \{0,1\}^{t(n)}$ as common input.*

When the length $t(n)$ of identities is n, we refer to $\langle C, R\rangle$ as a tag-based commitment scheme. We say that a tag-based scheme with $t(n)$-bit identities is perfectly binding (resp., $\mathcal{C}$-hiding) if binding (resp., $\mathcal{C}$-hiding) holds for every $\mathsf{id} \in \{0,1\}^{t(n)}$.

Definition 28 (Over-extractable commitment scheme) *A perfectly binding commitment scheme $\langle C, R\rangle$ is over-extractable w.r.t. extractor $o\mathcal{E} = \{o\mathcal{E}_n\}_{n\in\mathbb{N}}$ if for every polynomially bounded $\alpha(\cdot)$ and any $n \in \mathbb{N}$,*

$$\mathsf{Pr}\left[v' \leftarrow o\mathcal{E}_n(c) : c \text{ is valid } \wedge \mathsf{val}(c) \neq v'\right] = 0 \ , \tag{5.1}$$

where $n_c = n_c(n, \alpha(n))$ is the maximal length of the commitment generated by $\langle C, R\rangle$ with security parameter n and committing to $\alpha(n)$-bit values. Furthermore, we say $\langle C, R\rangle$ is (d, S)-over-extractable if the extractor $o\mathcal{E}$ belongs to the circuit class $\mathcal{C}^{\wedge}_{d,S}$.

Remark 9 *Note that the extractor $o\mathcal{E}$ must successfully (with probability 1) extract the correct value for any valid commitment (i.e., for which there exists a decommitment), even if the valid commitment is generated by a malicious committer.*

Remark 10 *In general, extractors $o\mathcal{E} = \{o\mathcal{E}_n\}_{n\in\mathbb{N}}$ (as in Definition 28) can be randomized and one can relax Equation 5.1 allowing extractors to fail with some negligible probability. As all extractors considered in this work are deterministic, we choose to state the stronger definition. We also note that our notion of (over)-extraction, commitment scheme differs from the notion of extractable commitments [59] where the extractors can additionally interact with a malicious committer to extract the value of the commitment.*

In the rest of the paper whenever we say a commitment scheme, we mean a perfectly binding commitment scheme.

The man-in-the-middle (MIM) execution Let $\langle C, R\rangle$ be a tag-based commitment scheme. Consider a non-uniform circuit family $A = \{A_n\}_{n\in\mathbb{N}}$. For security parameter n and challenge bit $b \in \{0, 1\}$ we refer to $\mathsf{MIM}^{A}_{\langle C,R\rangle}(1^n, b)$ as the man-in-the-middle execution where A_n participates in m-left and m-right concurrent interactions committing to values of length α.[6] We allow A_n complete control over scheduling of messages in all

[6]In standard definitions of non-malleability [51, 55], the man-in-the-middle adversary is also given some auxiliary information z. In this work, we consider non-malleability against non-uniform circuits, which can be thought of as having z hard-wired in them. This is why we ignore z in our definitions.

interactions. For every left interaction $i \in [m]$, A_n adaptively chooses a pair of values $(v_i^0, v_i^1) \in \{0,1\}^\alpha$ and an identity id_i at the start of this interaction, interacts with C to receive a commitment to the value v_i^b using the identity id_i. In right interactions A_n interacts with R attempting to commit to related values $\tilde{v}_1, \ldots, \tilde{v}_m$, using identities $\tilde{\mathsf{id}}_1, \tilde{\mathsf{id}}_2, \ldots, \tilde{\mathsf{id}}_m$ of its choice. We define the values $\tilde{v}_i$ committed on the right as $\tilde{v}_i = \mathsf{val}(\tilde{c}_i)$ where $\tilde{c}_i$ is the commitment in the ith right interaction. Recall that $\mathsf{val}(c) = \perp$, if c is not valid or that it can be opened to more than one value. Otherwise, $\mathsf{val}(c)$ equals the unique value v it can be opened to. Furthermore, if for any right interaction i, $\tilde{\mathsf{id}}_i = \mathsf{id}_j$ for some j, we set $\tilde{v}_i = \perp$.

We define two different flavours of non-malleability. First we recall the standard notion of non-malleability – a.k.a non-malleability w.r.t. commitment, for (tag-based) commitment schemes. Then, we introduce a new notion called non-malleability w.r.t. extraction for over-extractable commitment schemes.

Non-malleability w.r.t. commitment. Consider a MIM execution with A. For security parameter $n \in \mathbb{N}$ and bit $b \in \{0,1\}$, let $\mathsf{mim}^A_{\langle C,R\rangle}(1^n, b)$ denote the random variable that describes the values $\tilde{v}_1, \ldots, \tilde{v}_m$ that A commits to on the right and the view of A in $\mathsf{MIM}^A_{\langle C,R\rangle}(1^n, b)$ where view consists of the set of all messages A sends/receives in the MIM execution.

Definition 29 (Non-malleability) *A tag-based commitment scheme $\langle C, R\rangle$ is said to be concurrent $\mathcal{C}$-non-malleable if for every circuit family $A = \{A_n\}_{n\in\mathbb{N}} \in \mathcal{C}$ participating in $m = \mathsf{poly}(n)$ concurrent interactions, receiving/sending commitments to $\alpha = \mathsf{poly}(n)$-bit values, the following ensembles are computationally indistinguishable:*

$$\left\{\mathsf{mim}^A_{\langle C,R\rangle}(1^n, 0)\right\}_{n\in\mathbb{N}} \;;\; \left\{\mathsf{mim}^A_{\langle C,R\rangle}(1^n, 1)\right\}_{n\in\mathbb{N}} .$$

Non-malleability w.r.t. extraction. Let $\langle C, R\rangle$ be a tag-based commitment scheme which is over-extractable w.r.t. extractor $o\mathcal{E}$. We say that $\langle C, R\rangle$ is non-malleable w.r.t. extraction if the distributions of the random variable emim defined below are indistinguishable. Recall that mim describes the view of A and the values $\tilde{v}_i$ that A commits to on the right. However, the random variable $\mathsf{emim}^{A}_{\langle C,R\rangle,o\mathcal{E}}(1^n, b)$[7], instead, describes the view of A and the values $\tilde{v}_i{}'$ which are obtained by running the extractor $o\mathcal{E}$ on input $\tilde{c}_i$ (the ith right commitment); that is, $\tilde{v}_i{}' \leftarrow o\mathcal{E}_n(\tilde{c}_i)$. Note that, if for any right interaction i, $\tilde{\mathsf{id}}_i = \mathsf{id}_j$, for some j, then we set $\tilde{v}_i{}' = \perp$.

Definition 30 (Non-malleability w.r.t. extraction) *A tag-based commitment scheme $\langle C, R\rangle$ is said to be concurrent $\mathcal{C}$-non-malleable w.r.t. extraction by $o\mathcal{E}$ if the following hold:*

1. *$\langle C, R\rangle$ is over-extractable by $o\mathcal{E}$.*
2. *For every circuit $A = \{A_n\}_{n\in\mathbb{N}} \in \mathcal{C}$ participating in $m = \mathsf{poly}(n)$ concurrent interactions receiving/sending commitments to $\alpha = \mathsf{poly}(n)$-bit values, the following ensembles are computationally indistinguishable:*

$$\left\{\mathsf{emim}^{A}_{\langle C,R\rangle,o\mathcal{E}}(1^n, 0)\right\}_{n\in\mathbb{N}} \;;\; \left\{\mathsf{emim}^{A}_{\langle C,R\rangle,o\mathcal{E}}(1^n, 1)\right\}_{n\in\mathbb{N}} .$$

At first glance, it may seem that the new notion — non-malleability w.r.t. extraction, is no more interesting than the standard notion of non-malleability (w.r.t. commitment). After all, an extractor that agrees with the function val establishes that the two notions are equivalent. Most constructions of non-malleable commitment schemes in the literature, in fact, establish non-malleability by building such an extractor in their security

[7]Note that in general the random variable emim should be parametrized by the extractor $o\mathcal{E}$. But in rest of this work we will drop it from the subscript for notational convenience as the underlying extractor will be clear from the context

proofs. In this work, however, we consider extractors that may not always agree with val and have some *over-extraction*.

Relationship between Non-malleability w.r.t. Commitment and Extraction Over-extractability guarantees that for valid commitments, the extractor extracts out the committed value. However, given an invalid commitment, the value extracted by the extractor can be arbitrary. This inept behaviour of the extractor, on invalid commitments, is what makes the two notions incomparable (in general). For instance, there might exist an adversary A which depending on the value committed on the left may choose to send invalid transcripts on the right with different probabilities. Such an A certainly breaks the non-malleability of the scheme (w.r.t commitment) but depending on the extractor, A may not violate non-malleability w.r.t. extraction because the extracted values may still be indistinguishable. Furthermore, there might exist an adversary that irrespective of the value on the left always sends invalid commitments on the right. Such an A does not break the non-malleability w.r.t. commitment. But A may violate non-malleability w.r.t. extraction by establishing a co-relation between the value committed on the left and the value that will be over-extracted by the extractor on the right. Hence, the two notions are incomparable. However, if one sets up the decommitment condition (which defines the random variable mim) appropriately then we show that it is possible to base non-malleability w.r.t. commitment on non-malleability w.r.t. extraction. We believe this reduction as one of the main contributions of this work.

We also consider relaxed versions of both non-malleability and non-malleability w.r.t. extraction: one-one, one-many and many-one secure commitment schemes. In one-one (a.k.a. standalone), we consider an adversary A that participates in one left and one right interaction; in one-many A participates in one left and many right; and in many-one, A participates in many left and one right interaction.

Relationship between Non-malleability and Hiding We note that any commitment scheme that is $\mathcal{C}$-non-malleable w.r.t. extraction (by extractor $o\mathcal{E}$) is also $\mathcal{C}$-hiding. This is because any adversary $A \in \mathcal{C}$ that breaks hiding (say w.r.t. $v_0, v_1 \in \{0,1\}^\alpha$) can send valid commitments to b^α on the right when receiving a commitment to v_b on the left. Then, due to the over-extraction of $o\mathcal{E}$, A also breaks non-malleability w.r.t. extraction. In fact, this holds irrespective of the complexity of the extractor $o\mathcal{E}$ and also holds for the extractor that computes the function $\mathsf{val}(c)$ – the value of the commitment c.

Theorem 15 *Let $\langle C, R\rangle$ be a commitment scheme and $\mathcal{C}$ be a class of circuits that is closed under composition with $\mathcal{P}/\text{poly}$.*

1. *If $\langle C, R\rangle$ is one-one $\mathcal{C}$-non-malleable w.r.t. commitment then it is $\mathcal{C}$-hiding.*
2. *If $\langle C, R\rangle$ is one-one $\mathcal{C}$-non-malleable w.r.t. extractor $o\mathcal{E}$ then it is $\mathcal{C}$-hiding (irrespective of the complexity of the extractor $o\mathcal{E}$).*

5.2.6 Time-Lock Puzzles

Definition 31 (Time-lock puzzles [76]) *A (T, B)-time-lock (TL) puzzle is a tuple $(\mathsf{Gen}, \mathsf{Sol})$ satisfying the following requirements:*

1. *Syntax:*
   - *$Z \leftarrow \mathsf{Gen}(1^n, 1^t, s)$ is a probabilistic algorithm that takes as input a security parameter n, a solution $s \in \{0,1\}^n$ and a difficulty parameter t and outputs a puzzle Z.*
   - *$s \leftarrow \mathsf{Sol}(Z)$ is a deterministic algorithm that takes as input a puzzle Z and outputs a solution s.*

2. *Completeness:* *For every security parameter n, difficulty parameter t, solution $s \in \{0,1\}^n$ and puzzle Z in the support of $\mathsf{Gen}(1^n, 1^t, s)$, $\mathsf{Sol}(Z)$ outputs s.*

3. *Efficiency:*

   - *$Z \leftarrow \mathsf{Gen}(1^n, 1^t, s)$ is a poly-time algorithm, that is, it runs in time $\mathsf{poly}(t, n)$.*

   - *$s \leftarrow \mathsf{Sol}(Z)$ runs in time $\mathsf{poly}(2^t)$ for Z in the support of $\mathsf{Gen}(1^n, 1^t, \cdot)$.*

4. *(T, B)-hardness:* *$(\mathsf{Gen}, \mathsf{Sol})$ is a (T, B)-hard TL puzzle if for every $t(n) \in \omega(\log n) \cap n^{O(1)}$ and every adversary $A = \{A_n\}_{n \in \mathbb{N}}$ where,*

$$\mathsf{dep}(A_n) \leq T(t) \ ; \ \mathsf{size}(A_n) \leq B(n) \ ,$$

there exists a negligible function ν, such that for every $n \in \mathbb{N}$,

$$\mathsf{Pr}\left[s \leftarrow \{0,1\}^n; \ Z \leftarrow \mathsf{Gen}(1^n, 1^{t(n)}, s); \ s' \leftarrow A_n(Z) : s' = s\right] \leq \nu(n) \ .$$

The first candidate construction of TL puzzles was proposed by Rivest, Shamir and Wagner [70] and is based on the "inherently sequential" nature of exponentiation modulo an RSA integer. Twenty years after their proposal, there still does not exist a (parallelizable) strategy that can solve the puzzle (of difficulty parameter t) in parallel-time $T(t)$ which is significantly less than 2^t. Apart from the variants of RSW puzzles [75, 110], the only other construction of TL puzzles was given by Bitansky et al. [76] based on succinct randomized encodings for Turing machines (which in turn can be built from indistinguishability obfuscation and one-way functions) and the existence of non-parallelizing languages. These previous works have considered puzzles with strong parameters, that is, puzzles that are parallel-time hard for exponential $T = 2^{\delta t}$ ([76]) and even strongly exponential $T = \delta 2^t$ ([75, 110]).

However, for our task of constructing 2-round non-malleable commitments, much weaker TL puzzles are sufficient, that is, puzzles that remain hard for only subexponential

$T = 2^{t^\delta}$ parallel-time. More precisely, we need a $(T(t) = 2^{t^\delta}, B(n) = 2^{n^\varepsilon})$-TL puzzle for some $0 < \varepsilon, \delta < 1$. We present the RSW TL puzzles $\mathsf{RSW} = (\mathsf{Gen}, \mathsf{Sol})$ as a candidate.

- Algorithm $\mathsf{Gen}(1^n, 1^t, s)$:

  1. Select an n-bit RSA modulous $N = pq$.
  2. Compute the mask $y = g^{2^{2^t}} \mod N$ for some element $g \in \mathbb{Z}_N^*$. Note that since the factorization of N is known, Gen can first compute the exponent $e = 2^{2^t} \mod \phi(N)$ and then efficiently compute the mask $y = g^e \mod N$.
  3. Mask the solution s with y, that is, $z = (s + y) \mod N$.
  4. Return the tuple $Z = (z, N)$ as the puzzle.

- Solver $\mathsf{Sol}(Z = (z, N))$:

  1. By 2^t repeated squarings, compute $y = g^{2^{2^t}} \mod N$.
  2. Output $(z - y) \mod N$ as the solution.

As discussed in [70], the element g above can either be a fixed element such as 2, or sampled at random.

Next, we discuss that $\mathsf{RSW} = (\mathsf{Gen}, \mathsf{Sol})$ is a TL puzzle in the sense of Definition 31. It is easy to see that for security parameter n and difficulty parameter t, Gen runs in time $\mathsf{poly}(t, n)$ and Sol runs in time $\mathsf{poly}(2^t)$. Futhermore, we base the (T, B)-hardness of the RSW puzzle on the subexponential repeated squaring assumption as stated in Assumption **??**. Informally, it says that for some subexponential functions T and B, and any function $t \in \omega(\log n) \cap n^{O(1)}$, $B(n)$-sized adversaries with depth $T(t)$ cannot compute $g^{2^{2^t}} \mod N$. We define the assumption more formally below.

Assumption 1 (Subexponential Repeated Squaring Assumption) *There exists subexponential functions T, B such that for every function $t(\cdot) \in \omega(\log n) \cap n^{O(1)}$, the following holds: For every adversary $A = \{A_n\}_{n\in\mathbb{N}}$ such that*

$$\mathsf{dep}(A_n) \leq T(t(n));\ \mathsf{size}(A_n) \leq B(n)\ ,$$

there exists a negligible function μ such that for every $n \in \mathbb{N}$, [8]

$$\mathsf{Pr}\left[N \leftarrow \mathsf{RSA}(n);\ g \leftarrow Z_N^*;\ y \leftarrow A_n(g, N) : y = g^{2^{2^t}} \bmod N\right] \leq \mu(n)\ ,$$

where $\mathsf{RSA}(n)$ is the set of all n-bit RSA moduli.

Then, it is easy to see that if the subexponential repeated squaring assumption holds, then the RSW puzzle as defined above is a (T, B)-hard TL puzzle for some subexponential functions T and B.

Lemma 32 *If the subexponential repeated squaring assumption holds, then there exists subexponential functions T and B, such that, $\mathsf{RSW} = (\mathsf{Gen}, \mathsf{Sol})$ is a (T, B)-hard TL puzzle.*

5.2.7 Collision-resistant Hash Functions

Definition 32 *A family of $\mathcal{C}^{\wedge}_{S,S}$-collision-resistant hash functions (CRH) $\mathcal{H} = \{H_n\}_{n\in\mathbb{N}}$ is a function family ensemble such that for every $n \in \mathbb{N}$, $H_n = \{h : \{0,1\}^{m(n)} \to \{0,1\}^n\}$ such that $n < m(n)$ satisfying,*

1. *Efficient Sampling: There exists a poly-time TM S such that for every $n \in \mathbb{N}$, $S(1^n)$ outputs a uniform element of H_n.*

2. *Efficient Computation: There exists a poly-time TM M such that for every $n \in \mathbb{N}$, $h \in H_n$ and $x \in \{0,1\}^{m(n)}$, $M(h, x) = h(x)$.*

[8] g can also be fixed appropriately instead of sampling it randomly.

3. *S(n)-Collision-resistance: For every non-uniform circuit* $A = \{A_n\}_{n\in\mathbb{N}} \in \mathcal{C}^{\wedge}_{S,S}$ *there exists a negligible function* ν *such that for every* $n \in \mathbb{N}$,

$$\mathsf{Pr}\left[h \leftarrow H_n, (x_1, x_2) \leftarrow A(h) : x_1 \neq x_2 \wedge h(x_1) = h(x_2)\right] \leq \nu(S(n)) \ . \tag{5.2}$$

We will sometimes refer to $\mathcal{C}^{\wedge}_{S,S}$*-collision resistant hash family as* S*-collision resistant hash family. Moreover, a family of* uniform *collision resistant hash function (CRH) is as defined above, except that i) the family* H_n *only consists of a single function* h_n*, and ii)* $S(n)$*-collision resistence only holds against attackers that are* $\mathsf{poly}(S(n))$*-time* uniform *Turing machines. We denote such a family as* $\{h_n\}_{n\in\mathbb{N}}$.

In this work, we will use subexp-secure, uniform or non-uniform, collision-resistant hash functions. For $n \in \mathbb{N}$ and any $h \in H_n$, a collision can be found by a uniform Turing machine in time $2^{n/2}$ with high probability and in time $\mathsf{poly}(n) \cdot 2^n$ with probability 1. Furthermore, for some $0 < \varepsilon < 1/2$, we require that it be hard for any $\mathsf{poly}(2^{n^\varepsilon})$-sized circuit (or a $\mathsf{poly}(2^{n^\varepsilon})$-time uniform Turing machine) to find collisons for a randomly chosen hash function $h \leftarrow H_n$ (or for h_n in the uniform case) for $0 < \varepsilon < 1/2$.

5.3 Basic Commitment Schemes

In this section we construct three basic over-extractable commitment schemes, each one of them enjoys hiding against different circuit classes. Firstly, we construct a depth-robust commitment scheme which is (S', S')-over-extractable and hiding against any circuit whose depth is sufficiently smaller than S'. Next, we construct a size-robust commitment scheme which is hiding against any circuit whose size is at most $\mathsf{poly}(S)$ but there exists an extractor of polynomial depth and size larger than S. Finally, we construct a commitment scheme which is hiding against both depth-restricted and size-restricted circuits.

5.3.1 Depth-robust Over-extractable Commitment Scheme from a TL-puzzle

For some subexponential functions T and B, assume the existence of a (T, B)-TL puzzle $(\mathsf{Gen}, \mathsf{Sol})$. For any difficulty parameter $t(n) \in \omega(\log n) \cap n^{O(1)}$, these puzzles are solvable in time $\mathsf{poly}(2^t)$ but hard for $B(n)$-sized circuits having depth at most $\mathsf{poly}(T(t))$.[9] Furthermore, consider a difficulty parameter $t(n)$ that admits the following hierarchy of non-decreasing functions, $n << d = T(t) << S' = 2^t << S^* << B$. Using the (T, B)-TL puzzles, we construct a commitment scheme which is over-extractable in time $\mathsf{poly}(S')$ and is hiding against any circuit in $\mathcal{C}_d$ (hence the name *depth-robust* commitment scheme). We refer to the commitment scheme as $(\mathsf{ECom}_d, \mathsf{EOpen}_d)$ which is described below. [10]

On input a security parameter 1^n, the honest committer C runs the algorithm ECom_d described below to commit to a value $v \in \{0,1\}^\alpha$. After the commit stage, the honest receiver R decides whether to accept the commitment by running the function EOpen_d as described in the reveal stage.

- Commit stage - Algorithm ECom_d:

1. On input security parameter 1^n and value $v \in \{0,1\}^\alpha$, for every $0 \leq i \leq \alpha - 1$, ECom_d samples random strings $s_i, r_i \in \{0,1\}^n$ and computes the commitment c_i to $v[i]$, the ith bit of v, as follows,

$$c_i = (Z_i = \mathsf{Gen}(1^n, 1^{t(n)}, s_i \,; r),\ r_i,\ \langle r_i \cdot s_i \rangle \oplus v[i]) \ ,$$

[9] The definition of TL puzzles presented in Definition 31 defines hardness against circuits with depth at most T but for ease of description we assume hardness for $\mathsf{poly}(T)$ depth. This is without loss of generality for subexponential $T = 2^{t^{\delta'}}$, that is, hardness against $2^{t^{\delta'}}$ implies hardness against $\mathsf{poly}(2^{t^\delta})$ for any $\delta < \delta' < 1$.

[10] From now on, for notational convenience, we represent a non-interactive commitment scheme by the tuple of commit and open algorithms; that is $(\mathsf{ECom}, \mathsf{EOpen})$, instead of a pair of interactive TMs C and R.

where r is the random tape used by Gen and t is the difficulty parameter such that $d = T(t)$.

2. ECom_d sets the vector $c = \{c_i\}_{0 \le i \le \alpha-1}$ as the commitment and sets $(v, \{s_i\}_{0 \le i \le \alpha-1}, r)$ as the decommitment.

- Reveal stage - Function EOpen_d:

  On input commitment $c = \{c_i\}_{0 \le i \le \alpha-1}$ and decommitment $(v, \{s_i\}_{0 \le i \le \alpha-1}, r)$, EOpen_d returns 1 if $c_i = (\mathsf{Gen}(1^n, 1^t, s_i\,; r),\ r_i,\ \langle r_i \cdot s_i \rangle \oplus v[i])$ for every $0 \le i \le \alpha - 1$. Otherwise, outputs 0.

Furthermore, the extractor $o\mathcal{E}_d$ of the scheme proceeds as follows:

- Extraction - Extractor $o\mathcal{E}_d$:

  On input any commitment $c = \{c_i = (Z_i, r_i, z_i)\}_{0 \le i \le \alpha-1}$, the extractor $o\mathcal{E}_d$ computes the solution s_i of Z_i by running $\mathsf{Sol}(Z_i)$. Then, $o\mathcal{E}_d$ extracts bit $v[i]$ committed in c_i by computing $v[i] = z_i \oplus \langle r_i \cdot s_i \rangle$. $o\mathcal{E}_d$ returns the string $v[0]||\ldots||v[\alpha-1]$ as its output.

Theorem 16 *Assume the existence of (T, B)-TL puzzles $(\mathsf{Gen}, \mathsf{Sol})$ for some subpexponential functions T and B. Then, for any $t(n) \in \omega(\log n) \cap n^{O(1)}$ and any non-decreasing function S^* satisfying $n << d = T(t) << S' = 2^t << S^* << B$, $(\mathsf{ECom}_d, \mathsf{EOpen}_d)$ is a non-interactive, perfectly binding, $\mathcal{C}_d$-hiding, (S', S')-over-extractable commitment scheme w.r.t. extractor $o\mathcal{E}_d$.*

Proof:

We discuss each of the properties in the following:

- Efficiency: For any $n \in \mathbb{N}$, difficulty parameter $t = t(n)$ and length $\alpha = \alpha(n)$ which are upper-bounded by some polynomial, and $0 \le i \le \alpha - 1$, ECom_d runs Gen to

sample puzzles Z_i's and rest of computation (i.e., sampling n-bit strings, computing inner-product) takes $\mathsf{poly}(n)$ time. In fact for difficulty parameter $t(n)$, Gen runs in time $\mathsf{poly}(t, n)$ which is upper-bounded by some $\mathsf{poly}(n)$ as t is upper-bounded by a polynomial. Hence, ECom_d runs in time $\mathsf{poly}(n)$ for each $0 \leq i \leq \alpha - 1$. Furthermore, since α is also upper-bounded by a polynomial, ECom_d is efficient.

- Perfect binding: Note that TL-puzzles are injective, that is, any (even arbitrarily generated) Z belongs to the support of $\mathsf{Gen}(1^n, 1^t, s)$ for at most one solution $s \in \{0,1\}^n$. Assume towards a contradiction that there exists a Z that belongs to the support of both $\mathsf{Gen}(1^n, 1^t, s_0)$ and $\mathsf{Gen}(1^n, 1^t, s_1)$ for some $s_0 \neq s_1$. Let $s = \mathsf{Sol}(Z)$ be the output of the deterministic algorithm Sol on input Z. If $s \neq s_0$ then this contradicts the completeness of Sol w.r.t. puzzles in the support of $\mathsf{Gen}(1^n, 1^t, s_0)$. If $s = s_0$ then it contradicts the completeness of Sol w.r.t. puzzles in the support of $\mathsf{Gen}(1^n, 1^t, s_1)$. Therefore, for any puzzle Z there exists at most one solution s and in the case when a solution s exists we know $s = \mathsf{Sol}(Z)$.[11]

  Now, let $c = \{c_i = (Z_i, r_i, z_i)\}_{0 \leq i \leq \alpha(n)-1}$ be any commitment. From the above observation, we know that every Z_i falls in the support of at most one s_i. Therefore, for c there exists at most one sequence $(v, \{s_i\}_{0 \leq i \leq \alpha(n)-1})$ for which EOpen_d returns 1. This implies perfect binding of $(\mathsf{ECom}_d, \mathsf{EOpen}_d)$.

- Over-extractable: First, the extractor $o\mathcal{E}_d$ belongs to the class $\mathcal{C}^{\wedge}_{S',S'}$ since Sol runs in time $\mathsf{poly}(S') = \mathsf{poly}(2^t)$ and the rest of the computation takes $\mathsf{poly}(n)$ time.

  Note that for any valid commitment $c = \{c_i = (Z_i, r_i, z_i)\}_{0 \leq i \leq \alpha(n)-1}$, Z_i's are honestly generated puzzles and furthermore each Z_i belongs to the support of $\mathsf{Gen}(1^n, 1^t, s_i)$ for exactly one s_i. These s_i's along with the r_i's (from c) uniquely define $\mathsf{val}(c)$, the value

[11] It can be possible that some Z does not belong to the support of any $\mathsf{Gen}(1^n, 1^t, s)$ for any s, in which case we say that Z has no solution.

of the commitment. Moreover given (s_i, r_i)'s $\mathsf{val}(c)$ is efficiently computable.

Then on any valid commitment c as input, the extractor $o\mathcal{E}_d$ first runs Sol on each of the Z_i's. Due to the perfect correctness of Sol, the extractor $o\mathcal{E}_d$ always extracts the corresponding s_i's and hence also the correct unique committed value, $\mathsf{val}(c)$. Therefore, $(\mathsf{ECom}_d, \mathsf{EOpen}_d)$ is (S', S')-over-extractable.

- Hiding: Let $t(n) = \omega(\log n)$ be some polynomially bounded difficulty parameter. Then by the definition of (T, B)-hardness of the TL puzzle we know that any adversary $A = \{A_n\}_{n\in\mathbb{N}}$, with $\mathsf{dep}(A_n) \leq \mathsf{poly}(T(t))$ and $\mathsf{size}(A_n) \leq \mathsf{poly}(S^*) < B$, solves the puzzle $Z \leftarrow \mathsf{Gen}(1^n, 1^t, s)$ only with negligible probability for some randomly chosen s. Therefore, the distribution

$$\{s \leftarrow \{0,1\}^n,\ Z \leftarrow \mathsf{Gen}(1^n, 1^t, s)\ :\ (s, Z)\}\ , \tag{5.3}$$

is unpredictable for any such adversary A. In our construction of $(\mathsf{ECom}_d, \mathsf{EOpen}_d)$, we sample the TL puzzles with difficulty t such that $T(t) = d$. Therefore, the above distribution is $\mathcal{C}_d$-unpredictable. Then, by a standard argument (see Theorem 12) about the hardcoreness of the Goldreich Levin bit [107] extracted from an $\mathcal{C}_d$-unpredictable distribution, we can conclude that the function that on input (s, r) outputs $\langle s \cdot r\rangle$ is hardcore for circuits in the class $\mathcal{C}_d$.[12] This then implies that $(\mathsf{ECom}_d, \mathsf{EOpen}_d)$ is $\mathcal{C}_d$-hiding.

■

[12]Here we rely crucially on the fact that the GL reduction only blows up the depth of the adversary by a polynomial factor (Remark 8). Therefore, allowing us to base the $\mathcal{C}_d$-hardcoreness of the GL-bit $\langle s \cdot r\rangle$ on the $\mathcal{C}_d$-hardness of the TL puzzles.

5.3.2 Size-robust Over-extractable Commitment Scheme from Injective OWFs

For a non-decreasing function $S(n)$ ($<< S^*(n)$), assume that there exists an injective one-way function (OWF) f that is hard to invert for any $\mathsf{poly}(S)$-sized circuit (for any polynomial $\mathsf{poly}(\cdot)$), but there exists a non-decreasing function $S''(n)$ ($S << S'' << S^*$) such that a circuit of $\mathsf{poly}(n)$ depth and S'' size can invert it. Such an injective OWF can be instantiated from a subexponentially secure injective OWF by setting the input length k appropriately. More concretely, consider a subexponentially secure injective OWF that is hard for circuits of size $\mathsf{poly}(2^{k^\varepsilon})$ (for any polynomial $\mathsf{poly}()$ and some $0 < \varepsilon < 1$). For any S, we can design the required f which is hard to invert for $\mathsf{poly}(S)$-sized circuits by setting $k = (\log S)^{1/\varepsilon}$, thereby achieving security against circuits of size $\mathsf{poly}(2^{k^\varepsilon}) = \mathsf{poly}(2^{(\log S)})$. Furthermore, there exists a circuit which can invert (with probability 1) by enumerating all the 2^k pre-images. Such a circuit has size $S'' = \mathsf{poly}(2^k) = \mathsf{poly}(2^{(\log S)^{1/\varepsilon}}) >> S$ and polynomial depth.

Using such an injective OWF f, we construct $(\mathsf{ECom}_S, \mathsf{EOpen}_S)$ – a commitment scheme which is hiding against circuits of size $\mathsf{poly}(S)$ (hence the name *size-robust* commitment scheme) and $(\mathsf{poly}(n), S'')$-over-extractable. $(\mathsf{ECom}_S, \mathsf{EOpen}_S)$ is simply the non-interactive commitment scheme based on injective OWFs where the hard-core predicate is the Golreich-Levin bit [107]. For completeness, we describe the scheme below.

As before, on input a security parameter 1^n, the honest committer C runs the algorithm ECom_S described below to commit to a value $v \in \{0,1\}^\alpha$. After the commit stage, the honest receiver R decides whether to accept the commitment by running the function EOpen_S as described in the reveal stage.

- Commit stage - Algorithm ECom_S:

1. On input security parameter 1^n and value $v \in \{0,1\}^\alpha$, for every $0 \leq i \leq \alpha - 1$, ECom_S samples random strings s_i in the domain of f, random strings $r_i \xleftarrow{\$} \{0,1\}^{|s_i|}$ and computes the commitment c_i to $v[i]$, the ith bit of v, as follows,

$$c_i = (f(s_i),\ r_i, \langle r_i \cdot s_i \rangle \oplus v[i])\ .$$

2. ECom_S sets the vector $c = \{c_i\}_{0 \leq i \leq \alpha-1}$ as the commitment and sets $(v, \{s_i\}_{0 \leq i \leq \alpha-1})$ as the decommitment.

- Reveal stage - Function EOpen_S:

  On input commitment $c = \{c_i\}_{0 \leq i \leq \alpha-1}$ and decommitment $(v, \{s_i\}_{0 \leq i \leq \alpha-1})$, EOpen_S returns 1 if $c_i = (f(s_i),\ r_i,\ \langle r_i \cdot s_i \rangle \oplus v[i])$ for every $0 \leq i \leq \alpha - 1$. Otherwise, outputs 0.

The extractor $o\mathcal{E}_S$ for the scheme proceeds as follows:

- Extraction - Extractor $o\mathcal{E}_S$:

  On input any commitment $c = \{c_i = (y_i, r_i, z_i)\}_{0 \leq i \leq \alpha-1}$, the extractor $o\mathcal{E}_S$ computes the pre-image s_i of y_i under f (by assumption, f can be inverted using a circuit of polynomial depth and S'' size). $o\mathcal{E}_S$ extracts bit $v[i]$ committed in c_i by computing $v[i] = z_i \oplus \langle r_i \cdot s_i \rangle$. $o\mathcal{E}_S$ returns the string $v[0]||\ldots||v[\alpha-1]$ as its output.

Theorem 17 *If f is a $\mathcal{C}^\wedge_{S,S}$-secure injective OWF which is invertible by a circuit in $\mathcal{C}^\wedge_{\mathsf{poly}(n),S''}$ for non-decreasing functions S, S'' such that $n << S << S'' << S^*$ then $(\mathsf{ECom}_S, \mathsf{EOpen}_S)$ is a non-interactive, perfectly binding, $\mathcal{C}^\wedge_{S,S}$-hiding and $(\mathsf{poly}(n), S'')$-over-extractable commitment scheme w.r.t. extractor $o\mathcal{E}_S$.*

Proof: We discuss all the properties in the following:

- Binding and Hiding: The proof of perfect binding follows from the injectivity of f and proof of $\mathcal{C}^\wedge_{S,S}$-hiding follows from the hard-coreness of the Goldreich-Levin bit with f being $\mathcal{C}^\wedge_{S,S}$ one-way (hence the scheme is $\mathcal{C}^\wedge_{S,S}$-hiding).

- Over-extractable: First, the extractor $o\mathcal{E}_S$ belongs to the class $\mathcal{C}^{\wedge}_{\mathsf{poly}(n),S''}$ since f can be inverted by a circuit in $\mathcal{C}^{\wedge}_{\mathsf{poly}(n),S''}$ and the rest of the computation takes $\mathsf{poly}(n)$ time. Furthermore, since $o\mathcal{E}_S$ always inverts OWF images y_i's correctly, it always extracts the correct unique committed value. Therefore, $(\mathsf{ECom}_S, \mathsf{EOpen}_S)$ is $(\mathsf{poly}(n), S'')$-over-extractable.

■

5.3.3 Strong Over-extractable Commitment Scheme

For non-decreasing functions,

$$n << d(n) << S'(n), S(n) << S''(n) << S^*(n) << 2^{n^\varepsilon} ,$$

we construct a non-interactive perfectly binding commitment $(\mathsf{ECom}_{d,S}, \mathsf{EOpen}_{d,S})$ which is $\mathcal{C}^{\vee}_{d,S}$-hiding and (S', S'')-over-extractable w.r.t an extractor $o\mathcal{E}_{d,S}$. Note that, unlike the commitment schemes described in Sections 5.3.1 and 5.3.2 which were either hiding against depth-restricted circuits $\mathcal{C}_d$ or hiding against size-restricted circuits $\mathcal{C}^{\wedge}_{S,S}$, $(\mathsf{ECom}_{d,S}, \mathsf{EOpen}_{d,S})$ enjoys a *stronger* security property of being hiding against circuits in both depth-restricted and size-restricted circuit classes (i.e., $\mathcal{C}^{\vee}_{d,S}$). We describe the construction of the scheme $(\mathsf{ECom}_{d,S}, \mathsf{EOpen}_{d,S})$ for an honest committer C and an honest receiver R below. The idea is to commit to a random 2-out-of-2 secret share of the value v using each of the schemes described in Sections 5.3.1 and 5.3.2.

As before, on input a security parameter 1^n, the honest committer C runs the algorithm $\mathsf{ECom}_{d,S}$ described below to commit to a value $v \in \{0,1\}^\alpha$. After the commit stage, the honest receiver R decides whether to accept the commitment by running the function $\mathsf{EOpen}_{d,S}$ as described in the reveal stage.

- Commit stage - Algorithm $\mathsf{ECom}_{d,S}$:

1. On input security parameter 1^n and value $v \in \{0,1\}^\alpha$, $\mathsf{ECom}_{d,S}$ samples a random α-bit string r_0.
2. $\mathsf{ECom}_{d,S}$ computes a commitment c_1 to r_0 using ECom_d. Let d_1 be the corresponding decommitment string.
3. $\mathsf{ECom}_{d,S}$ computes a commitment c_2 to $v \oplus r_0$ using ECom_S. Let d_2 be the corresponding decommitment string.
4. $\mathsf{ECom}_{d,S}$ sets (c_1, c_2) as the commitment c and sets (v, r_0, d_1, d_2) as the decommitment.

- Reveal stage - Function $\mathsf{EOpen}_{d,S}$:

  On input a commitment $c = (c_1, c_2)$ and the decommitment (v, r_0, d_1, d_2), $\mathsf{EOpen}_{d,S}$ accepts it if both EOpen_d and EOpen_S accept the corresponding decommitments; that is,

$$\mathsf{EOpen}_d(c_1, r_0, d_1) = 1 \wedge \mathsf{EOpen}_S(c_2, v \oplus r_0, d_2) = 1 \ .$$

  Otherwise, $\mathsf{EOpen}_{d,S}$ rejects.

The extractor $o\mathcal{E}_{d,S}$ of the scheme proceeds as follows:

- Extraction - Extractor $o\mathcal{E}_{d,S}$:

  The extractor $o\mathcal{E}_{d,S}$ on input $c = (c_1, c_2)$ runs the extractors $o\mathcal{E}_d$ and $o\mathcal{E}_S$ with inputs c_1 and c_2, obtaining outputs r'_0 and r'_1 respectively. If either r'_0 or r'_1 is $\perp$ then $o\mathcal{E}_{d,S}$ outputs $\perp$. Otherwise, $o\mathcal{E}_{d,S}$ outputs $r'_0 \oplus r'_1$.

Theorem 18 *For the following hierarchy of non-decreasing functions on* $\mathbb{N}$

$$n << d << S' << S << S'' << S^* << B \ ,$$

let $(\mathsf{ECom}_d, \mathsf{EOpen}_d)$ *be a non-interactive, perfectly binding,* $\mathcal{C}_d$*-hiding and* (S', S')*-over-extractable commitment scheme w.r.t. extractor* $o\mathcal{E}_d$ *and let* $(\mathsf{ECom}_S, \mathsf{EOpen}_S)$ *be a non-interactive, perfectly binding,* $\mathcal{C}^{\wedge}_{S,S}$*-hiding and* $(\mathsf{poly}(n), S'')$*-over-extractable commitment*

scheme w.r.t. extractor $o\mathcal{E}_S$*. Then,* $(\mathsf{ECom}_{d,S}, \mathsf{EOpen}_{d,S})$ *is non-interactive, perfectly binding,* $\mathcal{C}^{\vee}_{d,S}$*-hiding and* (S', S'')*-over-extractable commitment scheme w.r.t. extractor* $o\mathcal{E}_{d,S}$*.*

Remark 11 *For our final construction of concurrent non-malleable commitment, we require the existence of* $(\mathsf{ECom}_d, \mathsf{EOpen}_d)$ *and* $(\mathsf{ECom}_S, \mathsf{EOpen}_S)$ *for some specific functions* d, S', S, S''*. Such schemes can be based on the existence of subexponentially secure injective OWFs and* (T, B)*-TL puzzles for some subexponential functions* T, B*. We provide concrete instantiations of such depth- and size-robust schemes in Section 5.7.2.*

Proof: We discuss each of the properties in the following:

- Perfect binding: The perfect binding follows from the perfect binding of ECom_d and ECom_S.

- Over-extractable: A valid commitment $c = (c_1, c_2)$ is such that both c_1 and c_2 are valid commitments for ECom_d and ECom_S respectively. Since ECom_d and ECom_S are over-extractable w.r.t. extractors $o\mathcal{E}_d$ and $o\mathcal{E}_S$ respectively, $o\mathcal{E}_{d,S}$ which runs $o\mathcal{E}_d(c_1)$ and $o\mathcal{E}_S(c_2)$ extracts out the unique committed values and hence outputs $\mathsf{val}(c)$ with probability 1. Furthermore, $o\mathcal{E}_d \in \mathcal{C}^{\wedge}_{S',S'}$ and $o\mathcal{E}_S \in \mathcal{C}^{\wedge}_{\mathsf{poly}(n),S''}$ implies that $o\mathcal{E}_{d,S}$ belongs to the circuit class $\mathcal{C}^{\wedge}_{S',S''}$.

- Hiding: Assume towards a contradiction that there exists a polynomially bounded function $\alpha(\cdot)$, a non-uniform circuit family $A = \{A_n\}_{n\in\mathbb{N}} \in \mathcal{C}^{\vee}_{d,S}$ and for some polynomial $p(\cdot)$ and infinitely many $n \in \mathbb{N}$, a pair of values $v_0, v_1 \in \{0,1\}^{\alpha}$,

$$\mathsf{Pr}\left[b \leftarrow \{0,1\}, c \leftarrow \mathsf{ECom}_{d,S}(1^n, v_b) : b = A_n(c)\right] \geq \frac{1}{2} + \frac{1}{p(n)} \,. \tag{5.4}$$

  Using A, we construct a non-uniform circuit family $B = \{B_n\}_{n\in\mathbb{N}}$ that breaks the hiding of either ECom_d or ECom_S depending on the depth and size of A. Since

$A \in \mathcal{C}^{\vee}_{d,S}$, it could either be that $A \in \mathcal{C}_d$ or $A \in \mathcal{C}^{\wedge}_{S,S}$. We will consider the two cases separately below.

Case 1 - $A \in \mathcal{C}^{\wedge}_{S,S}$: In this case, we construct a B that violates the hiding of ECom_S as follows: B_n with v_0 and v_1 hard-wired in it, samples a random $\alpha(n)$-bit string r_0 and computes a commitment c_1 to string r_0 using ECom_d. It sends $(v_0 \oplus r_0)$ and $(v_1 \oplus r_0)$ as challenges in the hiding game of ECom_S and receives a commitment c_2 to $(v_b \oplus r_0)$, for a randomly chosen bit b. Finally, B_n sends the tuple (c_1, c_2) as the commitment to A_n and forwards the output of A_n as its output. B perfectly simulates the hiding game of $\mathsf{ECom}_{d,S}$ for A while itself participating in the hiding game of ECom_S and hence succeeds with probability at least $\frac{1}{2} + \frac{1}{p(n)}$. Furthermore, since B incurs only polynomial blow-up in size over A (while simulating the game for A), we have $B \in \mathcal{C}^{\wedge}_{S,S}$. Therefore, $B \in \mathcal{C}^{\wedge}_{S,S}$ succeeds in the hiding game of ECom_S with non-negligible probability away from $\frac{1}{2}$, which is a contradiction.

Case 2 - $A \in \mathcal{C}_d$: The proof for Case 2 is similar to Case 1 but here we, instead, construct $B \in \mathcal{C}_d$ which succeeds in the hiding game of ECom_d with non-negligible probability away from $\frac{1}{2}$. The only difference from the previous case is that B commits to r_0 using the scheme ECom_S and forwards $(v_0 \oplus r_0)$ and $(v_1 \oplus r_0)$ as challenges in the hiding game of ECom_d. Since the marginal distribution of both random shares of v (i.e., r and $v \oplus r$ for a random r) are identical, B still perfectly simulates the hiding game of $\mathsf{ECom}_{d,S}$ for A.

■

5.4 Non-malleable Commitment Scheme w.r.t. Extraction for Short Identities

For $l = O(1)$ which is a power of 2, assume that we have the following hierarchy of non-decreasing functions on $\mathbb{N}$,

$$\begin{aligned} n << d_0 << d_1 << \ldots << d_{l-1} << d_l << \\ S_0 << S_1 << \ldots << S_{l-1} << S_l << S^* << 2^{n^\varepsilon} \ , \end{aligned} \tag{5.5}$$

such that for every $0 \leq \mathsf{id} \leq l-1$,

- there exists a depth-robust commitment scheme $(\mathsf{ECom}_{d_{\mathsf{id}}}, \mathsf{EOpen}_{d_{\mathsf{id}}})$ that is $\mathcal{C}_{d_{\mathsf{id}}}$-hiding and $(d_{\mathsf{id}+1}, d_{\mathsf{id}+1})$-over-extractable w.r.t. an extractor $o\mathcal{E}_{d_{\mathsf{id}}}$.

- there exists a size-robust commitment scheme $(\mathsf{ECom}_{S_{\mathsf{id}}}, \mathsf{EOpen}_{S_{\mathsf{id}}})$ that is $\mathcal{C}^{\wedge}_{S_{\mathsf{id}}, S_{\mathsf{id}}}$-hiding and $(\mathsf{poly}(n), S_{\mathsf{id}+1})$-over-extractable w.r.t. an extractor $o\mathcal{E}_{S_{\mathsf{id}}}$.

By Section 5.3.3, we can construct a family of l commitments $\{(\mathsf{ECom}_{\mathsf{id}}, \mathsf{EOpen}_{\mathsf{id}})\}_{\mathsf{id}}$ such that for every $0 \leq \mathsf{id} \leq l-1$,

$$(\mathsf{ECom}_{\mathsf{id}}, \mathsf{EOpen}_{\mathsf{id}}) = (\mathsf{ECom}_{d_{\mathsf{id}}, S_{l-\mathsf{id}-1}}, \mathsf{EOpen}_{d_{\mathsf{id}}, S_{l-\mathsf{id}-1}}) \ ,$$

and by Theorem 18 we have that $(\mathsf{ECom}_{\mathsf{id}}, \mathsf{EOpen}_{\mathsf{id}})$ is a non-interactive, perfectly binding, $\mathcal{C}^{\vee}_{d_{\mathsf{id}}, S_{l-\mathsf{id}-1}}$-hiding and also $(d_{\mathsf{id}+1}, S_{l-\mathsf{id}})$-over-extractable commitment scheme w.r.t. an extractor $o\mathcal{E}_{\mathsf{id}}$ (described in Section 5.3.3). We use this family of l commitment schemes to construct a tag-based commitment scheme $(\mathsf{ENMCom}, \mathsf{ENMOpen})$ for identities of length $\log l$-bits which is one-one non-malleable w.r.t. extraction by an extractor $o\mathcal{E}_{\mathsf{NM}}$. We describe the scheme $(\mathsf{ENMCom}, \mathsf{ENMOpen})$ and the extractor $o\mathcal{E}_{\mathsf{NM}}$ below.

On input a security parameter 1^n, the honest committer C runs the algorithm ENMCom described below to commit to a value $v \in \{0,1\}^\alpha$. After the commit stage, the honest

receiver R decides whether to accept the commitment by running the function $\mathsf{ENMOpen}$ as described in the reveal stage.

- Commit stage - Algorithm ENMCom:

  1. On input security parameter 1^n, identity $0 \leq \mathsf{id} \leq l-1$ and a value $v \in \{0,1\}^\alpha$, ENMCom computes a commitment c to v using $\mathsf{ECom}_{\mathsf{id}}$. Let d be the corresponding decommitment string.

- Reveal stage - Function $\mathsf{ENMOpen}$:

  On input a commitment c and the decommitment (v, d) and identity id, $\mathsf{ENMOpen}$ computes $\mathsf{ENMOpen}(\mathsf{id}, c, v, d)$ which returns 1 if $\mathsf{EOpen}_{\mathsf{id}}(c, v, d)$ returns 1. Otherwise, returns 0.

The extractor $o\mathcal{E}_{\mathsf{NM}}$ proceeds as follows,

- Extraction - Extractor $o\mathcal{E}_{\mathsf{NM}}$:

  The extractor $o\mathcal{E}_{\mathsf{NM}}$ on input c and identity id outputs the value extracted by $o\mathcal{E}_{\mathsf{id}}$ from c.

Remark 12 *We want* ENMCom *and* $\mathsf{ENMOpen}$ *to be computable by uniform TMs. This mandates that* $\{\mathsf{ECom}_{\mathsf{id}}\}_{0\leq\mathsf{id}\leq l-1}$ *and* $\{\mathsf{EOpen}_{\mathsf{id}}\}_{0\leq\mathsf{id}\leq l-1}$ *be uniformly and efficiently computable; that is, there must exist uniform* PPT *TMs* M_{com} *and* M_{open} *that on input* id *can compute* $\mathsf{ECom}_{\mathsf{id}}$ *and* $\mathsf{EOpen}_{\mathsf{id}}$ *respectively. If* $l = O(1)$ *then one can simply hard-code all the algorithms* $\{\mathsf{ECom}_{\mathsf{id}}\}_{0\leq\mathsf{id}\leq l-1}$ *and* $\{\mathsf{EOpen}_{\mathsf{id}}\}_{0\leq\mathsf{id}\leq l-1}$ *in* M_{com} *and* M_{open} *respectively. As will see later,* $l = O(1)$ *is sufficient for constructing non-malleable commitment scheme for* n*-bit identities. When* $l = \omega(1)$ *the hard-coding approach, in fact, does not work. Nevertheless, we note that the algorithms* $\mathsf{ECom}_{\mathsf{id}}$ *and* $\mathsf{EOpen}_{\mathsf{id}}$ *described in Section 5.3.3 are still efficiently and uniformly computable. Since, this case does not occur in our construction, we omit details here.*

Theorem 19 $(\mathsf{ENMCom}, \mathsf{ENMOpen})$ *is a non-interactive, perfectly binding,* $\mathcal{C}^{\wedge}_{d_0,S_0}$*-hiding and* (d_l, S_l)*-over-extractable tag-based commitment scheme for identities of length* $\log l$*.* $(\mathsf{ENMCom}, \mathsf{ENMOpen})$ *is also one-one* $\mathcal{C}^{\wedge}_{d_0,S_0}$*-non-malleable w.r.t. extraction by extractor* $o\mathcal{E}_{\mathsf{NM}}$*.*

We note that both hiding and non-malleability hold only against circuits in the restrictive class $\mathcal{C}^{\wedge}_{d_0,S_0}$; that is, circuits A whose depth and size are bounded by $\mathsf{poly}(d_0)$ and $\mathsf{poly}(S_0)$ respectively, even though the building blocks $\mathsf{ECom}_{\mathsf{id}}$'s have the stronger security of being hiding against circuits in $\mathcal{C}^{\vee}_{d_{\mathsf{id}},S_{l-\mathsf{id}-1}} \supset \mathcal{C}^{\wedge}_{d_0,S_0}$; that is, circuits A which are either restricted in their depths or their size but not both.

Proof: The perfect binding follows readily from the perfect binding of each of the $\mathsf{ECom}_{\mathsf{id}}$'s. We discuss over-extractability and non-malleability in the following:

- Over-extractable: A valid commitment c with identity id is a valid commitment for $\mathsf{ECom}_{\mathsf{id}}$. Therefore, the extractor $o\mathcal{E}_{\mathsf{NM}}$ which runs $o\mathcal{E}_{\mathsf{id}}$ on c extracts the correct unique committed value due to the over-extractability of $\mathsf{ECom}_{\mathsf{id}}$ w.r.t. $o\mathcal{E}_{\mathsf{id}}$. Furthermore, $\mathsf{ECom}_{\mathsf{id}}$'s are $(d_{\mathsf{id}+1}, S_{l-\mathsf{id}})$-over-extractable and hence the depth of $o\mathcal{E}_{\mathsf{id}}$ is at most $\mathsf{poly}(d_{\mathsf{id}+1})$ and size of $o\mathcal{E}_{\mathsf{id}}$ is at most $\mathsf{poly}(S_{l-\mathsf{id}})$. Therefore, $o\mathcal{E}_{\mathsf{NM}}$ (which runs $o\mathcal{E}_{\mathsf{id}}$) is a circuit with depth bounded by $\mathsf{poly}(d_l)$ and size bounded by $\mathsf{poly}(S_l)$ (see Inequality (5.5)). Hence, $(\mathsf{ENMCom}, \mathsf{ENMOpen})$ is (d_l, S_l)-over-extractable.

- Non-malleability and Hiding: By Theorem 15 hiding will follow from the proof of non-malleability which we describe next. For proving one-one non-malleability w.r.t. extraction by $o\mathcal{E}_{\mathsf{NM}}$, let us assume for contradiction that there exists a non-uniform attacker $A = \{A_n\}_{n\in\mathbb{N}} \in \mathcal{C}^{\wedge}_{d_0,S_0}$ sending/receiving commitments to values of length $\alpha = \mathsf{poly}(n)$, a non-uniform distinguisher $D = \{D_n\}_{n\in\mathbb{N}} \in \mathcal{P}/\mathrm{poly}$, and a polynomial $p(\cdot)$, such that, for infinitely many $n \in \mathbb{N}$,

$$\left|\mathsf{Pr}[D_n(\mathsf{emim}^{A_n}_{\mathsf{ENMCom}}(1^n,0))=1]-\mathsf{Pr}[D_n(\mathsf{emim}^{A_n}_{\mathsf{ENMCom}}(1^n,1))=1]\right|\geq 1/p(n). \tag{5.6}$$

Let id and $\tilde{\mathsf{id}}$ be the identities chosen by A in the left and right interactions respectively. Let $v_0, v_1 \in \{0,1\}^\alpha$ be the two challenge values chosen by A for the left interaction. Note that since the only message A receives in the execution is the left commitment and identity and the values for the left interaction needs to be chosen before that, we can assume that the left side identity id and the challenge values v_0, v_1 are fixed.

Using A and D, we will construct a non-uniform circuit $B = \{B_n\}_{n\in\mathbb{N}} \in \mathcal{C}^\vee_{d_{\mathsf{id}},S_{l-\mathsf{id}-1}}$ that breaks the hiding of $\mathsf{ECom}_{\mathsf{id}}$ with advantage at least $\frac{1}{p(n)}$. More concretely, B internally runs A and acts as an honest committer in the left interaction with A while acts as an honest receiver in the right interaction. In the hiding game of $\mathsf{ECom}_{\mathsf{id}}$, B sends (v_0, v_1) as challenges and receives a commitment c to v_b, for a randomly chosen bit b. B forwards c to A as the commitment in the left interaction. A sends a commitment $\tilde{c}$ to the honest right receiver (simulated by B). Then, B runs the extractor $o\mathcal{E}_{\tilde{\mathsf{id}}}$ on $\tilde{c}$ obtaining an extracted value $\tilde{v}'$. Depending on the value of b, the over-extracted value $\tilde{v}'$ along with the view of A is identical to $\mathsf{emim}^{A}_{\mathsf{ENMCom}}(1^n, b)$. B runs the distinguisher D with inputs $\tilde{v}'$ and the view of A. Finally, B returns the output of D as its output.

By our hypothesis, B succeeds in breaking the hiding of $\mathsf{ECom}_{\mathsf{id}}$ with advantage at least $\frac{1}{p(n)}$. Now to arrive at a contradiction it remains to show that $B \in \mathcal{C}^\vee_{d_{\mathsf{id}},S_{l-\mathsf{id}-1}}$. B runs the extractor $o\mathcal{E}_{\tilde{\mathsf{id}}} \in \mathcal{C}^\wedge_{d_{\tilde{\mathsf{id}}+1},S_{l-\tilde{\mathsf{id}}}}$ and $A \in \mathcal{C}^\wedge_{d_0,S_0}$, while the rest of the simulation

takes $\mathsf{poly}(n)$ time. Therefore the depth of B is such that,

$$\begin{aligned}\mathsf{dep}(B) &= \mathsf{dep}(A) + \mathsf{dep}(o\mathcal{E}_{\tilde{\mathsf{id}}}) + \mathsf{poly}(n) \\ &\leq \mathsf{poly}(d_0) + \mathsf{poly}(d_{\tilde{\mathsf{id}}+1}) + \mathsf{poly}(n) < \mathsf{poly}(d_{\tilde{\mathsf{id}}+1}) \ .\end{aligned} \tag{5.7}$$

Similarly, the size of B is such that,

$$\begin{aligned}\mathsf{size}(B) &= \mathsf{size}(A) + \mathsf{size}(o\mathcal{E}_{\tilde{\mathsf{id}}}) + \mathsf{poly}(n) \\ &\leq \mathsf{poly}(S_0) + \mathsf{poly}(S_{l-\tilde{\mathsf{id}}}) + \mathsf{poly}(n) \\ &< \mathsf{poly}(S_{l-\tilde{\mathsf{id}}}) << S^* \ .\end{aligned} \tag{5.8}$$

We consider two cases for the identities id and $\tilde{\mathsf{id}}$ as follows:[13]

Case 1 - $\mathsf{id} > \tilde{\mathsf{id}}$: In this case, $d_{\mathsf{id}} \geq d_{\tilde{\mathsf{id}}+1}$, we have that $\mathsf{dep}(B) < \mathsf{poly}(d_{\mathsf{id}})$ for some polynomial $\mathsf{poly}(\cdot)$. Therefore, $B \in \mathcal{C}_{d_{\mathsf{id}}}$ and hence $B \in \mathcal{C}^{\vee}_{d_{\mathsf{id}},S_{l-\mathsf{id}-1}}$.

Case 2 - $\mathsf{id} < \tilde{\mathsf{id}}$: In this case, $S_{l-\tilde{\mathsf{id}}} \leq S_{l-\mathsf{id}-1}$ and we have that $\mathsf{size}(B) < \mathsf{poly}(S_{l-\mathsf{id}-1})$ for some polynomial $\mathsf{poly}(\cdot)$. Therefore $B \in \mathcal{C}^{\vee}_{d_{\mathsf{id}},S_{l-\mathsf{id}-1}}$.

Thus, irrespective of the identity $\tilde{\mathsf{id}}$ chosen by A for the right interaction, we can construct $B \in \mathcal{C}^{\vee}_{d_{\mathsf{id}},S_{l-\mathsf{id}-1}}$ which breaks hiding of $\mathsf{ECom}_{\mathsf{id}}$ with non-negligible advantage, which is a contradiction.

■

Remark 13 *In the above proof, the reduction B which bases the one-one non-malleability w.r.t. extraction on the hiding of $\mathsf{ECom}_{\mathsf{id}}$, runs both A and the extractor $o\mathcal{E}_{\tilde{\mathsf{id}}}$ of the commitment scheme $\mathsf{ECom}_{\tilde{\mathsf{id}}}$. Therefore, B has depth at most $\mathsf{dep}(A) + \mathsf{poly}(d_{\tilde{\mathsf{id}}+1})$ and has size at most $\mathsf{size}(A) + \mathsf{poly}(S_{l-\tilde{\mathsf{id}}})$ respectively. To reach a contradiction, one must argue that the reduction B belongs to $\mathcal{C}^{\vee}_{d_{\mathsf{id}},S_{l-\mathsf{id}}}$. In other words, either $\mathsf{dep}(A) + \mathsf{poly}(d_{\tilde{\mathsf{id}}+1})$*

[13]Note that the case $\mathsf{id} = \tilde{\mathsf{id}}$ is an invalid execution and hence not considered.

is at most $\mathsf{poly}(d_{\mathsf{id}})$ *or* $\mathsf{size}(A) + \mathsf{poly}(S_{l-\tilde{\mathsf{id}}})$ *is at most* $\mathsf{poly}(S_{l-\mathsf{id}-1})$. *Since* A *chooses both* id *and* $\tilde{\mathsf{id}}$, *this can only hold if* $\mathsf{dep}(A)$ *and* $\mathsf{size}(A)$ *are both small; that is,* $o(d_1)$ *and* $o(S_1)$ *respectively. As a result, we only show non-malleability of* $(\mathsf{ENMCom}, \mathsf{ENMOpen})$ *against weak adversaries whose depth and size both are bounded by* $\mathsf{poly}(d_0) = o(d_1)$ *and* $\mathsf{poly}(S_0) = o(S_1)$ *respectively.*

Remark 14 *Furthermore, we note that even though* $(\mathsf{ENMCom}, \mathsf{ENMOpen})$ *is non-malleable w.r.t. extraction, we cannot prove that it is non-malleable (w.r.t. commitment). This is because the underlying commitment schemes* $\mathsf{ECom}_{\mathsf{id}}$*'s are only over-extractable. Over-extractability guarantees that for a valid commitment, the value extracted by the extractor is indeed the value committed. However, when a commitment is invalid, the extracted value can be arbitrary – hence the name over-extractable. Therefore, there might exist an adversary* A *that depending on the value committed on the left sends invalid commitments with different probabilities on the right. Such an adversary clearly violates the non-malleability (w.r.t. commitment) but may not violate non-malleability w.r.t. extraction. This is because the over-extracted values may still be indistinguishable. Hence, we cannot base non-malleability (w.r.t. commitment) on non-malleability w.r.t. extraction of* $(\mathsf{ENMCom}, \mathsf{ENMOpen})$.

5.5 Strengthening Non-malleability

The scheme $(\mathsf{ENMCom}, \mathsf{ENMOpen})$ described in Section 5.4 is only stand-alone (one-one) non-malleable w.r.t. extraction. However, our final goal is to construct a scheme that is concurrent non-malleable (w.r.t. commitment). In this section, we describe a transformation that transforms any 2-round commitment scheme $\langle C, R \rangle$ which is one-one non-malleable w.r.t. extraction (against adversaries of some bounded depth and size) into a 2-round commitment scheme $\langle \widehat{C}, \widehat{R} \rangle$ which is concurrent non-malleable w.r.t.

extraction as well as concurrent non-malleable (w.r.t. commitment) (against adversaries of some other bounded depth and size), while preserving the length of the identities.

We present the transformed protocol $\langle \widehat{C}, \widehat{R} \rangle$ in Section 5.5.3. Before that, we list the building blocks used in the transformation in Section 5.5.2, and we give high-level intuition on the design of the protocol $\langle \widehat{C}, \widehat{R} \rangle$ in Section 5.5.1. In particular, in a step by step fashion, we explain the purpose of different components in the protocol. If the reader prefers to read the actual protocol directly, please skip Section 5.5.1 and start from Section 5.5.2.

5.5.1 A Bare-Bone Protocol and Challenges

As discussed in the overview in Section 5.1, our construction of $\langle \widehat{C}, \widehat{R} \rangle$ is inspired by the non-malleability amplification technique in [56]. As a starting point, their technique suggests the following bare-bone protocol:

A Bare-Bone Protocol $\langle \widehat{C}, \widehat{R} \rangle$. The receiver sends a *puzzle* puzz. Here by puzzle we mean a computationally problem that i) is hard to solve when generated honestly, and ii) has a unique solution even when generated maliciously. For instance, a puzzle could be a random image $f(x)$ of an injective one-way function whose solution is the preimage, or a randomly sampled hash function whose solution is a collision. (In particular, this puzzle does not refer to time-lock puzzles.) In addition, the receiver also sends the first message a_{NM} of $\langle C, R \rangle$ and the first message a_{ZAP} of ZAP. The committer computes a commitment $c1$ to v using a non-interactive commitment scheme Com and sends the second message b_{NM} of $\langle C, R \rangle$ committing to a random string $r1$, and the second message b_{ZAP} of ZAP proving that either i) $c1$ commits to v or ii) $(a_{\mathsf{NM}}, b_{\mathsf{NM}})$ commits to a solution s of the puzzle puzz (which is efficiently verifiable).

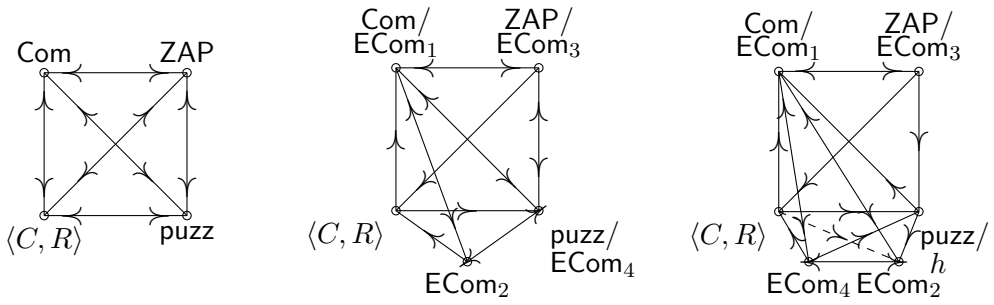
$$\widehat{C} \quad \xleftarrow{\mathsf{puzz},\, a_{\mathsf{NM}},\, a_{\mathsf{ZAP}}} \quad \widehat{R}$$
$$\widehat{C} \quad \xrightarrow{\mathsf{Com}(v),\, b_{\mathsf{NM}},\, b_{\mathsf{ZAP}}} \quad \widehat{R}$$

As discussed before, to show the security of such a bare-bone protocol, *ideally*, we would like different components — puzz, $\langle C, R\rangle$, Com, and ZAP — to be *mutually non-malleable*. Informally speaking, we say that a primitive P is more secure than a primitive Q, denoted as $\mathsf{P} \succ \mathsf{Q}$, if the security of P holds even when security of Q is broken by force; P and Q are mutually non-malleable if $\mathsf{P} \prec\succ \mathsf{Q}$. The ideal configuration is illustrated in Figure 5.2 (i). Towards realizing as many constraints in the ideal configuration as possible, the first idea is using three size-and-depth robust commitment schemes $\mathsf{ECom}_1, \mathsf{ECom}_4, \mathsf{ECom}_3$ [14] to implement Com and puzz, and augment ZAP so that they become mutually non-malleable. But, we run into problems with respect to the input non-malleable commitment $\langle C, R\rangle$.

Challenge 1: $\langle C, R\rangle$ is only secure against adversaries which have both bounded depth *AND* bounded size. (Technically, it is secure against $\mathcal{C}^{\wedge}_{d_{\mathsf{NM}}, S_{\mathsf{NM}}}$, for some d_{NM} and S_{NM}; this is the case for the basic schemes constructed in Section 5.4, as well as the schemes produced by the transformation in this section.) This type of *AND* security means either a primitive P is more secure than $\langle C, R\rangle$ or less, but cannot be mutually non-malleable. Though through a more careful analysis, we can remove some constraints w.r.t. the non-malleable commitment, it still requires $\langle C, R\rangle \prec\succ \mathsf{puzz}$, in order to show the security of the bare-bone protocol.

Challenge 2: In addition, constructing a puzzle from size-and-depth robust commitment ECom_4 is not straightforward. If we naively use $\mathsf{puzz} = \mathsf{ECom}_4(s)$ as a puzzle, a malicious man-in-the-middle can send an invalid commitment, which has no solu-

[14] The indexes are as such in order to match the protocol description later.



(i) Ideal Configuration (ii) Assume NIWI (iii) Assume CRH

Figure 5.2: The relation between different primitives. **(i)**: The ideal configuration where all primitives are mutually non-malleable to each other; however, it cannot be instantiated. **(ii)** A sufficient configuration; it can be instantiated assuming NIWI. **(iii)**: A sufficient configuration, which can be instantiated assuming collision resistant hash functions or one-way permutations. (The dashed line is by transitivity.)

tion; this would make the security proof stuck. To prevent this, one straightforward approach is asking the receiver to send two puzzles and prove using NIWI that at least one of them is well-formed. However, this requires relying on the existence of NIWI.

To resolve Challenge 1, we modify the bare bone protocol using an additional size-and-depth robust commitment ECom_2. The key idea is creating a "buffer" between $\langle C, R\rangle$ and puzz, by setting the following relation: $\mathsf{ECom}_2 \succ \langle C, R\rangle$, $\langle C, R\rangle \succ \mathsf{puzz}$, and $\mathsf{ECom}_2 \prec\succ \mathsf{puzz}$, as illustrated in Figure 5.2 (ii). Note that now the non-malleable commitment does not need to satisfy mutual non-malleability with either ECom_2 or puzz. On the other hand, the mutual non-malleability of ECom_2 and puzz helps the security proof to go through.

However, to fulfill the relation $\mathsf{ECom}_2 \prec\succ \mathsf{puzz}$, it seems necessary to instantiate

puzz using a size-and-depth robust commitment scheme, which however as mentioned in Challenge 2 above would involve using NIWI to prevent a malicious receiver from sending an invalid commitment as a puzzle which has no solution. To avoid this, we would like to set puzz to be, for example, a randomly chosen collision resistant hash (CRH) function h, or a randomly chosen image $y = f(s)$ of a one-way permutation (OWP), whose corresponding solutions are respectively a collision of h and a preimage of y. These puzzles have the advantage that their validity are efficiently verifiable and hence NIWI can be disposed. But, a problem with using, say, h as the puzzle is that, it cannot be mutually non-malleable with ECom_2. To resolve this, we use a $h \succ \mathsf{ECom}_2$, and to compensate for the fact that $h \not\prec \mathsf{ECom}_2$, we use non-uniformity in the proof as follows: When reducing to the security of ECom_2, the reduction instead of finding a collision of h by force, receives a collision as a non-uniform advice. This can be done since the puzzle h is sent in the first message completely before the ECom_2 commitment.

Unfortunately, instantiating the puzzles using CRH or OWP creates another problem: Given that $\langle C, R\rangle \succ \mathsf{puzz} = h$ and $h \succ \mathsf{ECom}_2$, it actually implies that $\langle C, R\rangle \succ \mathsf{ECom}_2$. This transitivity holds because h is only secure against attackers with bounded size. (If h were replaced with another size-and-depth robust commitment ECom', then transitivity does not hold in general.) But this means $\langle C, R\rangle$ needs to be mutually non-malleable with ECom_2 again. To solve this problem, we again use the idea of creating "buffers". More specifically, we set the following relation: $\mathsf{ECom}_4 \succ \langle C, R\rangle$, $\langle C, R\rangle \succ \mathsf{puzz}$, $\mathsf{puzz} \succ \mathsf{ECom}_2$, and $\mathsf{ECom}_2 \prec\succ \mathsf{ECom}_4$, as illustrated in Figure 5.2 (iii). Now transitivity implies that $\langle C, R\rangle \succ \mathsf{ECom}_2$, but $\langle C, R\rangle$ no longer need to be simultanously weaker than ECom_2, and only needs to be weaker than the new "buffer" ECom_4. Moreover, the mutual non-malleability between ECom_2 and ECom_4 helps the proof to go through.

5.5.2 Building Blocks

Our transformation will make use of the following building blocks. We note that the parameters associated with these building blocks are set so as to satisfy the relations as depicted in Figure 5.2 (iii), where an arrow from primitive X to primitive Y, denoted as $X \succ Y$, means that X is harder than Y.

For some hierarchy of non-decreasing functions on $\mathbb{N}$ satisfying,

$$\begin{aligned} &n << d_4 << d_3 << d_1 << d_2 << S_2 << S_1 << S_{\mathsf{CRH}} << \\ &S'_{\mathsf{CRH}} << S_{\mathsf{NM}} << S'_{\mathsf{NM}} << S_3 << S_4 << S'_4 << S^* \ , \end{aligned} \tag{5.9}$$

the transformation relies on the following building blocks,

1. $\langle C, R\rangle$ is a 2-round, tag-based commitment scheme for $t(n)$-bit identities that is $(S'_{\mathsf{NM}}, S'_{\mathsf{NM}})$-over-extractable by extractor $o\mathcal{E}_{\mathsf{NM}}$. Furthermore, $\langle C, R\rangle$ is one-one $\mathcal{C}^{\wedge}_{S_{\mathsf{NM}}, S_{\mathsf{NM}}}$-non-malleable w.r.t. extraction by $o\mathcal{E}_{\mathsf{NM}}$.[15]

2. $(\mathsf{ECom}_1, \mathsf{EOpen}_1)$ is a perfectly binding commitment scheme which is $\mathcal{C}^{\vee}_{d_1, S_1}$-hiding and (d_2, S_{CRH})-over-extractable w.r.t. extractor $o\mathcal{E}_1$.

3. $(\mathsf{ECom}_2, \mathsf{EOpen}_2)$ is a perfectly binding commitment scheme which is $\mathcal{C}^{\vee}_{d_2, S_2}$-hiding and (S_2, S_1)-over-extractable w.r.t. extractor $o\mathcal{E}_2$.

4. $(\mathsf{ECom}_3, \mathsf{EOpen}_3)$ is a perfectly binding commitment scheme which is $\mathcal{C}^{\vee}_{d_3, S_3}$-hiding and (d_1, S_4)-over-extractable w.r.t. extractor $o\mathcal{E}_3$.

5. $(\mathsf{ECom}_4, \mathsf{ECom}_4)$ is a perfectly binding commitment scheme which is $\mathcal{C}^{\vee}_{d_4, S_4}$-hiding and (d_3, S'_4)-over-extractable w.r.t. extractor $o\mathcal{E}_4$.

[15] The non-interactive scheme $(\mathsf{ENMCom}, \mathsf{ENMOpen})$ of Section 5.4 can be viewed as a 2-round scheme $\langle C, R\rangle$ where the first round message from R is the null string. Also, note that $(\mathsf{ENMCom}, \mathsf{ENMOpen})$ is stronger than what we require here – it is non-malleable against circuits in $\mathcal{C}^{\wedge}_{d,S}$ and (S', S'') over-extractable for $d << S << S' << S''$ while here $\langle \widehat{C}, \widehat{R}\rangle$ is only required to be non-malleable for circuits in $\mathcal{C}^{\wedge}_{d,d}$ and be (S, S)-over-extractable for $d << S$.

6. ZAP is a 2-round $\mathcal{C}^{\wedge}_{S^*,S^*}$-witness-indistinguishable proof.

7. $\mathcal{H} = \{H_n\}_{n\in\mathbb{N}}$ is a family of non-uniform $\mathcal{C}^{\wedge}_{S_{\mathsf{CRH}},S_{\mathsf{CRH}}}$-collision resistant hash functions such that there exists a circuit in $\mathcal{C}^{\wedge}_{S'_{\mathsf{CRH}},S'_{\mathsf{CRH}}}$ which finds collisions for $\mathcal{H}$ with probability 1.[16]

5.5.3 Commitment Scheme $\langle \widehat{C}, \widehat{R} \rangle$

Using building blocks described in the previous subsection, we now describe our construction of a 2-round, tag-based commitment scheme $\langle \widehat{C}, \widehat{R} \rangle$ for $t(n)$-bit identities that is (d_2, S_{CRH})-over-extractable w.r.t. an extractor $\widehat{o\mathcal{E}}_{\mathsf{NM}}$, and show that it is both concurrent $\mathcal{C}^{\wedge}_{d_4,d_4}$-non-malleable w.r.t. extraction by $\widehat{o\mathcal{E}}_{\mathsf{NM}}$ and concurrent $\mathcal{C}^{\wedge}_{d_4,d_4}$-non-malleable (w.r.t. commitment).

The committer $\widehat{C}$ and the receiver $\widehat{R}$ receive the security parameter 1^n and identity $\mathsf{id} \in \{0,1\}^{t(n)}$ as common input. Furthermore, $\widehat{C}$ gets a private input $v \in \{0,1\}^{\alpha}$ which is the value to be committed.

- Commit stage - First round:

  1. $\widehat{R}$ samples a hash function h from $\mathcal{H}_n$ uniformly at random.
  2. $\widehat{R}$ samples the first message a_{ZAP} of ZAP.
  3. $\widehat{R}$ generates the first message a_{NM} of $\langle C, R \rangle$ using the honest receiver R with identity id.
  4. $\widehat{R}$ sends $(h, a_{\mathsf{ZAP}}, a_{\mathsf{NM}})$ as the first round message to $\widehat{C}$.

- Commit stage - Second round:

[16] We obtain the $\mathcal{C}^{\wedge}_{S_{\mathsf{CRH}},S_{\mathsf{CRH}}}$-collision resistant family $\mathcal{H} = \{D_n\}_{n\in\mathbb{N}}$ from the $S(\lambda) = 2^{\lambda^{\varepsilon}}$-secure CRH family (for some $0 < \varepsilon < 1/2$) $\mathcal{H}' = \{H'_\lambda\}_{\lambda\in\mathbb{N}}$ (defined in Section 5.2.7) by setting $\lambda = (\log S_{\mathsf{CRH}}(n))^{\frac{1}{\varepsilon}}$ and letting $H_n = H'_\lambda$ where λ and n are the security parameters of $\mathcal{H}'$ and $\mathcal{H}$ respectively. See Section 5.7 for a rigorous discussion on instantiations of the basic building blocks required in this Section.

1. (a) $\widehat{C}$ computes a commitment $c1$ to the value v using ECom_1. Let $d1$ be the corresponding decommitment string.

   (b) $\widehat{C}$ computes a commitment $c3$ to the decommitment $(v, d1)$ of $c1$ using ECom_3.

2. (a) $\widehat{C}$ computes a commitment $c2$ to a random string $r1$ using ECom_2.

   (b) Given a_{NM}, $\widehat{C}$ computes the second message b_{NM} of $\langle C, R\rangle$ using the honest committer C with identity id to commit to a random string $r2$.

   (c) $\widehat{C}$ computes a commitment $c4$ to a random string $r3$ using ECom_4.

3. Given a_{ZAP}, $\widehat{C}$ computes the second message b_{ZAP} of ZAP to prove the following OR-statement:

   (a) *either* there exists a string $\bar{v}$ such that $c1$ is a commitment to $\bar{v}$ and $c3$ commits to a decommitment of $c1$.

   (b) *or* there exists a string $\bar{s} = (x_1, x_2)$ such that $c2$ is a commitment to $\bar{s}$ and $c4$ commits to a decommitment of $c2$ and $(a_{\mathsf{NM}}, b_{\mathsf{NM}})$ commit to a decommitment of $c4$ and $h(x_1) = h(x_2)$.

   $\widehat{C}$ proves the statement (a) by using a decommitment of $c3$ to $(v, d1)$ — decommitment of $c1$ to v — as the witness.

4. $\widehat{C}$ sends $(c1, c2, c3, c4, b_{\mathsf{NM}}, b_{\mathsf{ZAP}})$ as the second message to $\widehat{R}$ and keeps the decommitment $(v, d1)$ private.

- Reveal stage:

  On receiving $(v, d1)$ from $\widehat{C}$, $\widehat{R}$ accepts the decommitment if the ZAP proof is accepting and if $\mathsf{EOpen}_1(c1, v, d1) = 1$. Otherwise, it rejects.

We refer to the entire transcript of the interaction as the commitment c. Moreover, we say that an interaction (with transcript c) is *accepting* if the ZAP proof contained in

the commitment c is accepting. According to the reveal stage, the value of a commitment c, $\mathsf{val}(c)$ is the value committed under $c1$ (contained in c) if c is accepting. Otherwise, $\mathsf{val}(c)$ is $\perp$.

Next, we describe the extractor $\widehat{o\mathcal{E}}_{\mathsf{NM}}$ of the scheme below.

- Extraction - Extractor $\widehat{o\mathcal{E}}_{\mathsf{NM}}$:

  On receiving a commitment c and identity id, $\widehat{o\mathcal{E}}_{\mathsf{NM}}$ first verifies the ZAP proof and outputs $\perp$ if the proof is not accepting. Otherwise, it runs the extractor $o\mathcal{E}_1$ on $c1$ and outputs the extracted value v'.

Theorem 20 *$\langle \widehat{C}, \widehat{R} \rangle$ is a 2-round, perfectly binding, $\mathcal{C}^{\wedge}_{d_4,d_4}$-hiding, (d_2, S_{CRH})-over-extractable commitment scheme for identities of length $t(n)$.*

Proof: The perfectly binding property follows from that of the non-interactive commitment scheme $(\mathsf{ECom}_1, \mathsf{EOpen}_1)$. The proof of hiding will follow from the proof of Theorem 21, which we present later.

- Over-extractability: A valid commitment c to a value v, from the definition of reveal stage of $\langle \widehat{C}, \widehat{R} \rangle$, is such that the ZAP proof contained in c is accepting and $c1$ (contained in c) is a valid commitment to v using ECom_1. In this case, the extractor $\widehat{o\mathcal{E}}_{\mathsf{NM}}$ runs $o\mathcal{E}_1$ on $c1$, which by the over-extractability of ECom_1 w.r.t. $o\mathcal{E}_1$, outputs v with . Thus, $\widehat{o\mathcal{E}}_{\mathsf{NM}}$ extracts outputs $\mathsf{val}(c)$ for any valid commitment c. Moreover, $\widehat{o\mathcal{E}}_{\mathsf{NM}}$ belongs to the class $\mathcal{C}^{\wedge}_{d_2,S_{\mathsf{CRH}}}$, since $o\mathcal{E}_1 \in \mathcal{C}^{\wedge}_{d_2,S_{\mathsf{CRH}}}$ and the rest of computation by $\widehat{o\mathcal{E}}_{\mathsf{NM}}$ takes $\mathsf{poly}(n)$ time. Hence, the scheme $\langle \widehat{C}, \widehat{R} \rangle$ is (d_2, S_{CRH})-over-extractable.

■

Next, we establish the non-malleability of the scheme $\langle \widehat{C}, \widehat{R} \rangle$.

Theorem 21 *$\langle \widehat{C}, \widehat{R} \rangle$ is concurrent $\mathcal{C}^{\wedge}_{d_4,d_4}$-non-malleable w.r.t. extraction by extractor $\widehat{o\mathcal{E}}_{\mathsf{NM}}$.*

Theorem 22 *$\langle \widehat{C}, \widehat{R} \rangle$ is concurrent $\mathcal{C}^{\wedge}_{d_4,d_4}$-non-malleable (w.r.t. commitment).*

In order to prove concurrent non-malleability w.r.t. commitment, Lin, Pass and Venkitasubramaniam [55] showed that it is sufficient to prove non-malleability against adversaries participating in one left interaction and many right interactions. We refer to such an adversary as a *one-many* adversary. More precisely, they presented a reduction that, given an adversary A and a distinguisher D that break concurrent non-malleability, builds a one-many adversary $\widetilde{A}$ and a distinguisher $\widetilde{D}$ that violate one-many non-malleability. Their reduction blows up the size and the depth of the adversary $\widetilde{A}$ and the distinguisher $\widetilde{D}$ (over A and D respectively) by a $\mathsf{poly}(n)$ factor and thereby incurs a polynomial loss in security. We claim that the same reduction applies to the new notion of non-malleability w.r.t. extraction, therefore establishing that one-many non-malleability w.r.t. extraction implies concurrent non-malleability w.r.t. extraction. Moreover, we consider non-malleability (w.r.t. commitment and extraction) against circuit classes $\mathcal{C}$ which are closed under composition with $\mathcal{P}/\mathrm{poly}$, hence their reduction preserves security in terms of the circuit class against which (concurrent and one-many) non-malleability is considered — a $\mathcal{C}$-one-many non-malleable commitment scheme is $\mathcal{C}$-concurrent non-malleable. We state the extended version of their theorem below. The proof follows syntactically from the proof of Proposition 1 in [55] but for completeness we also include the formal proof in Appendix 5.10.

Theorem 23 (one-many to concurrent [55]) *Let $\langle \widehat{C}, \widehat{R} \rangle$ be a commitment scheme and $\mathcal{C}$ be a class of circuits that is closed under composition with $\mathcal{P}/\mathrm{poly}$.*

1. *If $\langle \widehat{C}, \widehat{R} \rangle$ is one-many $\mathcal{C}$-non-malleable then it is concurrent $\mathcal{C}$-non-malleable.*

2. *If* $\langle \widehat{C}, \widehat{R} \rangle$ *is one-many* $\mathcal{C}$*-non-malleable w.r.t. extraction (by extractor* $\widehat{o\mathcal{E}}_{\mathsf{NM}}$*) then it is concurrent* $\mathcal{C}$*-non-malleable w.r.t. extraction (by* $\widehat{o\mathcal{E}}_{\mathsf{NM}}$*).*

Proof of Theorem 21,22. We now proceed to prove Theorem 21, 22. Let us consider a fixed family of circuits $A = \{A_n\}_{n\in\mathbb{N}}$ belonging to the class $\mathcal{C}^{\wedge}_{d_4,d_4}$ which participates in one left interaction and $m = \mathsf{poly}(n)$ right interactions while sending/receiving commitments to values of length $\alpha = \mathsf{poly}(n)$-bits. By Theorem 23, to show Theorems 21, 22, it suffices to prove the the following:

$$\left\{\mathsf{emim}^{A}_{\langle \widehat{C},\widehat{R}\rangle}(1^n, 0)\right\}_n \approx_c \left\{\mathsf{emim}^{A}_{\langle \widehat{C},\widehat{R}\rangle}(1^n, 1)\right\}_n \tag{5.10}$$

$$\left\{\mathsf{mim}^{A}_{\langle \widehat{C},\widehat{R}\rangle}(1^n, 0)\right\}_n \approx_c \left\{\mathsf{mim}^{A}_{\langle \widehat{C},\widehat{R}\rangle}(1^n, 1)\right\}_n \tag{5.11}$$

We prove the above indistinguishability via a sequence of hybrids $\{H_j(b)\}_{0\le j\le 6}$ for $b \in \{0, 1\}$, where $H_0(b)$ is identical to an honest man-in-the-middle execution $\mathsf{MIM}(1^n, b)$ with A, and $H_j(b)$ for each $1 \le j \le 6$ runs a man-in-the-middle execution with A where the left interaction is gradually simulated. For notational convenience, we use the convention x to denote a random variable in the left interaction, and convention $\tilde{x}_i$ to denote the corresponding random variable in the i'th right interaction. For example, h denotes the hash function sent by A in the left interaction, while $\tilde{h}_i$ denotes that sent by the honest receiver in the i'th right interaction. Moreover, for each hybrid $H_j(b)$, we denote by $\mathsf{mim}^{A}_{H_j}(b)$ (and respectively, $\mathsf{emim}^{A}_{H_j}(b)$) the random variables that describe the view of A and the values $\{\widetilde{v}_i\}_{i\in[m]}$ committed to in (or respectively, $\{\widetilde{v}'_i\}_{i\in[m]}$ extracted from) the right interactions. Again, for every right interaction i, if the interaction is not accepting or its identity $\tilde{\mathsf{id}}_i$ equals to the left identity id, then $\widetilde{v}'_i = \widetilde{v}_i = \perp$; we say that a right interaction is *successful* if this case does not happen.

To show indistinguishability as described in Equation (5.11) and (5.10), we prove in Lemma 33 that the view of A and the values extracted from right interactions are indis-

tinguishable in neighboring hybrids $H_j(b)$ and $H_{j+1}(b)$ for the same b, and statistically close in $H_6(1)$ and $H_5(0)$ — this establishes Equation (5.10). Furthermore, we show that in every hybrid $H_j(b)$, values extracted from right interactions are actually identical to the actual values committed in right interactions, except with negligible probability. This shows that the emim and mim random variables are statistically close (as stated in Lemma 34) and hence establishes Equation (5.11).

Lemma 33 *For $b \in \{0,1\}$ and $0 \leq j \leq 5$, the following are computationally indistinguishable,*

$$\mathsf{emim}^A_{H_j}(b) \; ; \; \mathsf{emim}^A_{H_{j+1}}(b) \; ,$$

and $\mathsf{emim}^A_{H_0}(b) = \mathsf{emim}^A_{\langle \widehat{C}, \widehat{R} \rangle}(b)$ and $\mathsf{emim}^A_{H_6}(b) \approx_s \mathsf{emim}^A_{H_5}(0)$.

Lemma 34 *For $b \in \{0,1\}$ and $0 \leq j \leq 6$, the following are statistically close,*

$$\mathsf{emim}^A_{H_j}(b) \; ; \; \mathsf{mim}^A_{H_j}(b).$$

Towards proving the above two lemmas, we will maintain a *soundness invariant* throughout all hybrids. Recall that the protocol requires a committer to prove using ZAP that one of the following two statements is true; we refer to the first as the honest statement and the second as the fake statement.

The honest statement: either it has committed to v in $c1$ (of ECom_1) and to a decommitment $(v, d1)$ of $c1$ in $c3$ (of ECom_3),

The fake statement: or it has committed to a collision $s = (x_1, x_2)$ of the hash function h in $c2$ (of ECom_2), to a decommitment $(s, d2)$ of $c2$ in $c4$ (of ECom_4), and to a decommitment $((s, d2), d4)$ of $c4$ in $(a_{\mathsf{NM}}, b_{\mathsf{NM}})$ (of $\langle C, R \rangle$).

No-fake-witness Invariant. We say that A commits to a fake witness in a right interaction i, if the value committed by A in the non-malleable commitment $(\widetilde{a}_{\mathsf{NM}i}, \widetilde{b}_{\mathsf{NM}i})$ (i.e., $\mathsf{val}((\widetilde{a}_{\mathsf{NM}i}, \widetilde{b}_{\mathsf{NM}i}))$) is a decommitment $((\tilde{s}_i, \tilde{d2}_i), \tilde{d4}_i)$ of $\tilde{c4}_i$ such that $\tilde{s}_i$ is a collision of $\tilde{h}_i$ and $(\tilde{s}_i, \tilde{d2}_i)$ is a decommitment of $\tilde{c2}_i$.

Invariant 1 (No-fake-witness invariant) *In $H_j(b)$, the probability that there exists a right interaction i that is successful and A commits to a fake witness in it is negligible.*

We show below that this invariant holds in all hybrids. The reason that we maintain Invariant 1 is that it enforces the man-in-the-middle attacker to always prove the honest statement in every successful right interaction. When this is the case, we show that the values extracted from the right interactions are identical to the values committed to in the right interactions except from negligible probability. Formally,

Claim 9 *In every hybrid $H_j(b)$, if Invariant 1 holds, then $\mathsf{emim}^A_{H_j}(b)$ and $\mathsf{mim}^A_{H_j}(b)$ are statistically close.*

At a high level, Claim 9 follows from the soundness of ZAP and over-extractability of the commitment scheme $(\mathsf{ECom}_1, \mathsf{EOpen}_1)$. Since, Invariant 1, holds, A does not commit to a fake witness in any successful right interaction. This by the soundness of ZAP implies that A proves the honest statement which inturn, implies that commitment $\tilde{c1}_i$ is valid. By over-extractability of ECom_1 it follows that the value extracted from $\tilde{c1}_i$ (corresponds to emim) is indeed identical to the $\mathsf{val}(\tilde{c1}_i)$ which, by definition, is the value of the i-th right commitment (corresponds to mim). We detail a more formal proof in Section 5.5.4.

Moving ahead, by Claim 9 it is clear that showing Lemma 34 boils down to establishing Invariant 1. Towards this goal we further observe that Invariant 1 follows from the following invariant which will be easier to prove. Instead of reasoning about A committing to a fake witness, we keep the invariant that the value extracted from $(\widetilde{a}_{\mathsf{NM}i}, \widetilde{b}_{\mathsf{NM}i})$ is NOT a fake witness.

Invariant 2 *In $H_j(b)$, the probability that there exists a right interaction i that is successful and the value* extracted *from the non-malleable commitment $(\widetilde{a}_{\mathsf{NM}i}, \widetilde{b}_{\mathsf{NM}i})$ in this interaction is a fake witness is negligible.*

Claim 10 *In every hybrid $H_j(b)$, if Invariant 2 holds, then Invariant 1 also holds except with negligible probability.*

Proof: For every right interaction k, consider two cases:

- If the non-malleable commitment $(\widetilde{a}_{\mathsf{NM}k}, \widetilde{b}_{\mathsf{NM}k})$ in this right interaction is valid, by the over-extractability property of $\langle C, R\rangle$ w.r.t. extractor $o\mathcal{E}_{\mathsf{NM}}$ the value extracted from it is exactly equal to the value committed, . Therefore, if the value *extracted* is not a fake witness, neither is the value *committed*.
- If the non-malleable commitment $(\widetilde{a}_{\mathsf{NM}k}, \widetilde{b}_{\mathsf{NM}k})$ is not valid, the value committed is $\perp$ and cannot be a fake witness.

Hence, Invariant 2 implies Invariant 1. ■

Combining the above two claims, we have,

Lemma 35 *For $b \in \{0,1\}$ and $0 \leq j \leq 6$, if Invariant 2 holds in hybrid $H_j(b)$ then $\mathsf{emim}^A_{H_j}(b)$ and $\mathsf{mim}^A_{H_j}(b)$ are statistically close.*

Therefore, to show Theorem 21 and Theorem 22, it boils down to prove Lemma 33 and that Invariant 2 holds in all hybrids. Next, we describe our hybrids $\{H_j(b)\}_{0\leq j\leq 6}$ and show that Lemma 33 and Invariant 2 indeed hold. In this Section, we only give high level proofs of the Claims and direct the reader to Section 5.5.4 for formal proofs.

Hybrid $H_0(b)$: Hybrid $H_0(b)$ emulates an honest MIM execution $\mathsf{MIM}^A_{\langle \widehat{C}, \widehat{R}\rangle}(b)$ with A on the challenge bit b by honestly committing to the value v_b on the left and

simulating honest receivers on the right.[17] Therefore,

$$\mathsf{emim}^{A}_{H_0}(b) = \mathsf{emim}^{A}_{\langle \widehat{C}, \widehat{R} \rangle}(b) \ .$$

Next, we show that Invariant 2 holds in $H_0(b)$. In fact we show that the value extracted from the ECom_2 commitment $\widetilde{c2}_k$ in any right interaction k is not a collision of the hash function $\widetilde{h}_k$, which implies Invariant 2. At a high level this readily follows from the fact that the collision-resistance of the hash function is more secure than ECom_2, $h \succ \mathsf{ECom}_2$ (see Figure 5.2 (iii)). This is because if in some right interaction k, the attack commits to a collision of $\widetilde{h}_k$ using ECom_2, then we can construct a non-uniform circuit that violates the collision-resistance of $\widetilde{h}_k$ by extracting from $\widetilde{c2}_k$. A formal proof can be found in Section 5.5.4.

Claim 11 *For $b \in \{0,1\}$ and for every right interaction i in $H_0(b)$, the probability that i is successful and the value extracted from $(\widetilde{a}_{\mathsf{NM}i}, \widetilde{b}_{\mathsf{NM}i})$ is a fake witness, is negligible.*

Hybrid $H_1(b)$: Hybrid $H_1(b)$ proceeds identically to $H_0(b)$ except that the ECom_2 commitment $c2$ sent to A in the left interaction is generated differently. In $H_0(b)$, $c2$ is a commitment to a random string $r1$ whereas in $H_1(b)$ $c2$ is a commitment to the lexicographically first collision s of the hash function h (received as non-uniform advice). The rest of the execution is simulated identically to $H_0(b)$.

First, we show that Invariant 2 holds in $H_1(b)$. In fact we show that the value extracted from the ECom_4 commitment $\widetilde{c4}_k$ in any right interaction k is not a decommitment of $\widetilde{c2}_k$ to a collision of the hash function $\widetilde{h}_k$, which implies Invariant 2. At a high level this follows from the fact that ECom_2 is more secure than ECom_4,

[17] Recall that A in the MIM execution $\mathsf{MIM}^{A}_{\langle \widehat{C}, \widehat{R} \rangle}(b)$ sends (v_0, v_1) on the left and receives a commitment to v_b.

$\mathsf{ECom}_2 \succ \mathsf{ECom}_4$ (see Figure 5.2 (iii)), and the trick that the reduction can receive a collision of h as a non-uniform advice. Suppose that in $H_1(b)$, the value extracted from $\widetilde{c4}_k$ in some right interaction k satisfies the condition above with $1/\mathsf{poly}(n)$ probability. By Claim 11, this happens with only negligible probability in $H_0(b)$. Then we can construct a non-uniform circuit that violates the hiding of ECom_2 by extracting from $\tilde{c4}_k$. We give a formal proof in Section 5.5.4.

Claim 12 *For $b \in \{0,1\}$ and for every right interaction i in $H_1(b)$, the probability that i is successful and the value extracted from $(\widetilde{a}_{\mathsf{NM}i}, \widetilde{b}_{\mathsf{NM}i})$ is a fake witness, is negligible.*

Next we show that $\mathsf{emim}^A_{H_0}(b)$ and $\mathsf{emim}^A_{H_1}(b)$ are indistinguishable, that is, view of A and the values extracted from ECom_1 commitments in every successful right interaction are indistinguishable in $H_0(b)$ and $H_1(b)$. This essentially follows from the same proof as Claim 12, but now relying on the fact that ECom_2 is more secure than ECom_1, $\mathsf{ECom}_2 \succ \mathsf{ECom}_1$ (see Figure 5.2 (iii)). We give a formal proof in Section 5.5.4.

Claim 13 *For $b \in \{0,1\}$, the following are indistinguishable,*

$$\mathsf{emim}^A_{H_0}(b); \mathsf{emim}^A_{H_1}(b) \ .$$

Hybrid $H_2(b)$: Hybrid $H_2(b)$ proceeds identically to $H_1(b)$ except that the ECom_4 commitment $c4$ sent to A in the left interaction is generated differently. In $H_1(b)$, $c4$ is a commitment to a random string $r3$ whereas in $H_2(b)$ $c4$ is a commitment to a decommitment of $c2$ to a collision s of the hash function h. More precisely,

$H_2(b)$ first finds a collision s for the function h and then commits to s using ECom_2 under $c2$. Then it commits to the decommitment of $c2$ under $c4$. The rest of the execution is simulated identically to $H_1(b)$.

First, we show that Invariant 2 holds in $H_2(b)$. At a high level this follows from the fact that ECom_4 is more secure than $\langle C, R\rangle$, $\mathsf{ECom}_4 \succ \langle C, R\rangle$ (see Figure 5.2 (iii)). Suppose that Invariant 2 does not hold in $H_2(b)$. This means that the value extracted from the non-malleable commitment in some right interaction k is a fake witness with probability $1/\mathsf{poly}(n)$ in $H_2(b)$, but negligible in $H_1(b)$ by Claim 12. Then, we can construct a non-uniform circuit B that violates the hiding of ECom_4 by extracting from the non-malleable commitment. One slight difference from the proof of Claim 12 is that since ECom_4 is also more secure than h, $\mathsf{ECom}_4 \succ h$ (see Figure 5.2 (iii)), the reduction B can afford to find collision of h internally, instead of receiving it as a non-uniform advice. We give a formal proof in Section 5.5.4.

Claim 14 *For $b \in \{0,1\}$ and for every right interaction i in $H_2(b)$, the probability that i is successful and the value extracted from $(\widetilde{a}_{\mathsf{NM}i}, \widetilde{b}_{\mathsf{NM}i})$ is a fake witness, is negligible.*

Next we show that $\mathsf{emim}^A_{H_1}(b)$ and $\mathsf{emim}^A_{H_2}(b)$ are indistinguishable, that is, view of A and the values extracted from ECom_1 commitments in every successful right interactions are indistinguishable in $H_1(b)$ and $H_2(b)$. The proof is essentially the same as that for Claim 14, except it now relies on the fact that $\mathsf{ECom}_4 \succ \mathsf{ECom}_1$ (and $\mathsf{ECom}_4 \succ h$; see Figure 5.2 (iii)). We give a formal proof in Section 5.5.4.

Claim 15 *For $b \in \{0,1\}$, the following are indistinguishable,*

$$\mathsf{emim}^A_{H_1}(b); \mathsf{emim}^A_{H_2}(b) \ .$$

Hybrid $H_3(b)$: Hybrid $H_3(b)$ proceeds identically to $H_2(b)$ except that the second message b_{NM} of $\langle C, R\rangle$ sent to A in the left interaction is generated differently. In $H_2(b)$, b_{NM} is such that $(a_{\mathsf{NM}}, b_{\mathsf{NM}})$ commits to a random string $r2$ whereas in $H_3(b)$ b_{NM} is such that $(a_{\mathsf{NM}}, b_{\mathsf{NM}})$ commits to a decommitment of $c4$ to a decommitment of $c2$ to a collision s of the hash function h. More precisely, $H_3(b)$ generates a commitment $c2$ to the collision s (obtained by brute-force search). Let $d2$ be the corresponding decommitment string. Then, $H_3(b)$ computes the commitment $c4$ to the decommitment $(s, d2)$ of $c2$. Let $d4$ be the corresponding decommitment string. Then, given a_{NM}, $H_3(b)$ computes the second message b_{NM} to commit to $((s, d2), d4)$. The rest of the execution is simulated identically to $H_2(b)$.

First, we show that Invariant 2 holds in $H_3(b)$. At a high-level, this follows from the one-one non-malleability w.r.t. extraction of $\langle C, R\rangle$. Suppose that Invariant 2 does not hold in $H_3(b)$ then there exists a right interaction k such that the probability that it is successful and the value extracted the non-malleable commitment contained in this interaction is a fake witness is $1/\mathsf{poly}(n)$ in $H_3(b)$ and is negligible in $H_2(b)$ (by Claim 14). This violates the one-one non-malleability w.r.t. extraction of $\langle C, R\rangle$ as we formally show below. We give a formal proof in Section 5.5.4.

Claim 16 *For $b \in \{0,1\}$ and for every right interaction i in $H_3(b)$, the probability that i is successful and the value extracted from $(\widetilde{a}_{\mathsf{NM}_i}, \widetilde{b}_{\mathsf{NM}_i})$ is a fake witness, is negligible.*

Next we show that $\mathsf{emim}^A_{H_2}(b)$ and $\mathsf{emim}^A_{H_3}(b)$ are indistinguishable, that is, view of A and the values extracted from ECom_1 commitments in every successful right interactions are indistinguishable in $H_2(b)$ and $H_3(b)$. This follows from the fact that $\langle C, R\rangle$ is more secure than ECom_1, $\langle C, R\rangle \succ \mathsf{ECom}_1$ (see Figure 5.2 (iii)). Therefore,

if the distribution of values extracted from the ECom_1 commitments in the right interactions are distinguishable in $H_2(b)$ and $H_3(b)$, one can construct reduction that violates the hiding of $\langle C, R\rangle$ by extracting from the ECom_1 commitments on the right. We give a formal proof in Section 5.5.4.

Claim 17 *For $b \in \{0,1\}$, the following are indistinguishable,*

$$\mathsf{emim}^A_{H_2}(b); \mathsf{emim}^A_{H_3}(b) \ .$$

Hybrid $H_4(b)$: Hybrid $H_4(b)$ proceeds identically to $H_3(b)$ except that the second message b_{ZAP} of ZAP sent to A in the left interaction is generated differently. In $H_3(b)$, b_{ZAP} is computed by proving that $c3$ commits to a decommitment $(v_b, d1)$ of $c1$ whereas in $H_4(b)$ b_{ZAP} is computed by proving that $(a_{\mathsf{NM}}, b_{\mathsf{NM}})$ commits to $((s, d2), d4)$ which is a decommitment of $c4$ to a decommitment $(s, d2)$ of $c2$ to the collision s of the hash function h.

First, we show that Invariant 2 holds in $H_4(b)$. At a high-level, this follows from the witness indistinguishability of ZAP, which holds against subexp-sized attackers. Since $\langle C, R\rangle$ can be broken in the time that ZAP is secure against, changing the ZAP proof on the left should not change the distribution of values extracted from the right non-malleable commitments. As the values extracted from right non-malleable commitments are not fake witnesses in $H_3(b)$ (by Claim 16), the same holds for these values in $H_4(b)$. We give a formal proof in Section 5.5.4.

Claim 18 *For $b \in \{0,1\}$ and for every right interaction i in $H_4(b)$, the probability that i is successful and the value extracted from $(\widetilde{a}_{\mathsf{NM}i}, \widetilde{b}_{\mathsf{NM}i})$ is a fake witness, is negligible.*

Next we show that $\mathsf{emim}^{A}_{H_3}(b)$ and $\mathsf{emim}^{A}_{H_4}(b)$ are indistinguishable, that is, view of A and the values extracted from ECom_1 commitments in every successful right interactions are indistinguishable in $H_3(b)$ and $H_4(b)$. This follows from essentially the same proof of Claim 18, except that now we use the fact that ZAP is more secure than ECom_1. We give a formal proof in Section 5.5.4.

Claim 19 *For $b \in \{0,1\}$, the following are indistinguishable,*

$$\mathsf{emim}^{A}_{H_3}(b); \mathsf{emim}^{A}_{H_4}(b) \ .$$

Hybrid $H_5(b)$: Hybrid $H_5(b)$ proceeds identically to $H_4(b)$ except that the ECom_3 commitment $c3$ sent to A in the left interaction is generated differently. In $H_4(b)$ $c3$ is committing to the decommitment $(v_b, d1)$ of $c1$ whereas in $H_5(b)$ $c3$ is committing to 0^l where l is the length of the decommitment of $c1$. More precisely, $H_5(b)$ computes $(c1, c2, c4, b_{\mathsf{NM}})$ identically to $H_4(b)$. Then, $H_5(b)$ computes the ECom_3 commitment $c3$ to commit to 0^l. The rest of the execution is simulated identically to $H_4(b)$.

First, we show that Invariant 2 holds in $H_5(b)$. This follows from the fact that $\mathsf{ECom}_3 \succ \langle C, R \rangle$, (see Figure 5.2 (iii)). Suppose that Invariant 2 does not hold in $H_5(b)$ but holds in $H_4(b)$ by Claim 18, then there exists a right interaction k such that the probability that it is successful and the value extracted from the non-malleable commitment in it is a fake witness jumps from negligible in $H_4(b)$ to $1/\mathsf{poly}(n)$ in $H_5(b)$. Then, we can construct a reduction that violates the hiding of ECom_3 by extracting from the non-malleable commitment in the kth right interaction. We give a formal proof in Section 5.5.4.

Claim 20 *For $b \in \{0,1\}$ and for every right interaction i in $H_5(b)$, the probability that i is successful and the value extracted from $(\widetilde{a}_{\mathsf{NM}i}, \widetilde{b}_{\mathsf{NM}i})$ is a fake witness, is negligible.*

Next we show that $\mathsf{emim}^A_{H_4}(b)$ and $\mathsf{emim}^A_{H_5}(b)$ are indistinguishable, that is, view of A and the values extracted from ECom_1 commitments in every successful right interactions are indistinguishable in $H_4(b)$ and $H_5(b)$. This follows from the same proof as that of Claim 20, except that now it relies on the fact that $\mathsf{ECom}_3 \succ \mathsf{ECom}_1$. We give a formal proof in Section 5.5.4.

Claim 21 *For $b \in \{0,1\}$, the following are indistinguishable,*

$$\mathsf{emim}^A_{H_4}(b); \mathsf{emim}^A_{H_5}(b) \ .$$

Hybrid $H_6(b)$: Hybrid $H_6(b)$ proceeds identically to $H_5(b)$ except that the ECom_1 commitment $c1$ sent to A in the left interaction is generated differently. In $H_5(b)$, $c1$ is committing to the value v_b whereas in $H_6(b)$ $c1$ is committing to the value v_0 instead where (v_0, v_1) are the values sent by A in the left interaction. The rest of the execution is simulated identically to $H_5(v)$.

First, note that for $b \in \{0,1\}$ $H_6(b)$ is in fact identical to $H_5(0)$. Therefore by Claim 20 that Invariant 2 holds in $H_5(0)$, we directly have that it holds also in $H_6(b)$.

Claim 22 *For $b \in \{0,1\}$ and for every right interaction i in $H_6(b)$, the probability that i is successful and the value extracted from $(\widetilde{a}_{\mathsf{NM}i}, \widetilde{b}_{\mathsf{NM}i})$ is a fake witness, is negligible.*

Next we show that $\mathsf{emim}^{A}_{H_5}(b)$ and $\mathsf{emim}^{A}_{H_6}(b)$ are indistinguishable. This follows from the fact that ECom_1 is more secure than ECom_3, $\mathsf{ECom}_1 \succ \mathsf{ECom}_3$ (see Figure 5.2 (iii)), and the fact that Invariant 2 holds in both $H_5(b)$ and $H_6(b)$. The latter ensures that in every successful right interaction k, the attacker must prove the honest statement using ZAP that $c\tilde{3}_k$ is valid committing to a valid decommitment of $c\tilde{1}_k$ in that right interaction. Therefore, in every successful right interaction k, the value extracted from $c\tilde{3}_k$ and $\widetilde{c1_k}$ are identical. This implies that if the emim random variables are distinguishable in $H_5(b)$ and $H_6(b)$, the values extracted from the right ECom_3 commitments are also distinguishable. Then, we can construct a reduction that violates the hiding of ECom_1 by extracting from the right ECom_3 commitments. We give a formal proof in Section 5.5.4.

Claim 23 *For $b \in \{0,1\}$, the following are indistinguishable,*

$$\mathsf{emim}^{A}_{H_5}(b); \mathsf{emim}^{A}_{H_6}(b) \ .$$

This concludes the proof of Theorem 21 and Theorem 22. We direct the reader to Section 5.5.4 for the formal proofs of Claims in this Section.

5.5.4 Proofs of Claims from Section 5.5.3

In this Section, we provide formal proofs of Claims from Section 5.5.3.

Proof of Claim 9 To show that $\mathsf{emim}^{A}_{H_j}(b)$ and $\mathsf{mim}^{A}_{H_j}(b)$ are statistically close, it suffices to argue that in $H_j(b)$, in every right interaction i, the values $\widetilde{v}'_i$ extracted from this right interaction is identical to the value $\widetilde{v}_i$ committed in this right interaction,

except with negligible probability. Then the claim follows by taking a union bound over all $m = \mathsf{poly}(n)$ right interactions.

Firstly, note that if a right interaction i is not successful, then clearly $\widetilde{v}'_i = \widetilde{v}_i = \perp$. For a successful right interaction i, by the definition of extractor $\widehat{o\mathcal{E}}_{\mathsf{NM}}$, $\tilde{v}'_i$ is the value extracted by $o\mathcal{E}_1$ from the ECom_1 commitment $\widetilde{c1}_i$. Next, since Invariant 1 holds, we claim (proof presented shortly) that A proves the honest statement in successful right interactions except with negligible probability. That is,

Claim 24 *In $H_j(b)$ if Invariant 1 holds then the probability that there exists a right interaction i that is successful and A proves the fake statement in it is negligible.*

Since the honest statement is true in this right interaction i except with negligible probability, this implies that the ECom_1 commitment $\widetilde{c1}_i$ is valid. By the over-extractability of ECom_1 w.r.t. extractor $o\mathcal{E}_1$, the value extracted from $\widetilde{c1}_i$ (i.e., $\tilde{v}'_i$ in this case) is identical to the committed value $\tilde{v}_i$. Or equivalently, $\tilde{v}'_i = \tilde{v}_i$. Therefore, under Invariant 1, the random variable $\mathsf{emim}^A_{H_j}(b)$ is identical to $\mathsf{mim}^A_{H_j}(b)$, except with negligible probability. To conclude the proof of Claim 9, we now discuss the proof of Claim 24 below.

Proof of Claim 24 Let us assume that for $H_j(b)$ there exists a polynomial $p(\cdot)$ such that for infinitely many $n \in \mathbb{N}$ there exists some right interaction k that is successful and A proves the fake statement in this interaction with probability $1/p(n)$. Since Invariant 1 holds, A does not commit to the fake witness in this right interaction, except with negligible probability, which implies that the fake statement is false. Therefore, it must be that with probability at least $1/2p(n)$, the fake statement is false yet A proves the fake statement in this successful right interaction k. Given this we construct a cheating prover $\mathcal{P}^* = \{\mathcal{P}^*_n\}_{n\in\mathbb{N}}$ that breaks the soundness of ZAP with probability at least $1/2p(n)$.

$\mathcal{P}^*_n$ has k hardwired in it, participates in an interaction with the honest verifier of $\mathcal{V}$ of ZAP, internally runs A and simulates the left interaction with A as a honest committer

and all right interactions except the k-th interaction as a honest receiver. For the k-th interaction, $\mathcal{P}^*$ samples a random $\widetilde{h}_k$ and the first message $\widetilde{a}_{\mathsf{NM}k}$. It sets $\widetilde{a}_{\mathsf{ZAP}k} = \mathsf{a}$ where a is the first message received by $\mathcal{P}^*$ from the honest verifier. It sends $(\widetilde{h}_k, \widetilde{a}_{\mathsf{NM}k}, \widetilde{a}_{\mathsf{ZAP}k})$ as the first message to A for its k-th right interaction. On receiving the second message from A in the k-th right interaction, $\mathcal{P}^*$ forwards the second message $\widetilde{b}_{\mathsf{ZAP}k}$ of ZAP in the k-th right interaction as its second message $\mathsf{b} = \widetilde{b}_{\mathsf{ZAP}k}$ to the honest verifer. Then, with probability at least $1/2p(n)$ the ZAP proof (a, b) is accepting and A proves the fake statement while not committing to the fake witness. This contradicts the adaptive soundness of ZAP. □

Proof of Claim 11 We show that in $H_0(b)$ the probability that there exists a right interaction k that is successful and the value extracted from $\widetilde{c2}_k$ is a collision of the hash function $\widetilde{h}_k$ in this right interaction — refer to this event as bad — is negligible. Then the claim follows, since whenever the value extracted from the non-malleable commitment in a successful right interaction k is indeed a fake witness (refer to this event as bad_1) then the commitment $\widetilde{c2}_k$ is valid and furthermore $\widetilde{c2}_k$ commits to a collision $\widetilde{s}_k$ of the hash function $\widetilde{h}_k$. By the over-extractability of ECom_2 the value extracted from $\widetilde{c2}_k$ is indeed $\widetilde{s}_k$. In other words, the claim follows because conditioned on event bad_1 occuring, the event bad occurs.

Now suppose for contradiction that there exists $b \in \{0,1\}$ and a polynomial p such that for infinitely many $n \in \mathbb{N}$ event bad occurs with probability $1/p(n)$ in $H_0(b)$. Or equivalently, for all such n's there exists some right interaction k for which k is successful and the value extracted from $\widetilde{c2}_k$ is a collision of hash function $\widetilde{h}_k$ with probability at least $1/p(n)$.

Then, using A, we construct a non-uniform circuit $B = \{B_n\}_{n\in\mathbb{N}} \in \mathcal{C}_{S_{\mathsf{CRH}}}$ that outputs a collision for a hash function sampled from honestly from $\mathcal{H}$ (using D_n) with probability

at least $1/p(n)$. More concretely, B with k hard-wired in it, on receiving an honestly sampled hash function h^*, emulates $H_0(b)$ for A except for the kth right interaction. In the kth right interaction, B honestly computes the first message $\widetilde{a}_{\mathsf{NM}k}$ of $\langle C, R\rangle$ and the first message $\widetilde{a}_{\mathsf{ZAP}k}$ of ZAP (as in $H_0(b)$) and sends the tuple $(\widetilde{h}_k = h^*, \widetilde{a}_{\mathsf{ZAP}k}, \widetilde{a}_{\mathsf{NM}k})$ as its first round message to A. On receiving the second round message from A in the kth interaction, B runs the extractor $o\mathcal{E}_2$ on $\widetilde{c2}_k$ and returns the extracted value as its output (irrespective of whether the right interaction k is successful or not). Note that B perfectly emulates $H_0(b)$ for A as the distribution of hash function received by B is identical to the distribution of the hash function sent by the honest receiver $\widehat{R}$ of $\langle \widehat{C}, \widehat{R}\rangle$. Then by our hypothesis, the extracted value is a collision of the function $\widetilde{h}_k = h^*$ with probability at least $1/p(n)$.

Furthermore, we argue that B belongs to the circuit class $\mathcal{C}_{S_{\mathsf{CRH}}}$: B internally runs A and $o\mathcal{E}_2$, and the rest of computation performed by B for emulating $H_0(b)$ takes $\mathsf{poly}(n)$ time. Since $o\mathcal{E}_2 \in \mathcal{C}^{\wedge}_{S_2,S_1}$ and $A \in \mathcal{C}^{\wedge}_{d_4,d_4}$ we have,

$$\begin{aligned}\mathsf{size}(B) &= \mathsf{size}(A) + \mathsf{size}(o\mathcal{E}_2) + \mathsf{poly}(n)\\ &\leq \mathsf{poly}(d_4) + \mathsf{poly}(S_1)\\ &< \mathsf{poly}(S_{\mathsf{CRH}}) \qquad (\text{since, } S_{\mathsf{CRH}} >> d_4, S_1 \text{ from Equation (5.9)})\end{aligned}$$

Thus, B belongs to the class $\mathcal{C}^{\wedge}_{S_{\mathsf{CRH}},S_{\mathsf{CRH}}}$ which contradicts collision-resistance of $\mathcal{H}$. □

Proof of Claim 12 We show that in $H_1(b)$ the probability that there exists a right interaction k that is successful and the value extracted from $\tilde{c4}_k$ is a decommitment of $\tilde{c2}_k$ to a collision of the hash function $\tilde{h}_k$ in this right interaction — refer to this event as bad — is negligible. Then the claim follows, since whenever the value extracted from the non-malleable commitment in a successful right interaction k is indeed a fake witness (refer to this event as bad_1) then the commitment $\widetilde{c4}_k$ is valid and furthermore $\widetilde{c4}_k$ commits to a decommitment of $\tilde{c2}_i$ to a collision $\widetilde{s}_k$ of the hash function $\widetilde{h}_k$. Then,

by the over-extractability of ECom_4 we know that the value extracted from $\widetilde{c4}_k$ is indeed a decommitment of $\widetilde{c2}_k$ to a collision $\widetilde{s}_k$ of hash function $\widetilde{h}_k$. In other words, the claim follows because conditioned on event bad_1 occuring, the event bad occurs.

Towards bounding the probability of bad, first observe that by if bad occurs then for some right interaction k, $\widetilde{c2}_k$ must be a valid commitment and therefore extractor $o\mathcal{E}_2$ finds a collision $\tilde{s}_k$. Then, by Claim 11 we can conclude that the bad occurs in $H_0(b)$ only with negligible probability.

Now suppose for contradiction that there exists $b \in \{0,1\}$ and a polynomial p such that for infinitely many $n \in \mathbb{N}$ the event bad occurs with probability $1/p(n)$ in $H_1(b)$. Or equivalently, for all such n's there exists some right interaction k such that k is successful and the value extracted from $\widetilde{c4}_k$ is a decommitment of $\widetilde{c2}_k$ to a collision of hash function $\widetilde{h}_k$ with probability at least $1/p(n)$. Consider the set Γ of prefixes of transcripts up to the point where the first message in the left interaction is sent. By a standard averaging argument, there must exist a $1/2p(n)$ fraction of prefixes ρ in Γ, such that, conditioned on ρ occurring in $H_1(b)$, the probability that bad occurs is at least $1/2p(n)$. Therefore, there exist at least a $1/3p(n)$ fraction of prefixes ρ in Γ, such that, conditioned on ρ occurring in both $H_0(b)$ and $H_1(b)$, the probability that bad occurs increases by at least $1/3p(n)$ across hybrids. Fix one such prefix ρ; let h be the hash function contained in the first message in the left interaction in ρ and $s = (x_1, x_2)$ be the lexicographically first collision of h.

Then, using A, the prefix ρ and its collision s, we construct a non-uniform circuit $B \in \mathcal{C}^{\vee}_{d_2,S_2}$ that violates the hiding of $(\mathsf{ECom}_2, \mathsf{EOpen}_2)$ with advantage at least $1/3p(n)$.

The circuit B with k, ρ, and s hard-wired in it, participates in the hiding game of $(\mathsf{ECom}_2, \mathsf{EOpen}_2)$ and internally emulates an execution of $H_1(b)$ with A as follows: [18]

[18]For right interactions whose messages are not in ρ, B sends the first-round message by running the honest receiver $\widehat{R}$.

- Step 1: Feed A with messages in ρ; let $(h, a_{\mathsf{ZAP}}, a_{\mathsf{NM}})$ be the left first message.

- Step 2: It samples a random string $r1$, sends $r1$ and $s = (x_1, x_2)$ as challenges in the hiding game of $(\mathsf{ECom}_2, \mathsf{EOpen}_2)$, and receives a commitment c^* to either $r1$ or s.

- Step 3: B generates the second message of the left interaction identically to $H_1(b)$ except that it embeds c^* as the ECom_2 commitment in the message. That is, B computes $(c1, c3, c4, b_{\mathsf{NM}})$ as in $H_1(b)$ (and $H_0(b)$) and then computes the second message of ZAP (b_{ZAP}) by setting $c2 = c^*$ using the honest witness as done in $H_1(b)$. It then sends $(c1, c2, c3, c4, b_{\mathsf{NM}}, b_{\mathsf{ZAP}})$ as the second round message in the left interaction to A.

- Step 4: Once, B receives the second round message in the kth right interaction, if the interaction is not successful then B outputs 0. Otherwise, it runs the extractor $o\mathcal{E}_4$ on $\tilde{c4}_k$ and outputs 1 iff the extracted value is a decommitment of $\tilde{c2}_k$ to a collision of the function $\tilde{h}_k$ in right interaction k.

It is easy to see that if B receives a commitment to the random string $r1$, then it is perfectly emulates $H_0(b)$ conditioned on ρ occurring for A and if it receives a commitment to the solution s which is a collision of h then it perfectly emulates $H_1(b)$ conditioned on ρ occurring for A. As argued before, the probability that bad occurs increases by at least $1/3p(n)$. Therefore, B has advantage at least $1/3p(n)$ in violating the hiding of $(\mathsf{ECom}_2, \mathsf{EOpen}_2)$.

Moreover, we show that $B \in \mathcal{C}^{\vee}_{d_2,S_2}$: B internally runs $A \in \mathcal{C}^{\wedge}_{d_4,d_4}$, $o\mathcal{E}_4 \in \mathcal{C}^{\wedge}_{d_3,S'_4}$, and

the rest of the computation done by B takes $\mathsf{poly}(n)$ time. Thus,

$$\begin{aligned}\mathsf{dep}(B) &\leq \mathsf{dep}(A) + \mathsf{dep}(o\mathcal{E}_4) + \mathsf{poly}(n)\\ &\leq \mathsf{poly}(d_4) + \mathsf{poly}(d_3)\\ &< \mathsf{poly}(d_2) \qquad (\text{since, } d_2 >> d_4, d_3 \text{ from Equation (5.9)})\end{aligned}$$

and $\mathsf{size}(B) = \mathsf{poly}(S_4') < \mathsf{poly}(S^*)$. Therefore, B belongs to the circuit class $\mathcal{C}_{d_2}$ (resp., $B \in \mathcal{C}^{\vee}_{d_2,S_2}$) which contradicts the $\mathcal{C}^{\vee}_{d_2,S_2}$-hiding of the scheme $(\mathsf{ECom}_2, \mathsf{EOpen}_2)$. Hence, the claim holds. □

Proof of Claim 13 Let us assume for contradiction that there exists $b \in \{0,1\}$, a polynomial p and a distinguisher $D \in \mathcal{P}/\text{poly}$ such that for infinitely many $n \in \mathbb{N}$ D distinguishes $\mathsf{emim}^A_{H_0}(b)$ from $\mathsf{emim}^A_{H_1}(b)$ with probability $1/p(n)$.

Now, consider the set Γ of prefixes of transcripts up to the point where the first message in the left interaction is sent. By a standard averaging argument, there must exist a $1/2p(n)$ fraction of prefixes ρ in Γ, such that, conditioned on ρ occurring in both $H_0(b)$ and $H_1(b)$, the probability that D distinguishes the distributions is at least $1/2p(n)$. Fix one such prefix ρ; let h be the hash function contained in the first message in the left interaction in ρ and $s = (x_1, x_2)$ be lexicographically first collision of h. Then, using A, the prefix ρ and its collision s, we construct a non-uniform circuit $B \in \mathcal{C}^{\vee}_{d_2,S_2}$ that violates the hiding of $(\mathsf{ECom}_2, \mathsf{EOpen}_2)$ with advantage at least $1/2p(n)$.

The circuit B is similar in spirit to the circuit described in the proof of Claim 12. B with ρ and s hard-wired in it, participates in $(\mathsf{ECom}_2, \mathsf{EOpen}_2)$'s hiding game and internally emulates an execution of $H_1(b)$ with A as follows:

- Steps 1,2 and 3 are identical to the adversarial circuit described in Claim 12.

- Step 4: After A terminates, for every successful right interaction i, B runs the extractor $o\mathcal{E}_1$ on $\widetilde{c1}_i$ to obtain values $\widetilde{v}'_i$. For every unsuccessful right interaction i,

B sets $\widetilde{v}'_i = \perp$.

- Step 5: B then runs D with the view of A and the values $\{\widetilde{v}_i'\}_{i\in[m]}$ as inputs, and returns the output of D as its output.

It is easy to see that if B receives a commitment to the random string $r1$, then it perfectly emulates $H_0(b)$ conditioned on ρ occurring for A and if it receives a commitment to the solution s which is a collision of h then it perfectly emulates $H_1(b)$ conditioned on ρ occurring for A. Moreover, for every successful interaction i, B sets $\widetilde{v}'_i$ to the value extracted by $o\mathcal{E}_1$ from $\widetilde{c1}_i$ and for every unsuccessful interaction, it sets $\widetilde{v}'_i = \perp$. Therefore, the input to D (by B) is identical to $\mathsf{emim}^A_{H_0}(b)$ in the former case and it is identical to $\mathsf{emim}^A_{H_1}(b)$ in the latter case. Since D distinguishes the distributions with probability $1/2p(n)$, B wins the hiding game with advantage at least $1/2p(n)$.

Next, we argue that $B \in \mathcal{C}^{\vee}_{d_2,S_2}$: Apart from running A, B runs $o\mathcal{E}_1$ on $m = \mathsf{poly}(n)$ commitments $\widetilde{c1}_i$, and the rest of the computation takes polynomial time (includes running D). Since, $A \in \mathcal{C}^{\wedge}_{d_4,d_4}$ and $o\mathcal{E}_1 \in \mathcal{C}^{\wedge}_{d_2,S_{\mathsf{CRH}}}$, we have,

$$\begin{aligned}
\mathsf{dep}(B) &= \mathsf{dep}(A) + m \cdot \mathsf{dep}(o\mathcal{E}_1) + \mathsf{poly}(n) \\
&\leq \mathsf{poly}(d_4) + \mathsf{poly}(n) \cdot \mathsf{poly}(d_2) \\
&< \mathsf{poly}(d_2) \qquad \text{(since, } d_2 >> d_4 \text{ from Equation (5.9))}
\end{aligned}$$

and $\mathsf{size}(B) = \mathsf{poly}(S_{\mathsf{CRH}}) < \mathsf{poly}(S^*)$. Therefore, B belongs to the circuit class C_{d_2} (resp., $B \in \mathcal{C}^{\vee}_{d_2,S_2}$) which contradicts the $\mathcal{C}^{\vee}_{d_2,S_2}$-hiding of $(\mathsf{ECom}_2, \mathsf{EOpen}_2)$. Hence, the claim holds. □

Proof of Claim 14 Let us assume for contradiction that there exists $b \in \{0,1\}$ and a polynomial p such that for infinitely many $n \in \mathbb{N}$ there exists a right interaction k such that k is successful and the value extracted from $(\widetilde{a}_{\mathsf{NM}k}, \widetilde{b}_{\mathsf{NM}k})$, is a fake witness with

probability at least $1/p(n)$. Then, using A we construct a non-uniform circuit $B \in \mathcal{C}^{\vee}_{d_4,S_4}$ that violates the hiding of $(\mathsf{ECom}_4, \mathsf{EOpen}_4)$ with advantage at least $1/2p(n)$.

The circuit B with k hard-wired in it, participates $(\mathsf{ECom}_4, \mathsf{EOpen}_4)$'s hiding game and internally emulates an execution of $H_2(b)$ with A as follows:

- Step 1: On receiving the first message $(h, a_{\mathsf{ZAP}}, a_{\mathsf{NM}})$ from A, B obtains the lexicographically first collision s for the hash function h via brute-force.[19]

- Step 2: It computes commitment $c2$ to the collision s. Let $d2$ be the corresponding decommitment string.

- Step 3: It samples a random string $r3$ and sends $r3$ and $(s, d2)$ (decommitment of $c2$ to s) as challenges in the hiding game of $(\mathsf{ECom}_4, \mathsf{EOpen}_4)$, and receives a commitment c^* to either $r3$ or $(s, d2)$.

- Step 4: B generates the second message of the left interaction identically to $H_2(b)$ except that it embeds c^* as the ECom_4 commitment in the message. That is, B computes $(c1, c3, b_{\mathsf{NM}})$ as in $H_2(b)$ (and $H_1(b)$) and then computes the second message of ZAP (b_{ZAP}) by setting $c4 = c^*$ and using the honest witness. It then sends $(c1, c2, c3, c4, b_{\mathsf{NM}}, b_{\mathsf{ZAP}})$ as the second round message in the left interaction to A.

- Step 5: Once, B receives the second round message in the kth right interaction, if the interaction is not successful then B outputs 0. Otherwise, it runs the extractor $o\mathcal{E}_{\mathsf{NM}}$ on $(\widetilde{a}_{\mathsf{NM}k}, \widetilde{b}_{\mathsf{NM}k})$ and outputs 1 iff the extracted value is a fake witness (i.e., B outputs 1 iff the extracted value is a decommitment of $\widetilde{c4}_k$ to a decommitment of $\widetilde{c2}_k$ to a collision $\widetilde{s}_k$ of $\widetilde{h}_k$).

[19] From now onwards we will, unless specified otherwise, refer to the collision s for the hash function h in the left interaction as the lexicographically first such collision. We avoid writing it explicitly from now on.

It is easy to see that if B receives a commitment to the random string $r3$, then it perfectly emulates $H_1(b)$ for A and if it receives a commitment to the decommitment of $c2$ to a collision s of h then it perfectly emulates $H_2(b)$ for A. By Claim 12, in the former case, the extracted value is a fake witness with only negligible probability. Therefore, B outputs 1 with negligible probability. In the latter case, by our assumption that the right interaction k is successful and the value extracted is a fake witness with probability $1/p(n)$; B outputs 1 with probability at least $1/p(n)$. Therefore, B has advantage at least $1/2p(n)$ in violating the hiding of $(\mathsf{ECom}_4, \mathsf{EOpen}_4)$.

Moreover, we show that $B \in \mathcal{C}^{\vee}_{d_4,S_4}$: B internally runs $A \in \mathcal{C}^{\wedge}_{d_4,d_4}$, $o\mathcal{E}_{\mathsf{NM}} \in \mathcal{C}^{\wedge}_{S'_{\mathsf{NM}},S'_{\mathsf{NM}}}$, finds a collision for h using a circuit in $\mathcal{C}^{\wedge}_{S'_{\mathsf{CRH}},S'_{\mathsf{CRH}}}$ the rest of the computation done by B takes $\mathsf{poly}(n)$ time. Thus, we have,

$$\begin{aligned}
\mathsf{size}(B) &= \mathsf{size}(A) + \mathsf{size}(o\mathcal{E}_{\mathsf{NM}}) + \mathsf{poly}(S'_{\mathsf{CRH}}) + \mathsf{poly}(n) \\
&\leq \mathsf{poly}(d_4) + \mathsf{poly}(S'_{\mathsf{NM}}) + \mathsf{poly}(S'_{\mathsf{CRH}}) \\
&< \mathsf{poly}(S_4) \qquad (\text{since, } S_4 >> S'_{\mathsf{NM}}, S'_{\mathsf{CRH}}, d_4 \text{ from Equation (5.9)})
\end{aligned}$$

Therefore, B belongs to the circuit class $\mathcal{C}^{\wedge}_{S_4,S_4}$ (resp., $B \in \mathcal{C}^{\vee}_{d_4,S_4}$) which contradicts the $\mathcal{C}^{\vee}_{d_4,S_4}$-hiding of $(\mathsf{ECom}_4, \mathsf{EOpen}_4)$. Hence, the claim holds. □

Proof of Claim 15 Let us assume for contradiction that there exists $b \in \{0,1\}$, a polynomial p and a distinguisher $D \in \mathcal{P}/\text{poly}$ such that for infinitely many $n \in \mathbb{N}$ D distinguishes $\mathsf{emim}^A_{H_1}(b)$ from $\mathsf{emim}^A_{H_2}(b)$ with probability $1/p(n)$. Then using A and D, we construct a non-uniform circuit $B \in \mathcal{C}^{\vee}_{d_4,S_4}$ that violates the hiding of $(\mathsf{ECom}_4, \mathsf{EOpen}_4)$ with non-negligible advantage $1/p(n)$. B is similar in spirit to the circuit described in the proof of Claim 14.

B participates in the hiding game of ECom_4 and internally emulates an execution of $H_2(b)$ with A as follows:

- Steps 1, 2, 3 and 4 are identical to the adversarial circuit described in Claim 14.

- Step 5: After A terminates, for every successful right interaction i, B runs the extractor $o\mathcal{E}_1$ on $\widetilde{c1}_i$ to obtain values $\widetilde{v}'_i$. For every unsuccessful right interaction i, B sets $\widetilde{v}'_i = \perp$.

- Step 6: B then runs D with the view of A and the values $\{\widetilde{v}_i'\}_{i\in[m]}$ as inputs, and returns the output of D as its output.

It is easy to see that if B receives a commitment to the random string $r3$, then it perfectly emulates $H_1(b)$ for A and if it receives a commitment to the decommitment of $c2$ to a collision s of h then it perfectly emulates $H_2(b)$ for A. Moreover, for every successful interaction i, B sets $\widetilde{v}'_i$ to the value extracted by $o\mathcal{E}_1$ from $\widetilde{c1}_i$ and for every unsuccessful interaction, it sets $\widetilde{v}'_i = \perp$. Therefore, the input to D (by B) is identical to $\mathsf{emim}^A_{H_1}(b)$ in the former case and it is identical to $\mathsf{emim}^A_{H_2}(b)$ in the latter case. Since D distinguishes the distributions with probability $1/p(n)$, B wins the hiding game with advantage at least $1/p(n)$.

Next, we argue that $B \in \mathcal{C}^{\vee}_{d_4,S_4}$: Apart from running A and finding a collision for h, B runs $o\mathcal{E}_1$ on $m = \mathsf{poly}(n)$ commitments $\widetilde{c1}_i$, and the rest of the computation takes polynomial time (includes running D). Since, $A \in \mathcal{C}^{\wedge}_{d_4,d_4}$, $o\mathcal{E}_1 \in \mathcal{C}^{\wedge}_{d_2,S_{\mathsf{CRH}}}$ and a collision for h can be found by a circuit in $\mathcal{C}^{\wedge}_{S'_{\mathsf{CRH}},S'_{\mathsf{CRH}}}$, we have,

$$\begin{aligned}
\mathsf{size}(B) &= \mathsf{size}(A) + m \cdot \mathsf{size}(o\mathcal{E}_1) + \mathsf{poly}(S'_{\mathsf{CRH}}) + \mathsf{poly}(n) \\
&\leq \mathsf{poly}(d_4) + \mathsf{poly}(n) \cdot \mathsf{poly}(S_{\mathsf{CRH}}) + \mathsf{poly}(S'_{\mathsf{CRH}}) \\
&< \mathsf{poly}(S_4) \qquad (\text{since, } S_4 >> S_{\mathsf{CRH}}, S'_{\mathsf{CRH}}, d_4 \text{ from Equation (5.9)})
\end{aligned}$$

Therefore, B belongs to the circuit class C_{S_4} (resp., $B \in \mathcal{C}^{\vee}_{d_4,S_4}$) which contradicts the $\mathcal{C}^{\vee}_{d_4,S_4}$-hiding of $(\mathsf{ECom}_4, \mathsf{EOpen}_4)$. Hence, the claim holds. □

Proof of Claim 16 Let us assume for contradiction that there exists $b \in \{0,1\}$ and a polynomial p such that for infinitely many $n \in \mathbb{N}$ there exists a right interaction k

such that k is successful and the value $((\widetilde{s}'_k, \widetilde{d2}'_k), \widetilde{d4}'_k)$, extracted from $(\widetilde{a}_{\mathsf{NM}k}, \widetilde{b}_{\mathsf{NM}k})$, is a fake witness with probability at least $1/p(n)$. Then, using A we construct a non-uniform circuit $A_{\mathsf{NM}} \in \mathcal{C}^{\wedge}_{S_{\mathsf{NM}}, S_{\mathsf{NM}}}$, that participates in one left interaction with C and one right interaction with R, and a distinguisher D_{NM} that violate the one-one non-malleability of $\langle C, R\rangle$ w.r.t. extraction with advantage at least $1/2p(n)$. We detail the circuits A_{NM} and D_{NM} below.

The circuit A_{NM} with k hard-wired in it, participates in one left interaction with C and one right interaction with R and internally emulates an execution of $H_3(b)$ with A as follows:

- Step 1: A_{NM} waits for A to select identities for the left interaction with $\widehat{C}$ and the kth right interaction with $\widehat{R}$ while emulating $\widehat{R}$ for all other right interactions. Let id and $\widetilde{\mathsf{id}}_k$ be the respective identities.

- Step 2: A_{NM} selects identity $\mathsf{id}_l = \mathsf{id}$ for its left interaction and identity $\mathsf{id}_r = \widetilde{\mathsf{id}}_k$ for its right interaction r. On receiving the first-round message $a_{\mathsf{NM}r}$ from R, A_{NM} samples a hash function $\widetilde{h}_k$ and the first message of ZAP, $\widetilde{a}_{\mathsf{ZAP}k}$. It sends the tuple $(\widetilde{h}_k, \widetilde{a}_{\mathsf{NM}k} = a_{\mathsf{NM}r}, \widetilde{a}_{\mathsf{ZAP}k})$ as the first-round message to A in the kth right interaction.

- Step 3: On receiving the first message $(h, a_{\mathsf{ZAP}}, a_{\mathsf{NM}})$ from A, A_{NM} obtains a collision s for h via brute-force search.

- Step 4: A_{NM} computes commitments $(c1, c2, c3, c4)$ as in $H_3(b)$. Let $d2$ be the decommitment string of the commitment $c2$, which commits to the collison s. Furthermore, let $d4$ be the decommitment string of $c4$ which commits to a decommitment of $c2$.

- Step 5: A_{NM} samples a random string $r2$ and sends $a_{\mathsf{NM}l} = a_{\mathsf{NM}}$ as the first message

to C along with the values $r2$ and $((s, d2), d4)$ as challenges and receives the second message $b_{\mathsf{NM}l}$ such that $(a_{\mathsf{NM}l}, b_{\mathsf{NM}l})$ either commit to $r2$ or $((s, d2), d4)$.

- Step 6: A_{NM} computes the second message of ZAP (b_{ZAP}) by setting $b_{\mathsf{NM}} = b_{\mathsf{NM}l}$ using the honest witness. Then, it sends $(c1, c2, c3, c4, b_{\mathsf{NM}}, b_{\mathsf{ZAP}})$ as the second round message to A in the left interaction.

- Step 7: On receiving the second message $(\widetilde{c1}_k, \widetilde{c2}_k, \widetilde{c3}_k, \widetilde{c4}_k, \widetilde{b}_{\mathsf{NM}k}, \widetilde{b}_{\mathsf{ZAP}k})$ from A in the kth right interaction, B forwards $b_{\mathsf{NM}r} = \widetilde{b}_{\mathsf{NM}k}$ as the second message to R.

The distinguisher D_{NM} with input the view of A_{NM} and the value v'_r, extracted from $(a_{\mathsf{NM}r}, b_{\mathsf{NM}r})$ by $o\mathcal{E}_{\mathsf{NM}}$, runs as follows:

- D_{NM} reconstructs the entire transcript of the kth right interaction of A_{NM} with A from the view.

- If the ZAP proof $(\widetilde{a}_{\mathsf{ZAP}k}, \widetilde{b}_{\mathsf{ZAP}k})$ in the kth interaction is not accepting then D_{NM} outputs 0.

- Otherwise, D_{NM} outputs 1 iff the extracted value v'_r is such that it is a decommitment of $\widetilde{c4}_k$ to a decommitment of $\widetilde{c2}_k$ to a collision of the hash $\widetilde{h}_k$.

It is easy to see that if A_{NM} receives $b_{\mathsf{NM}l}$ such that $(a_{\mathsf{NM}l}, b_{\mathsf{NM}l})$ commit to a random string $r2$ then it perfectly emulates $H_2(v)$ for A and if $b_{\mathsf{NM}l}$ is such that $(a_{\mathsf{NM}l}, b_{\mathsf{NM}l})$ commit to $((s, d2), d4)$ then it perfectly emulates $H_3(b)$ for A. By Claim 14, in the former case, the extracted value v'_r is a fake witness with only negligible probability. Therefore, D_{NM} outputs 1 with negligible probability. In the latter case, by our assumption that the right interaction k is successful and the value extracted is a fake witness with probability $1/p(n)$; D_{NM} outputs 1 with probability at least $1/p(n)$. Therefore, D_{NM} has advantage at least $1/2p(n)$ in distinguishing the two cases, implying $(A_{\mathsf{NM}}, D_{\mathsf{NM}})$ break the one-one non-malleability w.r.t. extraction of $\langle C, R\rangle$.

Moreover, we argue that $A_{\mathsf{NM}} \in \mathcal{C}^{\wedge}_{S_{\mathsf{NM}},S_{\mathsf{NM}}}$ and $D_{\mathsf{NM}} \in \mathcal{P}/\text{poly}$: Firstly, note that $D_{\mathsf{NM}} \in \mathcal{P}/\text{poly}$ as all the computation done by D_{NM} only takes polynomial time.

Next, for A_{NM}: A_{NM} internally runs $A \in \mathcal{C}^{\wedge}_{d_4,d_4}$, finds a collision for h using a circuit in $\mathcal{C}_{S'_{\mathsf{CRH}}}$ and the rest of the computation done by A_{NM} takes $\mathsf{poly}(n)$ time. Therefore, the size $\mathsf{size}(A_{\mathsf{NM}})$ of A_{NM} satisfies the following,

$$\begin{aligned}\mathsf{size}(A_{\mathsf{NM}}) &= \mathsf{size}(A) + \mathsf{poly}(S'_{\mathsf{CRH}}) + \mathsf{poly}(n) \\ &\leq \mathsf{poly}(d_4) + \mathsf{poly}(S'_{\mathsf{CRH}}) \\ &< \mathsf{poly}(S_{\mathsf{NM}}) \qquad (\text{since, } S_{\mathsf{NM}} >> d_4, S'_{\mathsf{CRH}} \text{ from Equation (5.9)})\end{aligned} \tag{5.12}$$

Thus, A_{NM} belongs to the circuit class $\mathcal{C}^{\wedge}_{S_{\mathsf{NM}},S_{\mathsf{NM}}}$ which contradicts the $\mathcal{C}^{\wedge}_{S_{\mathsf{NM}},S_{\mathsf{NM}}}$-one-one non-malleability w.r.t. extraction of $\langle C, R\rangle$. Hence, the claim holds. □

Proof of Claim 17 Let us assume for contradiction that there exists $b \in \{0,1\}$, a distinguisher $D \in \mathcal{P}/\text{poly}$ and a polynomial p such that D distinguishes $\mathsf{emim}^{A}_{H_2}(b)$ from $\mathsf{emim}^{A}_{H_3}(b)$ with probability $1/p(n)$. Then using A and D, we construct a non-uniform circuit $B \in \mathcal{C}^{\wedge}_{S_{\mathsf{NM}},S_{\mathsf{NM}}}$ that violates the hiding of $\langle C, R\rangle$ with non-negligible advantage $1/p(n)$. B is similar in spirit to the circuit A_{NM} described in the proof of Claim 16.

B participates in the hiding game of $\langle C, R\rangle$ and internally emulates an execution of $H_3(b)$ with A as follows:

- Step 1: On receiving the first message $(h, a_{\mathsf{ZAP}}, a_{\mathsf{NM}})$ from A, B obtains a collision s for the hash function h via brute-force.

- Step 2: B computes commitments $(c1, c2, c3, c4)$ as in $H_3(b)$. Let $d2$ be the decommitment string of the commitment $c2$, which commits to the collision s. Furthermore, let $d4$ be the decommitment string of the commitment $c4$ to the decommitment $c2$.

- Step 3: B samples a random string $r2$ and sends a_{NM} as the first message to C along with the values $r2$ and $((s, d2), d4)$ as challenges and receives the second message b_{NM} such that $(a_{\mathsf{NM}}, b_{\mathsf{NM}})$ either commit to $r2$ or $((s, d2), d4)$.

- Step 4: B computes the ZAP proof using the honest witness and sends $(c1, c2, c3, c4, b_{\mathsf{NM}}, b_{\mathsf{ZAP}})$ as the second round message to A in the left interaction.

- Step 5: After A terminates, for every successful right interaction i, B runs the extractor $o\mathcal{E}_1$ on $\widetilde{c1}_i$ to extract values $\widetilde{v}'_i$. For every unsuccessful right interaction i, B sets $\widetilde{v}'_i = \perp$.

- Step 6: B then runs D with the view of A and the values $\{\widetilde{v_i}'\}_{i\in[m]}$ as inputs, and returns the output of D as its output.

It is easy to see that if second message b_{NM} received by B is such that $(a_{\mathsf{NM}}, b_{\mathsf{NM}})$ commit to a random string $r2$, then B is perfectly emulating $H_2(b)$ for A and if b_{NM} is such that $(a_{\mathsf{NM}}, b_{\mathsf{NM}})$ commits to $((s, d2), d4)$, then it perfectly emulating $H_3(b)$ for A. Moreover, for every successful interaction i, B sets $\widetilde{v}'_i$ to the value extracted by $o\mathcal{E}_1$ from $\widetilde{c1}_i$ and for every unsuccessful interaction B sets $\widetilde{v}'_i = \perp$. Therefore, the input to D (by B) is identical to $\mathsf{emim}^A_{H_2}(b)$ in the former case and it is identical to $\mathsf{emim}^A_{H_3}(b)$ in the latter case. Since D distinguishes the distributions with probability $1/p(n)$, B wins the hiding game with advantage at least $1/p(n)$.

Next, we argue that $B \in \mathcal{C}^{\wedge}_{S_{\mathsf{NM}}, S_{\mathsf{NM}}}$: Apart from running A and using a circuit in $\mathcal{C}_{S'_{\mathsf{CRH}}}$ to find the collision s, B runs $o\mathcal{E}_1$ on $m = \mathsf{poly}(n)$ commitments $\widetilde{c1}_i$, and the rest of the computation takes polynomial time (including running D). Since, $A \in \mathcal{C}^{\wedge}_{d_4, d_4}$ and

$o\mathcal{E}_1 \in \mathcal{C}^{\wedge}_{d_2,S_{\mathsf{CRH}}}$, the size of B satisfies the following,

$$\begin{aligned}\mathsf{size}(B) &= \mathsf{size}(A) + m \cdot \mathsf{size}(o\mathcal{E}_1) + \mathsf{poly}(S'_{\mathsf{CRH}}) + \mathsf{poly}(n) \\ &\leq \mathsf{poly}(d_4) + \mathsf{poly}(n) \cdot \mathsf{poly}(S_{\mathsf{CRH}}) + \mathsf{poly}(S'_{\mathsf{CRH}}) \\ &< \mathsf{poly}(S_{\mathsf{NM}}) \quad (\text{since, } S_{\mathsf{NM}} >> d_4, S_{\mathsf{CRH}}, S'_{\mathsf{CRH}} \text{ from Equation (5.9)})\end{aligned} \tag{5.13}$$

Therefore, B belongs to the circuit class $\mathcal{C}^{\wedge}_{S_{\mathsf{NM}},S_{\mathsf{NM}}}$ which contradicts $\mathcal{C}^{\wedge}_{S_{\mathsf{NM}},S_{\mathsf{NM}}}$-hiding of $\langle C, R\rangle$. Hence, the claim holds. □

Proof of Claim 18 Let us assume for contradiction that there exists $b \in \{0,1\}$ and a polynomial p such that for infinitely many $n \in \mathbb{N}$ there exists a right interaction k such that k is successful and the value extracted from $(\widetilde{a}_{\mathsf{NM}k}, \widetilde{b}_{\mathsf{NM}k})$, is a fake witness with probability at least $1/p(n)$. Then, using A we construct a non-uniform circuit $B \in \mathcal{C}_{S^*}$ that violates the $\mathcal{C}_{S^*}$-WI of ZAP with advantage at least $1/2p(n)$.

The circuit B with k hard-wired in it, participates in the WI game of ZAP and internally emulates an execution of $H_4(b)$ with A as follows:

- Step 1: On receiving the first message $(h, a_{\mathsf{ZAP}}, a_{\mathsf{NM}})$ from A, B obtains a collision s to the hash function h. Let (v_0, v_1) be the values chosen by A for the left interaction.

- Step 2: B computes commitments $(c1, c2, c3, c4, b_{\mathsf{NM}})$ (as in $H_4(b)$). Let $d1$ be the decommitment string of the commitment $c1$, which commits to the value v_b, $d4$ be the decommitment of $c4$ which commits to $(s, d2)$ where $d2$ is the decommitment string of the commitment $c2$, which commits to the collison s. Furthermore, let $d3$ and d be the decommitments of $c3$ and $(a_{\mathsf{NM}}, b_{\mathsf{NM}})$.

- Step 3: B sends a_{ZAP} as the first message in the WI game of ZAP with the statement $x = (h, c1, c2, c3, c4, a_{\mathsf{NM}}, b_{\mathsf{NM}})$ and witnesses $w_0 = (v_b, d1, d3)$ and $w_1 = (((s, d2), d4), d)$. B receives the second message b_{ZAP} of ZAP that is either computed by using the witness w_0 or w_1.

- Step 4: B sends $(c1, c2, c3, c4, b_{\mathsf{NM}}, b_{\mathsf{ZAP}})$ as the second message to A on the left.

- Step 5: Once, B receives the second round message in the kth right interaction, if the interaction is not successful then B outputs 0. Otherwise, it runs the extractor $o\mathcal{E}_{\mathsf{NM}}$ on $(\widetilde{a}_{\mathsf{NM}k}, \widetilde{b}_{\mathsf{NM}k})$ and outputs 1 iff the extracted value is a fake witness.

It is easy to see that if the second message b_{ZAP} of ZAP is computed using the witness $w_0 = (v_b, d1, d3)$ then B perfectly emulates $H_3(b)$ for A and if the second message b_{ZAP} of ZAP is computed using the witness $w_1 = (((s, d2), d4), d)$ then B perfectly emulates $H_4(b)$ for A. By Claim 16, in the former case, the extracted value is a fake witness with only negligible probability. Therefore, B outputs 1 with negligible probability. In the latter case, by our assumption that k is successful and the value extracted is a fake witness with probability $1/p(n)$; B outputs 1 with probability at least $1/p(n)$. Therefore, B has advantage at least $1/2p(n)$ in violating the WI of ZAP.

Moreover, we show that $B \in \mathcal{C}_{S^*}$: B internally runs $A \in \mathcal{C}^{\wedge}_{d_4,d_4}$, $o\mathcal{E}_{\mathsf{NM}} \in \mathcal{C}^{\wedge}_{S'_{\mathsf{NM}},S'_{\mathsf{NM}}}$, obtains a collision for h by using a circuit in $\mathcal{C}_{S'_{\mathsf{CRH}}}$ and the rest of the computation done by B takes $\mathsf{poly}(n)$ time. Thus, we have,

$$\begin{aligned}
\mathsf{size}(B) &= \mathsf{size}(A) + \mathsf{poly}(S'_{\mathsf{CRH}}) + \mathsf{size}(o\mathcal{E}_{\mathsf{NM}}) + \mathsf{poly}(n) \\
&\leq \mathsf{poly}(d_4) + \mathsf{poly}(S'_{\mathsf{CRH}}) + \mathsf{poly}(S'_{\mathsf{NM}}) \\
&< \mathsf{poly}(S^*) \qquad (\text{since, } S^* >> d_4, S'_{\mathsf{CRH}}, S'_{\mathsf{NM}} \text{ from Equation (5.9)})
\end{aligned}$$

Therefore, B belongs to the circuit class $\mathcal{C}_{S^*}$ which contradicts the $\mathcal{C}_{S^*}$-witness indistinguishability of ZAP. Hence, the claim holds. □

Proof of Claim 19 Let us assume for contradiction that there exists $b \in \{0, 1\}$, a polynomial p and a distinguisher D such that for infinitely many $n \in \mathbb{N}$ D distinguishes $\mathsf{emim}^A_{H_3}(b)$ from $\mathsf{emim}^A_{H_4}(b)$ with probability $\frac{1}{p(n)}$. Then using A and D, we construct a

non-uniform circuit $B \in \mathcal{C}_{S^*}$ that violates the $\mathcal{C}_{S^*}$-WI of ZAP with advantage at least $1/p(n)$. B is similar in spirit to the circuit described in the proof of Claim 18.

B with participates in the WI game of ZAP and internally emulates an execution of $H_4(b)$ with A as follows:

- Steps 1,2,3 and 4 are identical to the circuit described in Claim 18.
- Step 5: After A terminates, for every successful right interaction i, B runs the extractor $o\mathcal{E}_1$ on $\widetilde{c1}_i$ to extract values $\widetilde{v}'_i$. For every unsuccessful right interaction i, B sets $\widetilde{v}'_i = \perp$.
- Step 6: B then runs D with the view of A and the values $\{\widetilde{v_i}'\}_{i\in[m]}$ as inputs, and returns the output of D as its output.

It is easy to see that if the second message b_{ZAP} of ZAP is computed using the witness $w_0 = (v_b, d1, d3)$ then B perfectly emulates $H_3(b)$ for A and if the second message b_{ZAP} of ZAP is computed using the witness $w_1 = (((s, d2), d4), d)$ then B perfectly emulates $H_4(b)$ for A. Moreover, for every successful interaction i, B sets $\widetilde{v}'_i$ to the value extracted by $o\mathcal{E}_1$ from $\widetilde{c1}_i$ and for every unsuccessful interaction, it sets $\widetilde{v}'_i = \perp$. Therefore, the input to D (by B) is identical to $\mathsf{emim}^A_{H_3}(b)$ in the former case and it is identical to $\mathsf{emim}^A_{H_4}(b)$ in the latter case. Since D distinguishes the distributions with probability $1/p(n)$, B wins the hiding game with advantage at least $1/p(n)$.

Next, we argue that $B \in \mathcal{C}_{S^*}$: Apart from running A and finding a collision for h, B runs $o\mathcal{E}_1$ on $m = \mathsf{poly}(n)$ commitments $\widetilde{c1}_i$, and the rest of the computation takes polynomial time (includes running D). Since, $A \in \mathcal{C}^{\wedge}_{d_4,d_4}$ and $o\mathcal{E}_1 \in \mathcal{C}^{\wedge}_{d_2,S_{\mathsf{CRH}}}$, we have,

$$\begin{aligned}
\mathsf{size}(B) &= \mathsf{size}(A) + \mathsf{poly}(S'_{\mathsf{CRH}}) + m \cdot \mathsf{size}(o\mathcal{E}_1) + \mathsf{poly}(n) \\
&\leq \mathsf{poly}(d_4) + \mathsf{poly}(S'_{\mathsf{CRH}}) + \mathsf{poly}(n) \cdot \mathsf{poly}(S_{\mathsf{CRH}}) \\
&< \mathsf{poly}(S^*) \qquad (\text{since, } S^* >> d_4, S_{\mathsf{CRH}}, S'_{\mathsf{CRH}} \text{ from Equation (5.9)})
\end{aligned}$$

Therefore, B belongs to the circuit class $\mathcal{C}_{S^*}$ which contradicts the $\mathcal{C}_{S^*}$-WI of ZAP. Hence, the claim holds. □

Proof of Claim 20 Let us assume for contradiction that there exists $b \in \{0,1\}$ and a polynomial p such that for infinitely many $n \in \mathbb{N}$ there exists a right interaction k such that k is successful and the value extracted from $(\widetilde{a}_{\mathsf{NM}k}, \widetilde{b}_{\mathsf{NM}k})$, is a fake witness with probability at least $1/p(n)$. Then, using A we construct a non-uniform circuit $B \in \mathcal{C}^{\vee}_{d_3,S_3}$ that violates the hiding of $(\mathsf{ECom}_3, \mathsf{EOpen}_3)$ with advantage at least $1/2p(n)$.

The circuit B with k hard-wired in it, participates in $(\mathsf{ECom}_3, \mathsf{EOpen}_3)$'s hiding game, and internally emulates an execution of $H_5(b)$ with A as follows:

- Step 1: On receiving the first message $(h, a_{\mathsf{ZAP}}, a_{\mathsf{NM}})$ from A, B obtains a collison s to the hash function h. Let (v_0, v_1) be the values chosen by A for the left interaction.

- Step 2: It computes $(c1, c2, c4, b_{\mathsf{NM}})$ as in $H_5(b)$. Let $d1$ be the decommitment string of the commitment $c1$ which is a commitment to v_b.

- Step 3: Then in the hiding game of $(\mathsf{ECom}_3, \mathsf{EOpen}_3)$, B sends $(v_b, d1)$ and 0^l as challenges and receives a commitment c^* to either $(v_b, d1)$ or 0^l.

- Step 4: B generates the second message of ZAP (b_{ZAP}) by setting $c3 = c^*$. It then sends $(c1, c2, c3, c4, b_{\mathsf{NM}}, b_{\mathsf{ZAP}})$ as the second round message in the left interaction to A.

- Step 5: Once, B receives the second round message in the kth right interaction, if the interaction is not successful then B outputs 0. Otherwise, it runs the extractor $o\mathcal{E}_{\mathsf{NM}}$ on $(\widetilde{a}_{\mathsf{NM}k}, \widetilde{b}_{\mathsf{NM}k})$ and outputs 1 iff the extracted value is a fake witness.

It is easy to see that if B receives a commitment to $(v_b, d1)$, then it perfectly emulates $H_4(b)$ for A and if it receives a commitment to 0^l then it perfectly emulates $H_5(b)$ for A.

By Claim 18, in the former case, the extracted value is a fake witness with only negligible probability. Therefore, B outputs 1 with negligible probability. In the latter case, by our assumption that the right interaction k is successful and the value extracted is a fake witness with probability $1/p(n)$; B outputs 1 with probability at least $1/p(n)$. Therefore, B has advantage at least $1/2p(n)$ in violating the hiding of ECom_3.

Next, we argue that $B \in \mathcal{C}^{\vee}_{d_3,S_3}$: B internally runs $A \in \mathcal{C}^{\wedge}_{d_4,d_4}$, $o\mathcal{E}_{\mathsf{NM}} \in \mathcal{C}^{\wedge}_{S'_{\mathsf{NM}},S'_{\mathsf{NM}}}$, obtains a collision for h using a circuit in $\mathcal{C}_{S'_{\mathsf{CRH}}}$ and the rest of the computation done by B takes $\mathsf{poly}(n)$ time. Thus, we have,

$$\begin{aligned}
\mathsf{size}(B) &= \mathsf{size}(A) + \mathsf{size}(o\mathcal{E}_{\mathsf{NM}}) + \mathsf{poly}(S'_{\mathsf{CRH}}) + \mathsf{poly}(n) \\
&\leq \mathsf{poly}(d_4) + \mathsf{poly}(S'_{\mathsf{NM}}) + \mathsf{poly}(S'_{\mathsf{CRH}}) \\
&< \mathsf{poly}(S_3) \qquad (\text{since, } S_3 >> d_4, S'_{\mathsf{NM}}, S'_{\mathsf{CRH}} \text{ from Equation (5.9)})
\end{aligned}$$

Therefore, B belongs to the circuit class $\mathcal{C}^{\wedge}_{S_3,S_3}$ (resp., $B \in \mathcal{C}^{\vee}_{d_3,S_3}$) which contradicts the $\mathcal{C}^{\vee}_{d_3,S_3}$-hiding of $(\mathsf{ECom}_3, \mathsf{EOpen}_3)$. Hence, the claim holds. □

Proof of Claim 21 Let us assume for contradiction that there exists $b \in \{0,1\}$, a polynomial p and a distinguisher $D \in \mathcal{P}/\text{poly}$ such that for infinitely many $n \in \mathbb{N}$ D distinguishes $\mathsf{emim}^A_{H_4}(b)$ from $\mathsf{emim}^A_{H_5}(b)$ with probability $1/p(n)$. Then using A and D, we construct a non-uniform circuit $B \in \mathcal{C}^{\vee}_{d_3,S_3}$ that violates the hiding of $(\mathsf{ECom}_3, \mathsf{EOpen}_3)$ with non-negligible advantage $1/p(n)$. B is similar in spirit to the circuit described in the proof of Claim 18.

B participates in the hiding game of the scheme $(\mathsf{ECom}_3, \mathsf{EOpen}_3)$ and internally emulates an execution of $H_5(b)$ with A as follows:

- Steps 1-4 are identical to the adversarial circuit described in Claim 20.

- Step 5: After A terminates, for every successful right interaction i, B runs the extractor $o\mathcal{E}_1$ on $\widetilde{c1}_i$ to extract values $\widetilde{v}'_i$. For every unsuccessful right interaction i,

B sets $\widetilde{v}'_i = \perp$.

- Step 6: B then runs D with the view of A and the values $\{\widetilde{v}_i{}'\}_{i\in[m]}$ as inputs, and returns the output of D as its output.

It is easy to see that if B receives a commitment to $(v_b, d1)$, then it perfectly emulates $H_4(b)$ for A and if it receives a commitment to 0^l then it perfectly emulates $H_5(b)$ for A. Moreover, B for every successful interaction i, sets $\widetilde{v}'_i$ to the value extracted by $o\mathcal{E}_1$ from $\widetilde{c1}_i$ and for every unsuccessful interaction, it sets $\widetilde{v}'_i = \perp$. Therefore, the input to D (by B) is identical to $\mathsf{emim}^A_{H_4}(b)$ in the former case and it is identical to $\mathsf{emim}^A_{H_5}(b)$ in the latter case. Since D distinguishes the distributions with probability $1/p(n)$, B wins the hiding game with advantage at least $1/p(n)$.

Next, we argue that $B \in \mathcal{C}^{\vee}_{d_3,S_3}$: Apart from running A and finding a collision for h using a circuit in $\mathcal{C}_{S'_{\mathsf{CRH}}}$, B runs $o\mathcal{E}_1$ on $m = \mathsf{poly}(n)$ commitments $\widetilde{c1}_i$, and the rest of the computation takes polynomial time (includes running D). Since, $A \in \mathcal{C}^{\wedge}_{d_4,d_4}$ and $o\mathcal{E}_1 \in \mathcal{C}^{\wedge}_{d_2,S_{\mathsf{CRH}}}$, we have,

$$\begin{aligned}
\mathsf{size}(B) &= \mathsf{size}(A) + m \cdot \mathsf{size}(o\mathcal{E}_1) + \mathsf{poly}(S'_{\mathsf{CRH}}) + \mathsf{poly}(n) \\
&\leq \mathsf{poly}(d_4) + \mathsf{poly}(n) \cdot \mathsf{poly}(S_{\mathsf{CRH}}) + \mathsf{poly}(S'_{\mathsf{CRH}}) \\
&< \mathsf{poly}(S_3) \qquad (\text{since, } S_3 >> S_{\mathsf{CRH}}, d_4, S'_{\mathsf{CRH}} \text{ from Equation (5.9)})
\end{aligned}$$

Therefore, B belongs to the circuit class $\mathcal{C}_{S_3}$ (resp., $B \in \mathcal{C}^{\vee}_{d_3,S_3}$) which contradicts the $\mathcal{C}^{\vee}_{d_3,S_3}$-hiding of $(\mathsf{ECom}_3, \mathsf{EOpen}_3)$. Hence, the claim holds. □

Proof of Claim 23 Let us assume for contradiction that there exists $b \in \{0,1\}$, a polynomial p and a distinguisher $D \in \mathcal{P}/\text{poly}$ such that for infinitely many $n \in \mathbb{N}$ D distinguishes $\mathsf{emim}^A_{H_5}(b)$ from $\mathsf{emim}^A_{H_6}(b)$ with probability $1/p(n)$.

Now, consider the set Γ of prefixes of transcripts up to the point where the first message in the left interaction is sent. By a standard averaging argument, there must

exist a $1/2p(n)$ fraction of prefixes ρ in Γ, such that, conditioned on ρ occurring in both $H_5(b)$ and $H_6(b)$, the probability that D distinguishes the distributions is at least $1/2p(n)$. Fix one such prefix ρ; let h be the hash function contained in the first message in the left interaction in ρ and $s = (x_1, x_2)$ be the lexicographically first collision of h. Then, using A, the prefix ρ and its collision s, we construct a non-uniform circuit $B \in \mathcal{C}_{d_1}$ that violates the hiding of $(\mathsf{ECom}_1, \mathsf{EOpen}_1)$ with advantage at least $1/3p(n)$.

B with ρ and s hard-wired in it, participates in $(\mathsf{ECom}_1, \mathsf{EOpen}_1)$'s hiding game and internally emulates an execution of $H_6(b)$ with A as follows:

- Step 1: Feed A with messages in ρ; let $(h, a_{\mathsf{ZAP}}, a_{\mathsf{NM}})$ be the left first message. Let (v_0, v_1) be the values sent by A in the left interaction.

- Step 2: B sends v_b and v_0 as challenges in the hiding game of the scheme $(\mathsf{ECom}_1, \mathsf{EOpen}_1)$ and receives a commitment c^* to either v_b or v_0.

- Step 3: B generates the second message of the left interaction identically to $H_6(b)$ except that it embeds c^* as the ECom_1 commitment in the message. That is, B computes $(c2, c3, c4, b_{\mathsf{NM}})$ as in $H_6(b)$ (using the collision s received as non-uniform advice) and then computes the second message of ZAP (b_{ZAP}) by setting $c1 = c^*$. It then sends $(c1, c2, c3, c4, b_{\mathsf{NM}}, b_{\mathsf{ZAP}})$ as the second round message in the left interaction to A.

- Step 4: After A terminates, for every successful right interaction i, B runs the extractor $o\mathcal{E}_3$ on $\widetilde{c3}_i$ to extract values $(\widetilde{v}'_i, \widetilde{d1}'_i)$. For every unsuccessful right interaction i, B sets $\widetilde{v}'_i = \perp$.

- Step 4: B then runs D with the view of A and the values $\{\widetilde{v}'_i\}_{i\in[m]}$ as inputs, and returns the output of D as its output.

It is easy to see that if B receives a commitment to v_b, then it perfectly emulates $H_5(b)$ conditioned on ρ occurring for A and if it receives a commitment to v_0 then it perfectly emulates $H_6(b)$ conditioned on ρ occurring for A. Moreover, for every successful interaction i, B sets $\widetilde{v}'_i$ to the value extracted by $o\mathcal{E}_3$ from $\widetilde{c3}_i$ and for every unsuccessful interaction, it sets $\widetilde{v}'_i = \perp$. We claim that the input to D (by B) is statistically close to $\mathsf{emim}^A_{H_5}(b)$ in the former case and it is statistically close to $\mathsf{emim}^A_{H_6}(b)$ in the latter case; the proof of claim is presented shortly. Since D distinguishes $\mathsf{emim}^A_{H_5}(b)$ from $\mathsf{emim}^A_{H_5}(b)$ with probability $1/2p(n)$, we conclude that B wins the hiding game with advantage at least $1/3p(n)$.

Next, we argue that $B \in \mathcal{C}_{d_1}$: Apart from running A, B runs $o\mathcal{E}_3$ on $m = \mathsf{poly}(n)$ commitments $\widetilde{c3}_i$, and the rest of the computation takes polynomial time (includes running D). Since, $A \in \mathcal{C}^{\wedge}_{d_4,d_4}$ and $o\mathcal{E}_3 \in \mathcal{C}^{\wedge}_{d_1,S_4}$,

$$\begin{aligned}
\mathsf{dep}(B) &= \mathsf{dep}(A) + m \cdot \mathsf{dep}(o\mathcal{E}_3) + \mathsf{poly}(n) \\
&\leq \mathsf{poly}(d_4) + \mathsf{poly}(n) \cdot \mathsf{poly}(d_1) \\
&< \mathsf{poly}(d_1) \qquad (\text{since, } d_1 >> d_4 \text{ from Equation (5.9)})
\end{aligned}$$

Furthermore, $\mathsf{size}(B) < \mathsf{poly}(S^*)$. Therefore, B belongs to the circuit class $\mathcal{C}_{d_1}$ (resp., $B \in \mathcal{C}^{\vee}_{d_1,S_1}$) which contradicts the $\mathcal{C}^{\vee}_{d_1,S_1}$-hiding of $(\mathsf{ECom}_1, \mathsf{EOpen}_1)$.

It remains to show our claim that the input to distinguisher D by adversary B (i.e., view of A and the values $\{\widetilde{v}_i{}'\}_{i\in[m]}$) is indeed (1) statistically close to $\mathsf{emim}^A_{H_5}(b)$ when B receives a commitment to v_b and (2) statistically close to $\mathsf{emim}^A_{H_6}(0)$ when it receives a commitment to v_0. We will argue (1) and the proof of (2) follows similarly. Recall that for every successful right interaction i, B runs $o\mathcal{E}_3$ on $\widetilde{c3}_i$ to obtain $(\widetilde{v}'_i, \widetilde{d1}_i{}')$. We claim that the value $\widetilde{v}'_i$ is identical to the value extracted by $o\mathcal{E}_1$ from $\widetilde{c1}_i$, except with negligible probability. Since i is successful, by Claim 22 we know that with overwhelming probability the value extracted from $(\widetilde{a}_{\mathsf{NM}i}, \widetilde{b}_{\mathsf{NM}i})$ is not a fake witness with overwhelming

probability. Then by the over-extractability $\langle C, R\rangle$ we know that the value committed in $(\widetilde{a}_{\mathsf{NM}i}, \widetilde{b}_{\mathsf{NM}i})$ is not a fake witness. Furthermore, due to soundness of ZAP, it must be that with overwhelming probability the commitments $\widetilde{c1}_i$ and $\widetilde{c3}_i$ are valid and $\widetilde{c3}_i$ commits to a decommitment of $\widetilde{c1}_i$. Then, by the over-extractability of $(\mathsf{ECom}_3, \mathsf{EOpen}_3)$ the value $(\widetilde{v}'_i, \widetilde{d1}'_i)$ extracted from $\widetilde{c3}_i$ is identical to $\mathsf{val}(\widetilde{c3}_i)$ with over-whelming probability, where $\mathsf{val}(\widetilde{c3}_i)$ is a decommitment of $\widetilde{c1}_i$ — $(\widetilde{v}_i, \widetilde{d1}_i)$. Next, due to the over-extractability of ECom_1, the value extracted by $o\mathcal{E}_1$ from $\widetilde{c1}_i$ is identical to $\mathsf{val}(\widetilde{c1}_i) = \widetilde{v}_i$ with overwhelming probability. Therefore, the value $\widetilde{v}_i$ obtained by B is identical to the value that $o\mathcal{E}_1$ extracts from $\widetilde{c1}_i$ with overwhelming probability. This is now sufficient to conclude that the input to D is statistically close to $\mathsf{emim}^A_{H_5}(b)$ when B receives a commitment to v_b except with negligible probability. This establishes (1) and (2) follows by the same argument. Hence the claim holds. □

5.6 Amplifying Length of Identities – Log n trick

The Non-malleability strengthening technique (Section 5.5.3) applied to the scheme $(\mathsf{ENMCom}, \mathsf{ENMOpen})$ of Section 5.4, that supports identities of length $t(n) = O(1)$, results in a concurrent non-malleable commitment scheme but still only supports identities of length $t(n) = O(1)$. However, our final goal is to construct a scheme that supports identities of length n. In this section, we provide a transformation that amplifies the length of identities exponentially.

Given a tag-based commitment scheme $\langle \widehat{C}, \widehat{R}\rangle$ for $t(n)$-bit identities which is concurrent non-malleable w.r.t. commitment, Dolev, Dwork and Naor [51] construct a tag-based commitment scheme $\langle \tilde{C}, \tilde{R}\rangle$ for exponentially larger identities, namely identities of length $2^{t(n)-1}$-bits. In their work [51], they show that their transformation results in a commitment scheme that can accomodate significantly larger length of identities but

degrades concurrent non-malleability w.r.t. commitment to stand-alone non-malleability w.r.t. commitment. Furthermore, their reduction also incurs a polynomial security loss.

The commitment schemes considered in this work are non-malleable w.r.t. extraction and we claim that their transformation also works for such schemes. That is, we show that if $\langle \widehat{C}, \widehat{R} \rangle$ is concurrent non-malleable w.r.t. extraction then commitment scheme $\langle \tilde{C}, \tilde{R} \rangle$ is standalone non-malleable w.r.t. extraction. The key idea towards amplifying the length of identities is embedding a $2^{t(n)-1}$-bit identity into $2^{t(n)-1}$ number of $t(n)$-bit identities — we, thereby, refer to this idea as the "log-n" trick. The protocol from [51] is based on the log-n trick and is described below.

The committer $\widetilde{C}$ and receiver $\widetilde{R}$ receive the security parameter 1^n and identity $\mathsf{id} \in \{0,1\}^{t'(n)}$ as common input where $t'(n) = 2^{t(n)-1}$. Furthermore, $\widetilde{C}$ gets a private input $v \in \{0,1\}^{\alpha}$ which is the value to be committed.

- *Commit stage:*

  1. To commit to a value $v \in \{0,1\}^{\alpha}$, $\widetilde{C}$ chooses t' random shares $r_0, r_1, \ldots, r_{t'-1} \in \{0,1\}^{\alpha}$ such that $v = r_0 \oplus r_1 \oplus \ldots \oplus r_{t'-1}$.
  2. For each $0 \leq i \leq t'-1$, $\widetilde{C}$ and $\widetilde{R}$ run $\langle \widehat{C}, \widehat{R} \rangle$ to commit to r_i (in parallel) using identity $(i, \mathsf{id}[i])$ where $\mathsf{id}[i]$ is the ith bit of id. Let d_i be the corresponding decommitment string.

  Let c_i be the transcript of $\langle \widehat{C}, \widehat{R} \rangle$ committing to r_i with identity $(i, \mathsf{id}[i])$. Then we denote by $c = \{c_i\}_{0 \leq i \leq t'-1}$ the entire transcript of the interaction.

- *Reveal stage:*

  On receiving the decommitment $(v, \{r_i\}_i, \{d_i\}_i)$, $\widetilde{R}$ verifies (1) for each $0 \leq i \leq t'-1$,

c_i is a commitment to r_i using $\langle \widehat{C}, \widehat{R} \rangle$ and identity $(i, \mathsf{id}[i])$, and (2) $v = r_0 \oplus r_1 \oplus \ldots \oplus r_{t'-1}$. $\widetilde{R}$ accepts the decommitment iff (1) and (2) hold.

Furthermore, let us assume that $\langle \widehat{C}, \widehat{R} \rangle$ is over-extractable w.r.t. extractor $\widehat{o\mathcal{E}}_{\mathsf{NM}}$ then we construct an extractor $\widetilde{o\mathcal{E}}_{\mathsf{NM}}$ for $\langle \tilde{C}, \tilde{R} \rangle$ as follows,

- Extraction - Algorithm $\widetilde{o\mathcal{E}}_{\mathsf{NM}}$:

  On receiving $\mathsf{id} \in \{0,1\}^{t'}$ and commitment $c = \{c_i\}_{0 \leq i \leq t'-1}$, $\widetilde{o\mathcal{E}}_{\mathsf{NM}}$ runs $\widehat{o\mathcal{E}}_{\mathsf{NM}}$ on each c_i obtaining output r'_i. If any of the r'_i is $\perp$ then $\widetilde{o\mathcal{E}}_{\mathsf{NM}}$ outputs $\perp$. Otherwise, it outputs $v' = r'_0 \oplus r'_2 \oplus \ldots \oplus r'_{t'-1}$ as the extracted value.

Theorem 24 (Log-n trick [51]) *Let t be such that $t'(n) = 2^{t(n)-1}$ is a polynomial. Let $\langle \widehat{C}, \widehat{R} \rangle$ be a commitment scheme and $\mathcal{C}$ be a class of circuits that is closed under composition with $\mathcal{P}/\mathrm{poly}$.*

1. *If $\langle \widehat{C}, \widehat{R} \rangle$ is a tag based perfectly binding commitment scheme for $t(n)$-bit identities then $\langle \widetilde{C}, \widetilde{R} \rangle$ is a tag based perfectly binding commitment scheme for identities of length $t'(n) = 2^{t(n)-1}$ bits.*

2. *If $\langle \widehat{C}, \widehat{R} \rangle$ is concurrent $\mathcal{C}$-non-malleable w.r.t. commitment then $\langle \widetilde{C}, \widetilde{R} \rangle$ is one-one $\mathcal{C}$-non-malleable w.r.t. commitment.*

3. *If $\langle \widehat{C}, \widehat{R} \rangle$ is (d, S)-over-extractable by $\widehat{o\mathcal{E}}_{\mathsf{NM}}$ then $\langle \widetilde{C}, \widetilde{R} \rangle$ is (d, S)-over-extractable by $\widetilde{o\mathcal{E}}_{\mathsf{NM}}$. Furthermore, if $\langle \widehat{C}, \widehat{R} \rangle$ is concurrent $\mathcal{C}$-non-malleable w.r.t. extraction by $\widehat{o\mathcal{E}}_{\mathsf{NM}}$ then $\langle \widetilde{C}, \widetilde{R} \rangle$ is standalone $\mathcal{C}$-non-malleable w.r.t. extraction by $\widetilde{o\mathcal{E}}_{\mathsf{NM}}$.*

Proof: We prove each of the above in the following:

- Perfect binding and tag lengths: The perfect binding of $\langle \widetilde{C}, \widetilde{R} \rangle$ follows from the statistical binding of $\langle \widehat{C}, \widehat{R} \rangle$. Furthermore, $\langle \widetilde{C}, \widetilde{R} \rangle$ as defined above accomodates identities of length $t' = 2^{t(n)-1}$-bits.

- Non-malleability w.r.t. extraction: Assume for contradiction that there exists a non-uniform attacker $A = \{A_n\}_{n\in\mathbb{N}} \in \mathcal{C}$ that participates in one left with $\widetilde{C}$ and one right interaction with $\widetilde{R}$ sending/receiving commitments to values of length $\alpha = \mathsf{poly}(n)$-bits, a non-uniform distinguisher $D = \{D_n\}_{n\in\mathbb{N}} \in \mathcal{P}/\text{poly}$ and a polynomial $p(\cdot)$ such that for infinitely many $n \in \mathbb{N}$,

$$\left|\mathsf{Pr}[D_n(\mathsf{emim}^{A_n}_{\langle\widetilde{C},\widetilde{R}\rangle}(1^n,0)) = 1] - \mathsf{Pr}[D_n(\mathsf{emim}^{A_n}_{\langle\widetilde{C},\widetilde{R}\rangle}(1^n,1)) = 1]\right| \geq 1/p(n) \ ,$$

where $\mathsf{emim}^{A}_{\langle\widetilde{C},\widetilde{R}\rangle}(1^n,b)$ describes the view of A, and value $\tilde{v}'$ extracted from the right commitment $\widetilde{c} = \{\widetilde{c}_i\}_{0\leq i\leq t'-1}$ by extractor $\widetilde{o\mathcal{E}}_{\mathsf{NM}}$. Recall that $\tilde{v}'$ is set to $\perp$ when $\mathsf{id} = \tilde{\mathsf{id}}$ for A's choice of left and right identities id and $\tilde{\mathsf{id}}$ respectively. When $\mathsf{id} \neq \tilde{\mathsf{id}}$, by the definition of extractor $\widetilde{o\mathcal{E}}_{\mathsf{NM}}$, $\tilde{v}' = \tilde{v}'_0 \oplus \ldots \oplus \tilde{v}'_{t'-1}$ where $\tilde{v}'_i$ are the values extracted from $\tilde{c}_i$ by extractor $\widehat{o\mathcal{E}}_{\mathsf{NM}}$.

Next, we construct a one-many non-uniform adversary $A' = \{A'_n\}_{n\in\mathbb{N}}$, and a non-uniform distinguisher $D' = \{D'_n\}_{n\in\mathbb{N}}$ such that for infinitely many $n \in \mathbb{N}$

$$\left|\mathsf{Pr}[D'_n(\mathsf{emim}^{A'}_{\langle\widehat{C},\widehat{R}\rangle}(1^n,0)) = 1] - \mathsf{Pr}[D'_n(\mathsf{emim}^{A'}_{\langle\widehat{C},\widehat{R}\rangle}(1^n,1)) = 1]\right| \geq 1/(p(n)\cdot t').$$

The adversary A' internally runs A, participates in one left interaction with $\widehat{C}$ and $m = t'(n)$ right interactions with $\widehat{R}$ and internally emulates an execution of $\mathsf{IND}^{A}_{\langle\widetilde{C},\widetilde{R}\rangle}(b)$ for A as follows:

  - Step 1: For the right interaction with A, A' emulates the honest receiver $\widetilde{R}$ using its $t'(n)$ right interactions with $\widehat{R}$, by simply forwarding messages between A and $\widehat{R}$. A' waits for A to select identity for its left interaction. Let id be the $t'(n)$-bit identity and v_0, v_1 be the values sent by A for the left interaction. Let $\mathsf{s}_i = (i, \mathsf{id}[i])$ for $0 \leq i \leq t'-1$.
  - Step 2: To continue with the left interaction, A' samples a random $j \xleftarrow{\$} \{0,\ldots,t'-1\}$. Let $I = \{0,\ldots,t'-1\} \setminus \{j\}$. A' samples random shares

$r_i \xleftarrow{\$} \{0,1\}^\alpha$ for $i \in I$ and sets $u_b = v_b \oplus r$ where $r = \oplus_{i \in I}\, r_i$. Then, A' begins its left interaction with $\widehat{C}$ with identity $\mathsf{s} = \mathsf{s}_j$ and challenges values u_0, u_1.

– Step 3: A' interacts with A acting as a honest committer $\widetilde{C}$ to compute the commitment $c = \{c_i\}_{0 \leq i \leq t'-1}$. More precisely, for all $i \in I$, A acts as the honest committer $\widehat{C}$ to generate the commitment c_i to value r_i under identity s_i. The commitment c_j is the commitment to value u_b under identity $\mathsf{s} = \mathsf{s}_j$ generated by forwarding messages between A and the external committer $\widehat{C}$. It is easy to see that if $\widehat{C}$ commits to u_b then c is a commitment to v_b under $\langle \widetilde{C}, \widetilde{R} \rangle$ with identity id.

The distinguisher D' with input the view of A' and the values $(\tilde{v}'_1, \ldots, \tilde{v}'_{t'})$ extracted by extractor $\widehat{o\mathcal{E}}_{\mathsf{NM}}$ from the t' right commitments of A', runs as follows:

– Step 1: D' reconstructs the view of A in emulation by A'. Furthermore, let id,$\tilde{\mathsf{id}}$ be the identities chosen by A for its left and right interactions (defined by the view of A) respectively and let $\mathsf{s} = (j, \mathsf{id}[j])$ be the identity chosen by A' for some $0 \leq j \leq t' - 1$. And let $\tilde{\mathsf{s}}_i = (i, \tilde{\mathsf{id}}[i])$ for all $0 \leq i \leq t' - 1$.

– Step 2: If $\mathsf{id} \neq \tilde{\mathsf{id}}$ but $\mathsf{s} = \tilde{\mathsf{s}}_j$ for some $j \in \{0, \ldots, t' - 1\}$ then D' aborts.[20] Otherwise, D' sets $\tilde{v}' = \oplus_{0 \leq i \leq t'-1}\, v'_i$.

– Step 2: D' then runs D on the above reconstructed view of A and $\tilde{v}'$ and returns whatever D' returns.

First, observe that whenever $\widehat{C}$ commits to u_b, A' perfectly simulates the MIM experiment $\mathsf{MIM}^{A}_{\langle \widetilde{C}, \widetilde{R} \rangle}(b)$ for A. Conditioned on D' not aborting, we know that A' choice of left identity s is distinct from all right identities $\tilde{\mathsf{s}}_j$. Therefore, by

[20] This is because, the value $\tilde{v}'_j$ given as input to D' will be replaced with $\perp$ disallowing D' to reconstruct the input to D.

definition of $\mathsf{emim}^{A'}_{\langle\widehat{C},\widehat{R}\rangle}$, D''s inputs $\tilde{v}'_i$ are the values extracted by the extractor $\widehat{o\mathcal{E}}_{\mathsf{NM}}$ from the i-th right commitment $\tilde{c}_i$. Therefore, the value $\tilde{v}'$ reconstructed by D' is identical to the value extracted by $\widetilde{o\mathcal{E}}_{\mathsf{NM}}$ from A's right commitment $\tilde{c} = \{\tilde{c}_i\}_{0\leq i\leq t'-1}$. Therefore, conditioned on not aborting, D' perfectly reconstructs $\mathsf{emim}^{A}_{\langle\widetilde{C},\widetilde{R}\rangle}(b)$ from its input $\mathsf{emim}^{A'}_{\langle\widehat{C},\widehat{R}\rangle}(b)$. Since D distinguishes $\mathsf{emim}^{A}_{\langle\widetilde{C},\widetilde{R}\rangle}(0)$ from $\mathsf{emim}^{A}_{\langle\widetilde{C},\widetilde{R}\rangle}(1)$ with advantage at least $1/p(n)$ and D' does not abort with probability $1/t'(n)$, we have (A', D') break the one-many non-malleability w.r.t. extraction of $\langle\widehat{C},\widehat{R}\rangle$ with advantage $1/p(n)\cdot t'(n)$. Furthermore, note that A' internally runs A and the rest of the computation takes $\mathsf{poly}(n)$-time. Also, D' internally runs D and the rest of its computation also takes $\mathsf{poly}(n)$-time. Therefore, since $A \in \mathcal{C}$ and $\mathcal{C}$ is closed under composition with $\mathcal{P}/\mathrm{poly}$, we have $A' \in \mathcal{C}$. Also, $D \in \mathcal{P}/\mathrm{poly}$ implies that $D' \in \mathcal{P}/\mathrm{poly}$. This contradicts the one-many non-malleability of $\langle\widehat{C},\widehat{R}\rangle$ w.r.t. extraction by $\widehat{o\mathcal{E}}_{\mathsf{NM}}$.

Remark 15 *We note that the $1/t'$ loss in the advantage of the reduction can be avoided if A sends the identity of the right interaction $\tilde{\mathsf{id}}$ before sending id. In this case, whenenver $\mathsf{id} \neq \tilde{\mathsf{id}}$ there exists at least one index $0 \leq j \leq t'-1$ such that $\mathsf{id}[j] \neq \tilde{\mathsf{id}}[j]$ and hence $\mathsf{s}_j = (j, \mathsf{id}[j])$ is distinct from all $\tilde{\mathsf{s}}_i = (i, \tilde{\mathsf{id}}[i])$. This ensures D' never aborts. However, since we allow our MIM adversary A total control over the scheduling of messages (even choosing identities), given the left identity id we can only guess the special index j thereby inccurring a $1/t'$ loss in the advantage where $t' = |\mathsf{id}|$.*

- Non-malleability w.r.t. commitment: This follows syntactically from the same proof as Non-malleability w.r.t. extraction by replacing emim random variables with their respective mim random variables. We skip the formal proof.

- Over-extractability: A valid commitment $c = \{c_i\}_{0 \leq i \leq t'-1}$ is such that every c_i is a valid commitment for $\langle \widehat{C}, \widehat{R} \rangle$. Due to the over-extractability of $\langle \widehat{C}, \widehat{R} \rangle$ w.r.t. $\widehat{o\mathcal{E}}_{\mathsf{NM}}$, for every $0 \leq i \leq t' - 1$, the extractor $\widetilde{o\mathcal{E}}_{\mathsf{NM}}$ always extracts the correct value r'_i. Therefore, $\widetilde{o\mathcal{E}}_{\mathsf{NM}}$ always extracts the correct value from c. Since, t' is a polynomial, $\widetilde{o\mathcal{E}}_{\mathsf{NM}}$ fails with negligible probability. Moreover, $\widetilde{o\mathcal{E}}_{\mathsf{NM}}$ runs $\widehat{o\mathcal{E}}_{\mathsf{NM}}$ on t' commitments and rest of the computation takes $\mathsf{poly}(n)$ time. Therefore, if $\widehat{o\mathcal{E}}_{\mathsf{NM}} \in \mathcal{C}^{\wedge}_{d,S}$ then $\widetilde{o\mathcal{E}}_{\mathsf{NM}} \in \mathcal{C}^{\wedge}_{d,S}$. Therefore, $\langle \tilde{C}, \tilde{R} \rangle$ is (d, S)-over-extractable w.r.t. $\widetilde{o\mathcal{E}}_{\mathsf{NM}}$.

■

5.7 Concurrent Non-malleable Commitment for n-bit Identities

In this section, we construct a concurrent non-malleable commitment scheme $\langle C^*, R^* \rangle$ that can accomodate n-bit identities. This then concludes the proof of Theorem 25. The idea is to start with the basic commitment scheme from Section 5.4 that is one-one non-malleable w.r.t. extraction for short identities, say $t(n)$-bits. Then apply the non-malleability strengthening technique described in Section 5.5.3 followed by the log-n trick [51] described in Section 5.6 repeatedly until the length of the identities reaches n-bits. The resulting commitment scheme is the commitment scheme $\langle C^*, R^* \rangle$. We detail the construction of $\langle C^*, R^* \rangle$ more formally in Section 5.7.1, provide instantiations in Section 5.7.2, discuss the efficiency of the scheme $\langle C^*, R^* \rangle$ in Section 5.7.3 and argue about the security $\langle C^*, R^* \rangle$ in Section 5.7.4.

Theorem 25 *For some sub-exponential functions T, B assume the existence of $\mathcal{C}^{\wedge}_{B,B}$-secure injective one-way functions, $\mathcal{C}^{\wedge}_{B,B}$-WI ZAP, $\mathcal{C}^{\wedge}_{B,B}$-collison-resistant hash function*

family and (T, B)-secure Time-lock puzzles. Then, $\langle C^*, R^* \rangle$ is a 2-round, perfectly binding concurrent non-malleable w.r.t. extraction and w.r.t. commitment against poly-size adversaries.

5.7.1 Commitment Scheme $\langle C^*, R^* \rangle$

We formally describe the construction of $\langle C^*, R^* \rangle$ that is concurrent non-malleable w.r.t. commitment (and extraction) for n-bit identities. As mentioned above we initially start with a commitment scheme $\langle C^0, R^0 \rangle$ for $t(n)$-bit identities and apply the non-malleability strengthening and log-n trick repeatedly, for say $r(n)$ times, until we reach identities of length n-bits.

- Initial Scheme $\langle C^0, R^0 \rangle$:

  The initial scheme $\langle C^0, R^0 \rangle$ is the basic scheme $(\mathsf{ENMCom}, \mathsf{ENMOpen})$, as constructed in Section 5.4, that is one-one non-malleable w.r.t. extraction for identities of length $\mathsf{id}^0(n) = t(n)$-bits. Furthermore, let $\langle C^0, R^0 \rangle$ be non-malleable against circuits of depth at most $\mathsf{poly}(S^0)$ and size at most $\mathsf{poly}(S^0)$ and extractable by an extractor of depth $\mathsf{poly}(S'^0)$ and size $\mathsf{poly}(S'^0)$.[21]

- Identity Amplification Step for $r(n)$ Times:

  Next, we repeatedly apply the following two steps $r(n)$ number of times. Let $\langle C^{j-1}, R^{j-1} \rangle$ be the commitment scheme at the end of the $j-1$-st iteration for $j \in [r(n)]$. We describe below the j-th iteration below. Let $\langle C^{j-1}, R^{j-1} \rangle$ be one-one non-malleable w.r.t. commitment (and extraction) for identities of length $\mathsf{id}^{j-1}(n)$-bits. Furthermore, let $\langle C^{j-1}, R^{j-1} \rangle$ be non-malleable against circuits of depth at

[21] Note that the initial scheme as presented in Section 5.4 is non-malleable against circuits of depth at most $\mathsf{poly}(d_0)$ and size at most $\mathsf{poly}(S_0)$ where $d_0 << S_0$. However, note that such a scheme is still non-malleable against circuits of depth at most $\mathsf{poly}(d_0)$ and size at most $\mathsf{poly}(d_0)$.

most $\mathsf{poly}(S^{j-1})$ and size at most $\mathsf{poly}(S^{j-1})$ and extractable by an extractor of depth $\mathsf{poly}(S'^{j-1})$ and size $\mathsf{poly}(S'^{j-1})$.

1. Non-malleability Strengthening Technique:

   First, using an appropriate hierarchy of functions as described in Eq (5.9), we apply the non-malleability strengthening technique to the scheme $\langle C^{j-1}, R^{j-1}\rangle$ to boost its one-one non-malleability to concurrent non-malleability. The resulting scheme $\langle \widehat{C}^j, \widehat{R}^j\rangle$ is concurrent non-malleable w.r.t. commitment (and extraction) for identities of length $\mathsf{id}^{j-1}(n)$-bits.

2. Log-n Trick:

   Second, we apply the log-n trick to the concurrent non-malleable scheme $\langle \widehat{C}^j, \widehat{R}^j\rangle$ to construct a one-one non-malleable commitment $\langle C^j, R^j\rangle$ for identities of length $\mathsf{id}^j(n)$ such that $\mathsf{id}^j(n) = 2^{\mathsf{id}^{j-1}(n)-1}$.

- Final Scheme $\langle C^*, R^*\rangle$:

  The commitment scheme $\langle C^{r(n)}, R^{r(n)}\rangle$ constructed at the end of $r(n)$ iterations is one-one non-malleable for identities of length $\mathsf{id}^{r(n)}$. We apply the non-malleability strengthening technique one more time to $\langle C^{r(n)}, R^{r(n)}\rangle$ to boost its one-one non-malleability to concurrent non-malleability. The resulting scheme $\langle C^*, R^*\rangle$ is concurrent non-malleable for identities of length $\mathsf{id}^{r(n)}(n)$-bits.

Note that we begin we identities of length $\mathsf{id}^0 = t(n)$ and identities in successive iterations satisfy the following,

$$\mathsf{id}^j(n) = 2^{\mathsf{id}^{j-1}(n)-1} .$$

Then it is easy to see that for $\mathsf{id}^{r(n)}(n) \geq n$ and $t(n) > 2$, we need to apply the identity amplification step $r(n) = O(\log^* n - \log^* t(n))$ times.

5.7.2 Instantiations

The initial scheme constructed in Section 5.4 and the identity amplification step described in Sections 5.5.3,5.6 require a family of depth-robust and size-robust commitment schemes, and a family of non-uniform collision resistant hash functions which are based on some hierarchy of non-decreasing functions. Below we detail the size of this hierarchy required for constructing $\langle C^*, R^* \rangle$ from the initial scheme $\langle C^0, R^0 \rangle$ for $t(n)$-bit identities and $r(n)$ iterations of the identity amplification step. Then we give instantiations of this hierarchy firstly from sub-exponential security and then from the strictly weaker sub-subexponential security.

Initial Scheme $\langle C^0, R^0 \rangle$. We start with the basic scheme $(\mathsf{ENMCom}, \mathsf{ENMOpen})$ for $t(n)$-bit identities. As described in Section 5.4, the construction of the scheme $(\mathsf{ENMCom}, \mathsf{ENMOpen})$ for $t(n)$-bit identities requires a family of $2^{t(n)}$ size-robust and depth-robust commitment schemes w.r.t. the following hierarchy of non-decreasing functions,

$$n << d_0 << d_1 << \ldots << d_{l-1} << d_l << S_0 << S_1 << \ldots << S_{l-1} << S_l \ ,$$

where $l = 2^{t(n)}$ such that for every $i \in \{0,1\}^{t(n)}$,

- there exists a depth-robust commitment scheme $(\mathsf{ECom}_{d_i}, \mathsf{EOpen}_{d_i})$ that is $\mathcal{C}_{d_i}$-hiding and (d_{i+1}, d_{i+1})-over-extractable w.r.t. an extractor $o\mathcal{E}_{d_i}$.

- there exists a size-robust commitment scheme $(\mathsf{ECom}_{S_i}, \mathsf{EOpen}_{S_i})$ that is $\mathcal{C}^{\wedge}_{S_i,S_i}$-hiding and $(\mathsf{poly}(n), S_{i+1})$-over-extractable w.r.t. an extractor $o\mathcal{E}_{S_i}$.

Therefore, to construct the initial commitment scheme we need a hierarchy of $2(l+1) = 2(2^{t(n)}+1)$ non-decreasing functions.

Identity Amplification Step. Consider the $j+1$-st iteration of the identity amplification step described in the construction of $\langle C^*, R^* \rangle$. In the $j+1$-st iteration, we are applying the strengthening technique to the commitment scheme $\langle C^j, R^j \rangle$ which is $\mathcal{C}^{\wedge}_{S^j, S^j}$-non-malleable and extractable by a circuit of size $\mathsf{poly}(S'^j)$. The strengthening technique requries a family of four depth-robust [22] and four size-robust commitment schemes. Furthermore, it also requires a family of non-uniform collision-resistant hash functions w.r.t. the following hierarchy of non-decreasing functions,

$$\begin{aligned} &n << d_4^j << d_3^j << d_1^j << d_2^j << S_2^j << S_1^j << S_{\mathsf{CRH}}^j << \\ &S_{\mathsf{CRH}}'^j << S^j << S'^j << S_3^j << S_4^j << S_4'^j << S^* \ , \end{aligned} \tag{5.14}$$

such that,

- $(\mathsf{ECom}_1, \mathsf{EOpen}_1)$ is a perfectly binding commitment scheme which is $\mathcal{C}^{\vee}_{d_1^j, S_1^j}$-hiding and $(d_2^j, S_{\mathsf{CRH}}^j)$-over-extractable w.r.t. extractor $o\mathcal{E}_1$.

- $(\mathsf{ECom}_2, \mathsf{EOpen}_2)$ is a perfectly binding commitment scheme which is $\mathcal{C}^{\vee}_{d_2^j, S_2^j}$-hiding and (S_2^j, S_1^j)-over-extractable w.r.t. extractor $o\mathcal{E}_2$.

- $(\mathsf{ECom}_3, \mathsf{EOpen}_3)$ is a perfectly binding commitment scheme which is $\mathcal{C}^{\vee}_{d_3^j, S_3^j}$-hiding and $(d_1^j, S_4'^j)$-over-extractable w.r.t. extractor $o\mathcal{E}_3$.

- $(\mathsf{ECom}_4, \mathsf{ECom}_4)$ is a perfectly binding commitment scheme which is $\mathcal{C}^{\vee}_{d_4^j, S_4^j}$-hiding and $(d_3^j, S_4'^j)$-over-extractable w.r.t. extractor $o\mathcal{E}_4$.

- $\mathcal{H} = \{H_n\}_{n \in \mathbb{N}}$ is a $\mathcal{C}^{\wedge}_{S_{\mathsf{CRH}}^j, S_{\mathsf{CRH}}^j}$-collision-resistant family of hash functions such that a collision can be found by a circuit of size $\mathsf{poly}(S_{\mathsf{CRH}}'^j)$.

Furthermore, we apply the log-n trick to the resulting commitment scheme. Note that the log-n trick does not rely on any additional tools. Therefore, in an iteration of the

[22] Note that the transformation actually requires four depth-and-size robust commitment schemes but as described in Section 5.3.3 depth-and-size robust commitment scheme can be constructed from a single depth-robust and a size-robust commitment scheme.

identity amplification step, we need four depth-robust[22], four size-robust commitment schemes and a hash function family. In other words, we need an additional at most eleven[23] non-decreasing functions per iteration. Therefore, over $r(n)$ iterations, we will need a hierarchy of $11r(n) + 11$ functions.[24]

Therefore, to construct the commitment scheme $\langle C^*, R^* \rangle$ from $\langle C^0, R^0 \rangle$ for $t(n)$-bit identities, we need a hierarchy of $L = 2^{t(n)+1}+11r(n)+13$ non-decreasing functions, where $r(n) = O(\log^* n - \log^* t(n))$. Furthermore, L is minimized when $t(n) = O(1)$, implying $r(n) = O(\log^* n)$ and $L = O(\log^* n)$. Next, we show two approaches to instantiate a hierarchy of $L = O(\log^* n)$ non-decreasing functions, one from sub-exponential security and another from sub-subexponential security.

Instantiation from Sub-exponential Security. As mentioned above, we need to instantiate a hierarchy of L non-decreasing functions for constructing $\langle C^*, R^* \rangle$. Let the required hierarchy be the following,

$$p_1 << p_2 << \ldots << p_L \ . \tag{5.15}$$

Let $\mathcal{F}(\lambda)$ be some non-decreasing, invertible function defined on $\mathbb{N}$ such that $\mathcal{F}(\lambda) = \omega(\log \lambda)$ but $\mathcal{F}(\lambda) = o(\lambda)$. It is easy to see that $\mathcal{F}(\lambda) = \lambda^\varepsilon$ satisfies the requirements for any $0 < \varepsilon < 1$. First we will instantiate the hierarchy (Equation 5.15) based on the existence of $2^{\mathcal{F}(\lambda)}$-secure primitives and then provide concrete parameters for the special case of sub-exponential security, that is, for $\mathcal{F}(\lambda) = \lambda^\varepsilon$ for some $\varepsilon < 1$.

Towards this, first assume the existence of $(T(t) = 2^{\mathcal{F}(t)}, B(n) = 2^{\mathcal{F}(n)})$-secure TL puzzle, $2^{\mathcal{F}(k)}$-secure injective OWF, $2^{\mathcal{F}(\theta)}$-collision-resistant hash family where $(n, t), k, \theta$ are

[23] nine levels are required for the four depth-and size-robust commitment schemes (see Equation 5.14) namely $d_1, \ldots, d_4$, $S_1, \ldots, S_4, S_4'$ and additional two levels namely S_{CRH} and S'_{CRH} for the collision-resistant hash function.

[24] The additional eleven functions is due an extra application of the non-malleability strengthening to boost the non-malleability of $\langle C^{r(n)}, R^{r(n)} \rangle$.

security parameters for the underlying TL puzzle, injective OWF and collision-resistant hash respectively. We instantiate the above hierarchy, that is, p_1 through p_L from $2^{\mathcal{F}(\lambda)}$-security by varying the security parameter λ. Let n be the security parameter of the non-malleable commitment scheme we want to construct. Then, consider the following sequence of security parameters

$$n_0 \ , n_1 \ , \ldots \ , n_L \ ,$$

where each n_i is some function of n (we specify these shortly). We set i-th level (i.e., p_i) in the required hierarchy as,

$$p_i = 2^{\mathcal{F}(n_i)} \ .$$

We expect the functions in the hierarchy to satisfy certain constraints in order for us to be able to instantiate the required depth-robust, size-robust commitment schemes, and collision-resistant hash function from them. We list the properties below.

1. Since we expect all our primitives to be secure against any poly-sized circuit, we require that the first security parameter n_0 be such that $2^{\mathcal{F}(n_0)} \geq 2^{\omega(\log n)}$ that is,

$$\mathcal{F}(n_0) = \omega(\log n) \ ,$$
$$n_0 = \mathcal{F}^{-1}(\omega(\log n)) \ .$$

2. For any i, we need to be able to instantiate the following primitives,

   (a) (p_i, p_{i+1})-depth-robust commitment scheme: Commitment scheme that is $\mathcal{C}p_i$-hiding but (p_{i+1}, p_{i+1})-over-extractable. We instantiate such a scheme from TL puzzles with security parameter $t(n) = n_i(n)$.[25] Then by the $2^{\mathcal{F}(t)}$-security of TL puzzles combined with $t(n) = n_i(n)$ (or equivalently $2^{\mathcal{F}(t)} = 2^{\mathcal{F}(n_i)} = p_i$),

[25] Recall that TL puzzles have two security parameters n and t. The security parameter n is the security parameter of the non-malleable commitment scheme. Therefore, we sample puzzles from the support of $\mathsf{Gen}(1^n, 1^{n_i}, \cdot)$ in the depth-robust commitment scheme in Section 5.3.1.

the resulting puzzles are hard for adversaries in $\mathcal{C}p_i$. To guarantee that the puzzles can be solved by some circuit of size $\mathsf{poly}(p_{i+1})$, we require that

$$2^{t(n)} = 2^{n_i(n)} \leq p_{i+1} \ . \tag{5.16}$$

If Equation 5.16 holds then by Theorem 16, we have a (p_i, p_{i+1})-depth-robust-commitment scheme.

(b) (p_i, p_{i+1})-size-robust commitment scheme: A commitment scheme that is $\mathcal{C}^{\wedge}_{p_i,p_i}$-hiding but $(\mathsf{poly}(n), p_{i+1})$-over-extractable. We instantiate such a scheme from $2^{\mathcal{F}(k)}$-secure injective OWF on input-length $k(n) = n_i(n)$. Then, the $2^{\mathcal{F}(k)}$-security guarantees that the resulting OWF (one with security parameter $k = n_i$) is hard to invert for adversaries in $\mathcal{C}^{\wedge}_{p_i,p_i}$. Furthermore, such a function can be inverted by a circuit of size $\mathsf{poly}(n_i) \cdot 2^{n_i}$ and depth $\mathsf{poly}(n_i)$. Therefore, to guarantee that function can be inverted by a circuit of size $\mathcal{C}^{\wedge}_{\mathsf{poly}(n),p_{i+1}}$, we require that,

$$n_i \leq \mathsf{poly}(n) \ \ ; \ \ 2^{n_i} \leq p_{i+1} \ . \tag{5.17}$$

If Equation 5.17 holds then by Theorem 17, we have a (p_i, p_{i+1})-size-robust-commitment scheme.

(c) (p_i, p_{i+1})-collision-resistant hash function family: A (p_i, p_{i+1})-collision-resistant hash function family is a family of hash functions that is $\mathcal{C}^{\wedge}_{p_i,p_i}$-collision resistant and for which there exists a circuit of size $\mathsf{poly}(p_{i+1})$ that finds collisions with probability 1. We instantiate such a family by setting the security parameter θ of $\mathcal{H}$ as $\theta(n) = n_i(n)$, where $\mathcal{H}$ is a family of $2^{\mathcal{F}(\theta)}$-collision-resistant hash functions. As discussed above, the $2^{\mathcal{F}(\theta)}$-collision resistance of $\mathcal{H}$ implies that the resulting function is $\mathcal{C}^{\wedge}_{p_i,p_i}$-collision resistant. To guarantee that a

circuit of size $\mathsf{poly}(p_{i+1})$ finds collisions, we require that

$$2^{n_i} \leq p_{i+1} \ . \tag{5.18}$$

Setting $n_{i+1} = \mathcal{F}^{-1}(n_i)$ implies $p_{i+1} = 2^{\mathcal{F}(n_{i+1})} = 2^{n_i}$ which guarantees that Equations 5.16, 5.17, 5.18 hold. This entails a sequence $n_1, \ldots, n_L$ where the i-th security parameter n_i is,

$$n_i = \left(\mathcal{F}^{-1}\right)^{i+1} (\omega(\log n)) \ .$$

3. Finally we require that the last security parameter n_L be upper-bounded by some $\mathsf{poly}(n)$,

$$n_L = \left(\mathcal{F}^{-1}\right)^{L+1} (\omega(\log n)) \leq \mathsf{poly}(n) \ . \tag{5.19}$$

Now let us consider the case of sub-exponential security, that is, let $\mathcal{F} = \lambda^{\varepsilon}$ for some $0 < \varepsilon < 1/2$. Then, $\mathcal{F}^{-1}(y) = y^{1/\varepsilon}$ be the inverse of $\mathcal{F}$. For the last security level n_L to be polynomially bounded, we require that,

$$(\omega(\log n))^{(1/\varepsilon)^{L+1}} \leq \mathsf{poly}(n) \ .$$

It is easy to see that from subexponential security, we can derive $L = \Theta(\log \log n)$ levels. Recall that to construct $\langle C^*, R^* \rangle$ we need $O(\log^* n)$ levels in the hierarchy, hence the above hierarchy finds an instantiation from subexponential security.

However, for our transformation, we require only $L = O(\log^* n)$ levels which is significantly less than $\Theta(\log \log n)$ levels that can be extracted from sub-exponential security. Hence, there is hope to instantiate the hierarchy from weaker than sub-exponential security. In fact, such a hierarchy can, indeed, be instantiated from strictly weaker security — *sub-subexponential* security — which we show below.

Instantiation from Sub-subexponential Security. First we define the notion of sub-subexponential security and then provide an instantiation of the hierarchy. Informally, a $2^{\mathcal{F}(\lambda)}$-secure primitive is sub-subexponential -secure if

$$\mathcal{F}(\lambda) \in \lambda^{o(1)} \ .$$

A candidate for $\mathcal{F}$ for sub-subexponential security is the following,

$$\mathcal{F}(\lambda) = \lambda^{\frac{1}{\mathcal{X}(\lambda)}} \ ,$$

where $\mathcal{X}(\lambda) = \omega(1)$ be some non-decreasing function on $\mathbb{N}$.

We ask how large (if at all) such an $\mathcal{X}(\lambda) = \omega(1)$ can be so that we can still instantiate the above hierarchy. The only point of concern is bounding the security parameter n_L of the last level, that is, we ask how large $\mathcal{X}(\lambda)$ be such that for $\mathcal{F}(\lambda) = \lambda^{\frac{1}{\mathcal{X}(\lambda)}}$ and $L = O(\log^* n)$ the following holds,

$$n_L = \left(\mathcal{F}^{-1}\right)^L (\omega(\log n)) \leq \mathsf{poly}(n) \ .$$

However the above closed form is hard to analyse so we restrict the right hand side to be n instead of a generic $\mathsf{poly}(n)$, that is,

$$\left(\mathcal{F}^{-1}\right)^L (\omega(\log n)) \leq n \tag{5.20}$$

Applying $\mathcal{F}$ on both sides we get,

$$\left(\mathcal{F}^{-1}\right)^{L-1} (\omega(\log n)) \leq \mathcal{F}(n) \ , \tag{5.21}$$

Let $n' = \mathcal{F}(n) = n^{\frac{1}{\mathcal{X}(n)}} < n$. We have,

$$\mathcal{F}(n') = (n')^{\frac{1}{\mathcal{X}(n')}} = (\mathcal{F}(n))^{\frac{1}{\mathcal{X}(n')}} \ .$$

Since $\mathcal{X}$ is a non-decreasing function we have,

$$\mathcal{F}(n') = (\mathcal{F}(n))^{\frac{1}{\mathcal{X}(n')}} > (\mathcal{F}(n))^{\frac{1}{\mathcal{X}(n)}} \ , \tag{5.22}$$

Applying again $\mathcal{F}$ on both sides of Equation (5.21),

$$\left(\mathcal{F}^{-1}\right)^{L-2}(\omega(\log n)) \leq \mathcal{F}(n') \ , \tag{5.23}$$

Therefore by Equation (5.22) we know that as long as the following holds, Equation (5.23) holds.

$$\left(\mathcal{F}^{-1}\right)^{L-2}(\omega(\log n)) \leq \mathcal{F}(n)^{\frac{1}{\mathcal{X}(n)}} = n^{\frac{1}{\mathcal{X}(n)}^2} \ .$$

After repeatedly applying $\mathcal{F}$, it is easy to see that as long as the following holds, Equation (5.20) holds.

$$\omega(\log n) \leq n^{\frac{1}{\mathcal{X}(n)^L}} \ .$$

Furthermore, the if the following holds then the above Equation holds,

$$\mathcal{X}(n)^L \leq \frac{\log n}{\omega(\log\log n)}$$

$$\mathcal{X}(n) \leq \left(\frac{\log n}{\omega(\log\log n)}\right)^{\frac{1}{O(\log^* n)}}$$

Finally, as long as the following holds for some $c > 0$ then Equation (5.20) holds.

$$\mathcal{X}(n) \leq (\log^c n)^{\frac{1}{O(\log^* n)}}$$

$$\mathcal{X}(n) \leq (\log n)^{\frac{1}{\Theta(\log^* n)}} \tag{5.24}$$

For $\mathcal{X}(n) = \log\log n$, it is easy to see that Equation (5.24) holds and hence Equation (5.20) holds. Therefore we can instantiate the above hierarchy from $2^{n^{\frac{1}{\log\log n}}}$-secure OWPs, TL puzzles and CRHs which is strictly weaker than assuming 2^{n^ε}-security.

5.7.3 Efficiency of $\langle C^*, R^*\rangle$

As described in Section 5.7.1, to construct the scheme $\langle C^*, R^*\rangle$ we apply the identity amplification step — non-malleability strengthening technique followed by the log-n trick

— $O(\log^* n)$ times. Suppose that the identity amplification step incurrs a polynomial overhead, that is, on input a scheme with computational complexity $\tau(n)$, it outputs a scheme with computational complexity $p(\tau(n))$ for some fixed polynomial p. Applying this step for a super-constant number of times leads to a scheme $\langle C^*, R^* \rangle$ with super-polynomial computational complexity.

Unfortunately, our non-malleability strengthening technique presented in Section 5.5 indeed incurrs polynomial overhead. Recall that on input a non-malleable commitment $\langle C, R \rangle$, the technique produces an output scheme $\langle \widehat{C}, \widehat{R} \rangle$ which uses ZAP to prove a statement that involves verifying the decommitment to a commitment of $\langle C, R \rangle$. Therefore, if the decommiment function $\mathsf{Open}(c, v, d)$ of $\langle C, R \rangle$ has complexity $\tau_{\mathsf{Open}}(n)$, the output scheme has complexity at least $p_{\mathsf{ZAP}}(\tau_{\mathsf{Open}}(n))$, where p_{ZAP} is the polynomial overhead induced by ZAP.

We show below that a simple modification can fix the problem. (We chose to present the strengthening technique in simpler terms earlier for ease of exposition.) Towards this, we introduce a new property called *open-decomposability* for commtiment schemes. We say that a scheme $\langle C, R \rangle$ is g-open-decomposable, if it is the case that, its decommitment function $\mathsf{Open}(c, v, d)$ can be decomposed into two functions of the following form:

- a "public" function $\mathsf{PubOpen}(c)$ that can be verified without the decommitment (v, d), and

- a "private" function $\mathsf{PrivOpen}(c^*, v, d)$ that depends on the decommitment and only a small part $c^* = \pi(c)$ of the commitment c, and takes polynomial time $g(n)$.

Open accepts iff both PubOpen and PrivOpen accept. Consider applying the non-malleability strengthening technique on such a g-open-decomposable commitment scheme. Instead of using ZAP to verify whether Open accepts, it is equivalent to verify whether PubOpen accepts in the clear (outside ZAP) and only verifies whether PrivOpen

accepts using ZAP. This simple change reduces the overhead induced by the ZAP proof from $p_{\mathsf{ZAP}}(\tau_{\mathsf{Open}}(n))$ to $p_{\mathsf{ZAP}}(g(n))$. Our key observation is that the initial non-malleable schemes, as well as all intermediate schemes produced throughout the iterations, are all open-decomposable w.r.t. small polynomials. Based on this, we can show that the complexity of the final scheme is polynomially bounded.

Open-decomposability. We formally define the notion of open-decomposability below.

Definition 33 (g-open-decomposability) *Let g be a polynomial. We say that a commitment scheme $\langle C, R\rangle$ is g-open-decomposable if there exist efficiently computable functions* PubOpen, PrivOpen, *and π, such that, for all $n \in \mathbb{N}$, $c \in \{0,1\}^{m(n)}$, $v \in \{0,1\}^{\alpha(n)}$, $d \in \{0,1\}^{l(n)}$ and $c^* = \pi(c)$,*

$$(\mathsf{Open}(c, v, d) = 1 \iff \mathsf{PubOpen}(c) = 1) \ \wedge \ (\mathsf{PrivOpen}(c^*, v, d) = 1)\,,$$

where PrivOpen *runs in time $g(n)$. Above, $m(n)$ and $l(n)$ are respectively the maximal lengths of commitments and decommitments generated using $\langle C, R\rangle$ for values of length $\alpha(n)$ with security parameter n.*

Using the above notion, we next describe the modified non-malleability strengthening technique and log-n trick. We analyze the open-decomposability property of the schemes produced by iteratively applying these two transformations to the initial schemes constructed in Section 5.4, and show that the growth of the complexity of these schemes is polynomially bounded.

More specifically, let g be a sufficiently large polynomial that, in particular, is larger than the complexity of all depth-and-size robust commtiment schemes, ECom's, used for constructing the initial schemes and in the transformations. By the analysis in Sec-

tion 5.7.2, all the ECom's used have polynomial complexity. This implies that the initial non-malleable commitment schemes (consisting of invokation of two ECom schemes) does satisfy g-open-decomposability (by simply setting PubOpen to the constant function outputting 1 and PrivOpen = Open itself). Then, we show that the non-malleability stengthening technique always outputs a scheme that is g-open-decomposable, and on input such a scheme, the log-n trick produces a scheme that is $h(n)$-open-decomposable for $h(n) = ng(n)$.

Modification to the strengthening technique described in Section 5.5.3. Let $\langle C, R\rangle$ be one-one non-malleable w.r.t. extraction and satisfy h-open-decomposable w.r.t. $(\mathsf{PubOpen}, \mathsf{PrivOpen}, \pi)$. We describe the changes (highlighted in red) to the non-malleability strengthening technique.

- Commit stage - First round: Same as before.

- Commit stage - Second round: Steps 1, 2 and 4 are same as before.

  3. Given a_{ZAP} and for $c^* = \pi(a_{\mathsf{NM}}, b_{\mathsf{NM}})$, $\widehat{C}$ computes the second message b_{ZAP} of ZAP to prove the following OR-statement:

     (a) *either* there exists a string $\bar{v}$ such that $c1$ is a commitment to $\bar{v}$ and $c3$ commits to a decommitment of $c1$.

     (b) *or* there exists a string $\bar{s} = (x_1, x_2)$, such that,

     - $h(x_1) = h(x_2)$,
     - $c2$ is a commitment to $\bar{s}$,
     - $c4$ commits to a decommitment of $c2$,
     - PrivOpen accepts $(c^*, d4, v4)$, and $(d4, v4)$ is a valid decommitment to $c4$.

$\widehat{C}$ proves the statement (a) by using a decommitment of $c3$ to $(v, d1)$ — decommitment of $c1$ to v — as the witness.

Denote by $(\hat{a}_{\mathsf{NM}}, \hat{b}_{\mathsf{NM}})$ the produced commitment.

- Reveal stage - Function $\widehat{\mathsf{Open}}((\hat{a}_{\mathsf{NM}}, \hat{b}_{\mathsf{NM}})), d1, v)$:

  Parse $(\hat{a}_{\mathsf{NM}}, \hat{b}_{\mathsf{NM}})$ and let $(a_{\mathsf{ZAP}}, b_{\mathsf{ZAP}})$, $(a_{\mathsf{NM}}, b_{\mathsf{NM}})$, and $c1$ be the ZAP proof, the commitment of $\langle C, R\rangle$, and the ECom_1 commitment contained in it. Accept if and only if the following functions both accept.

  - $\widehat{\mathsf{PubOpen}}(\hat{a}_{\mathsf{NM}}, \hat{b}_{\mathsf{NM}})$ accepts iff the ZAP proof $(a_{\mathsf{ZAP}}, b_{\mathsf{ZAP}})$ is accepting and $\mathsf{PubOpen}((a_{\mathsf{NM}}, b_{\mathsf{NM}})) = 1$.
  - $\widehat{\pi}(\hat{a}_{\mathsf{NM}}, \hat{b}_{\mathsf{NM}}) = c1$ and $\widehat{\mathsf{PrivOpen}}(c1, v, d1)$ accepts iff $\mathsf{EOpen}_1(c1, v, d1) = 1$.

The scheme $\langle \widehat{C}, \widehat{R}\rangle$ is open-decomposable w.r.t. $(\widehat{\mathsf{PubOpen}}, \widehat{\mathsf{PrivOpen}}, \widehat{\pi})$. Since $\widehat{\mathsf{PrivOpen}}$ only checks the decommitment of the ECom_1 commitment, its runtime is bounded by $g(n)$. Thus, $\langle \widehat{C}, \widehat{R}\rangle$ is $g(n)$-open-decomposable. On the other hand, since $\mathsf{PrivOpen}$ has complexity $h(n)$, the ZAP proof incurrs an additive $\mathsf{poly}(n, g(n), h(n))$ overhead. Then,

$$\widehat{cc}(n) = cc(n) + \mathsf{poly}(n, g(n), h(n)) \ ,$$

where $cc(n)$ and $\widehat{cc}(n)$ are the computational complexities of $\langle C, R\rangle$ and $\langle \widehat{C}, \widehat{R}\rangle$ respectively.

Modification to log-n trick described in Section 5.6. Let $\langle \widehat{C}, \widehat{R}\rangle$ be concurrent non-malleable (w.r.t. commitment and extraction) for $l(n)$-bit identities, and be $g(n)$-open-decomposable w.r.t. $(\widehat{\mathsf{PubOpen}}, \widehat{\mathsf{PrivOpen}}, \widehat{\pi})$. The log-n trick results in a commitment scheme $\langle \widetilde{C}, \widetilde{R}\rangle$ which is one-one non-malleable (w.r.t. commitment and extrac-

tion) for identities of length $l'(n) = 2^{l(n)-1} < n$. We show that $\langle \widetilde{C}, \widetilde{R} \rangle$ is $h(n)$-open-decomposable w.r.t. $(\widetilde{\mathsf{PubOpen}}, \widetilde{\mathsf{PrivOpen}}, \widetilde{\pi})$ described below.

- Commit stage: Same as before.

  Let $\widetilde{a}_{\mathsf{NM}}, \widetilde{b}_{\mathsf{NM}}$ be the produced commitment, which contains l' commitments of $\langle \widehat{C}, \widehat{R} \rangle$, denoted as $\left\{ \widehat{a}^i_{\mathsf{NM}}, \widehat{b}^i_{\mathsf{NM}} \right\}_{i \in [l']}$.

- Reveal stage - Function $\widetilde{\mathsf{Open}}((\tilde{a}_{\mathsf{NM}}, \tilde{b}_{\mathsf{NM}})), d, v)$: Accept if and only if the following functions both accept.

  - $\widetilde{\mathsf{PubOpen}}$ accepts iff for every i, $\widehat{\mathsf{PubOpen}}(\widehat{a}^i_{\mathsf{NM}}, \widehat{b}^i_{\mathsf{NM}})$ accepts.
  - $\widetilde{\pi}(\widetilde{a}_{\mathsf{NM}}, \widetilde{b}_{\mathsf{NM}}) = \left\{ c^*_i = \widehat{\pi}(\widehat{a}^i_{\mathsf{NM}}, \widehat{b}^i_{\mathsf{NM}}) \right\}_i$ and $\widetilde{\mathsf{PrivOpen}}$ accepts iff for every i, $\widehat{\mathsf{PrivOpen}}$ accepts c^*_i w.r.t. d, v.

Note that the running time of $\widetilde{\mathsf{PrivOpen}}$ is at most $l'(n) \cdot g(n) \leq h(n)$, and hence $\langle \widetilde{C}, \widetilde{R} \rangle$ is $h(n)$-open-decomposable. Furthermore, if the computational complexity of $\langle \widehat{C}, \widehat{R} \rangle$ is $\widehat{cc}(n)$, the computational complexity of $\langle \widetilde{C}, \widetilde{R} \rangle$ is bounded by $l'(n)\widehat{cc}(n)$.

Putting Pieces Together. Every iteration, say the j'th iteration, starts with a commitment scheme $\langle C^j, R^j \rangle$ supporting $\mathsf{id}^j(n)$ length identities, that is $h(n)$-open-decomposable (the inital schemes are g-open-decomposable). Applying the non-malleability strengthening technique produces a scheme $\langle \widehat{C}^j, \widehat{R}^j \rangle$ that is $g(n)$-open-decomposable. Following that, the log-n trick produces a scheme $\langle C^{j+1}, R^{j+1} \rangle$, supporting $\mathsf{id}^{j+1}(n) = 2^{\mathsf{id}^j(n)-1}$ length identities, that is $h(n)$-open-decomposable for $h(n) = ng(n)$. Furthermore, Let $cc(j)$ denote the computational complexity of the scheme $\langle C^j, R^j \rangle$.

Then we have:

$$\begin{aligned} cc(j+1) &= \mathsf{id}^{j+1}(n)\left(cc(j) + \mathsf{poly}(n, g(n), h(n))\right) \\ &= \mathsf{id}^{j+1}(n)\left(\mathsf{id}^{j}(n)(cc(j-1) + \mathsf{poly}(n)) + \mathsf{poly}(n)\right) \\ &\leq \mathsf{id}^{j+1}(n)\mathsf{id}^{j}(n)cc(j-1) + \mathsf{id}^{j+1}(n)\mathsf{id}^{j}(n)\mathsf{poly}(n) + \mathsf{id}^{j+1}(n)\mathsf{poly}(n) \end{aligned}$$

Then,

$$\begin{aligned} cc(j+1) &\leq \mathsf{id}^{j+1}(n)\mathsf{id}^{j}(n)cc(j-1) + 2\mathsf{id}^{j+1}(n)\mathsf{id}^{j}(n)\mathsf{poly}(n) \\ &\leq \prod_{1\leq k\leq j+1} \mathsf{id}^{k}(n)cc(0) + (j+1)\left(\prod_{1\leq k\leq j+1} \mathsf{id}^{k}(n)\right)\mathsf{poly}(n) \end{aligned}$$

Since the total number of iterations is $O(\log^* n)$ and the lengths of identities grow exponentially fast, we have that the running time of the final scheme $\langle C^*, R^*\rangle$ is upper-bounded by a polynomial.

5.7.4 Security of $\langle C^*, R^*\rangle$

Recall from Section 5.7.1 $\langle C^*, R^*\rangle$ is the commitment scheme obtained by applying the non-malleability strengthening step to the commitment scheme $\langle C^{r(n)}, R^{r(n)}\rangle$ which inturn was constructed by recursively applying, for $r(n)$ iterations, the non-malleability strengthening step followed by the log-n trick starting from the basic commitment scheme $\langle C^0, R^0\rangle$. Since, the number of iterations $r(n) = O(\log^* n)$ (i.e., $r(n) = w(1)$)[26], it is not apriori clear whether $\langle C^*, R^*\rangle$ is concurrent non-malleable for poly-size adversaries. Towards establishing the security of $\langle C^*, R^*\rangle$, we first focus on showing $\langle C^{r(n)}, R^{r(n)}\rangle$ is one-one non-malleable against poly-size adversaries. Then the security of $\langle C^*, R^*\rangle$ would follow from Theorem 21.

Recall the j-th step of the iteration: Starting from $\langle C^j, R^j\rangle$ commitment scheme on $\mathsf{id}^j(n)$-bit identities, first the non-malleability strengthening step is applied to $\langle C^j, R^j\rangle$

[26]Both non-malleability strengthening (Theorem 21) and log-n trick (Theorem 24) incur $O(m) = \mathsf{poly}(n)$ loss in the security where m is the number of concurrent interactions A participates in. Applying the transformation $r(n) = O(1)$ would have trivially implied security of $\langle C^*, R^*\rangle$.

resulting in a scheme $\langle \widehat{C}^{j+1}, \widehat{R}^{j+1} \rangle$ on $\mathsf{id}^j(n)$-bit identities. Then, the logn trick applied to $\langle \widehat{C}^{j+1}, \widehat{R}^{j+1} \rangle$ resulting in the commitment scheme $\langle C^{j+1}, R^{j+1} \rangle$ on $\mathsf{id}^{j+1}(n)$-bit identities. By Theorems 21, 24 we know that if $\langle C^j, R^j \rangle$ is one-one non-malleable then $\langle C^{j+1}, R^{j+1} \rangle$. First, let us establish some notation for the "advantage" of a certain adversary in breaking the non-malleability of the intermediate commitment schemes.

Notation For $j \in [r(n)]$ consider the commitment scheme $\langle C^j, R^j \rangle$. Consider some (A, D) where $A \in \mathcal{C}^{\wedge}_{S^j, S^j}$ and $D \in \mathcal{P}/\text{poly}$ let $\varepsilon^j_{A,D}(n)$ be a function $\mathbb{N} \to [0,1]$ such that for all $n \in \mathbb{N}$,

$$\varepsilon^j_{A,D}(n) = |\mathsf{Pr}[D_n(\mathsf{emim}^A_{\langle C^j, R^j \rangle}(1^n, 0))] - D_n(\mathsf{emim}^A_{\langle C^j, R^j \rangle}(1^n, 1))| \ .$$

Let $\varepsilon^j_A(n)$ be the maximum of $\varepsilon^j_{A,D}(n)$ over all $D \in \mathcal{P}/\text{poly}$. We refer to $\varepsilon^j_A(n)$ as A's advantage in breaking one-one non-malleability w.r.t. extraction of $\langle C^{j+1}, R^{j+1} \rangle$. Furthermore, let $\varepsilon^j(n)$ be the maximum of $\varepsilon^j_A(n)$ over all one-one adversary $A \in \mathcal{C}^{\wedge}_{S^j, S^j}$. Similarly, we define such an advantage function for the commitment scheme $\langle \widehat{C}^j, \widehat{R}^j \rangle$: For $A \in \mathcal{C}^{\wedge}_{S^j, S^j}$ participating in one left interaction with $\widehat{C}^j$ and $m^j(n) = \mathsf{id}^j(n)$ right interactions with $\widehat{R}^j$, let $\hat{\varepsilon}^j_A(n)$ be the advantage of A in breaking the one-many non-malleability of $\langle \widehat{C}^j, \widehat{R}^j \rangle$.

We are interested in showing that $\varepsilon^{r(n)}(n)$ is negligible. That is, the scheme $\langle C^{r(n)}, R^{r(n)} \rangle$ is $\mathcal{C}^{\wedge}_{S^{r(n)}, S^{r(n)}}$-one-one non-malleable commitment scheme on $\mathsf{id}^{r(n)}(n) \geq n$-bit identites. Since $\mathcal{P}/\text{poly} \subseteq \mathcal{C}^{\wedge}_{S^{r(n)}, S^{r(n)}}$, this also establishes the security of $\langle C^{r(n)}, R^{r(n)} \rangle$ against poly-size adversaries. Towards bounding $\varepsilon^r(n)$, we first bound $\varepsilon^{j+1}(n)$ as a function of $\varepsilon^j(n)$.

First, recall that by Theorem 24, for any $A \in \mathcal{C}^{\wedge}_{S^{j+1}, S^{j+1}}$ that breaks the one-one non-malleability of $\langle C^{j+1}, R^{j+1} \rangle$ with probability δ, there exists $A' \in \mathcal{C}^{\wedge}_{S^{j+1}, S^{j+1}}$ that participates in one left and $\mathsf{id}^{j+1}(n)$ right interactions and breaks the one-many non-

malleability of $\langle \widehat{C}^{j+1}, \widehat{R}^{j+1} \rangle$ with probability δ' such that

$$\delta \leq \mathsf{id}^{j+1}(n) \cdot \delta' \ . \tag{5.25}$$

Therefore, we can upperbound $\varepsilon^{j+1}(n)$ by,

$$\varepsilon^{j+1}(n) \leq \mathsf{id}^{j+1}(n) \cdot \hat{\varepsilon}^{j+1}(n) \ , \tag{5.26}$$

Next, recall that in the proof of Theorem 21, we reduce to the security of primitives inccuring a multiplicative loss in the advantage proportional to m – number of right interactions that the one-many adversary takes part in. While relating $\hat{\varepsilon}^{j+1}(n)$ with $\varepsilon^{j}(n)$ it suffices to restrict ourselves to adversaries that participate in one left and $m^{j+1}(n)$ right interactions (like the adversary A' above). Therefore,

$$\hat{\varepsilon}^{j+1}(n) \leq c \cdot m^{j+1}(n) \cdot \varepsilon^{j}(n) \ , \tag{5.27}$$

for some constant $c = O(1)$ dictated by proof of Theorem 21. More importantly, we note that $\hat{\varepsilon}^{j+1}(n)$ blows up by only a factor of $m^{j+1}(n) = \mathsf{id}^{j+1}(n)$ over $\varepsilon^{j}(n)$.

Combining Equations 5.27 and 5.26, we get

$$\varepsilon^{j+1}(n) \leq c \cdot \left(m^{j+1}(n)\right)^2 \cdot \varepsilon^{j}(n) \leq c \cdot \left(\mathsf{id}^{j+1}(n)\right)^2 \cdot \varepsilon^{j}(n) \ ,$$

Therefore,

$$\varepsilon^{j+1}(n) \leq c^{j} \cdot \prod_{0 \leq k \leq j+1} \left(\mathsf{id}^{k}(n)\right)^2 \cdot \varepsilon^{0}(n)$$

Plugging in $j + 1 = r(n)$, we get

$$\varepsilon^{r(n)}(n) \leq c^{r(n)} \cdot \prod_{0 \leq k \leq r(n)} \left(\mathsf{id}^{k}(n)\right)^2 \cdot \varepsilon^{0}(n)$$

Since, $r(n) = O(\log^* n)$, $c = O(1)$ and $\varepsilon^0(n)$ are negligible functions we conclude that $\varepsilon^{r(n)}(n)$ is negligible. This then establishes the security of $\langle \widehat{C}^*, \widehat{R}^* \rangle$. This now concludes the proof of Theorem 25.

5.8 Two-round Robust CCA-secure Commitment

In this section we consider a stronger notion of security for commitments – security against adaptive chosen commitment attacks (CCA-security). CCA-security for commitment schemes was defined in [82, 83] and is analogous to the extensively studied notion of security under chosen-ciphertext attacks for encryption schemes. Roughly speaking, a tag based commitment scheme is CCA-secure if the value committed using an tag id remains hidden even if the receiver has access to an oracle that "breaks" any commitment using any tag $\mathsf{id}' \neq \mathsf{id}$, and returns the (unique) *value committed* inside the commitment. We call such an oracle the *committed-value oracle*. CCA-security can be viewed as a natural strengthening of concurrent non-malleability – roughly speaking, a commitment scheme is concurrently non-malleable if it is CCA-secure with respect to restricted classes of adversaries that only make a single parallel (non-adaptive) query to the oracle after completing all interactions with the honest committer.

In this section, we show that the 2-round concurrent non-malleable commitment scheme described in Section 5.7 is in fact also CCA-secure. Recall that the 2-round scheme is constructed by iteratively applying the amplification transformation in Section 5.5 to the basic schemes for short identities in Section 5.4. The basic schemes for short identities are only one-one non-malleable which is amplified to concurrent non-malleability for n-bit identities by a two-step amplification procedure: first by applying the 2-round strengthening technique in Section 5.5.3 which strengthens the one-one non-malleability to concurrent non-malleability while preserving the length of identities; then applying the DDN *log n* trick (Section 5.6) to increase the length of identities while loosing concurrent non-malleability. The above two-step amplification step is iteratively applied for $O(\log^* n)$ times resulting in a scheme for n-bit identities but is only one-one non-malleable. To restore concurrent non-malleability the 2-round strengthening tech-

nique is applied once more. Since the strengthening technique is the final step in the construction, to show that the resulting scheme $\langle C^*, R^* \rangle$ is also CCA-secure, it is sufficient to show that the strengthening technique described in Section 5.5.3 produces a CCA-secure commitment scheme.

Below we first formally define the notion of CCA-secure commitments and then prove that the strengthening technique of Section 5.5.3 produces a CCA-secure scheme.

5.8.1 CCA-secure Commitment w.r.t. Committed-Value Oracle

Committed-value Oracle. Let $\langle C, R \rangle$ be a tag-based perfectly binding commitment scheme with $t(n)$-bit identities. Consider a non-uniform circuit family $A = \{A_n\}_{n \in \mathbb{N}}$. A committed-value oracle $\mathcal{O}$ of $\langle C, R \rangle$ acts as follows in interaction with A: For security parameter n, it participates with A in m-interactions acting as a honest receiver, using identities of length $t(n)$ which are adaptively chosen by A. At the end of each interaction, $\mathcal{O}$ returns the unique value committed in the interaction if it exists, otherwise it returns $\perp$. More precisely, $\mathcal{O}$ at the end of an interaction say with transcript c, computes the function val on c and returns $\mathsf{val}(c)$ to A. Recall that $\mathsf{val}(c)$ equals the (unique) value committed in c when c is a valid commitment, else $\mathsf{val}(c)$ is $\perp$.

A tag-based commitment scheme $\langle C, R \rangle$ is CCA-secure w.r.t. committed-value oracle, if the hiding property of the commitment scheme holds even with respect to adversaries that have access to the committed-value oracle $\mathcal{O}$. More precisely, let $A^{\mathcal{O}}$ denote the adversary A having access to the committed-value oracle $\mathcal{O}$. Consider the following probabilistic experiment $\mathsf{IND}(1^n, b)$, where $b \in \{0,1\}$: For security parameter n, $A^{\mathcal{O}}$ adaptively[27] chooses a pair of challenge values $(v_0, v_1) \in \{0,1\}^\alpha$ – the values to be committed to – and an identity id of length $t(n)$, and interacts with the honest committer

[27] the choice of values v_0, v_1 and the identity id can depend on the right interactions of A with the committed value oracle.

C to receive a commitment to v_b using identity id. Finally, the experiment outputs the output y of $A^{\mathcal{O}}$ where y is replaced with $\perp$ if A queries the oracle $\mathcal{O}$ on a commitment using an identity which is same as the identity id of the commitment it receives. We will denote the output of the above experiment by $\mathsf{IND}^{A}_{\langle C,R\rangle}(1^n, b)$.

Definition 34 (CCA-secure Commitments [83]) *Let $\langle C, R\rangle$ be a tag-based commitment scheme for $t(n)$-bit identities, and $\mathcal{C}$ a class of circuits. We say that $\langle C, R\rangle$ is $\mathcal{C}$-CCA-secure w.r.t. the committed-value oracle, if for every circuit family $A = \{A_n\}_{n\in\mathbb{N}} \in \mathcal{C}$ participating in $m = \mathsf{poly}(n)$ interactions with the oracle while sending/receiving commitments to $\alpha = \mathsf{poly}(n)$-bit values, the following ensembles are computationally indistinguishable:*

$$\left\{\mathsf{IND}^{A}_{\langle C,R\rangle}(1^n, 0)\right\}_{n\in\mathbb{N}} \;;\; \left\{\mathsf{IND}^{A}_{\langle C,R\rangle}(1^n, 1)\right\}_{n\in\mathbb{N}} . \tag{5.28}$$

As stated before and observed in [82, 83], CCA-security can be viewed as a natural strengthening of concurrent non-malleability. The proof is standard and is omitted but for completeness we state the theorem below.

Theorem 26 *Let $\langle C, R\rangle$ be a commitment scheme and $\mathcal{C}$ be a class of circuits that is closed under composition with $\mathcal{P}$/poly. Then if $\langle C, R\rangle$ is $\mathcal{C}$-CCA-secure w.r.t. the committed-value oracle then it is $\mathcal{C}$-concurrent non-malleable w.r.t. commitment.*

5.8.2 k-Robustness w.r.t. Committed-value Oracle

In the literature, CCA-security is usually used together with another property – *robustness* which captures security against a man-in-the-middle adversary that participates in an *arbitrary* left interaction with a *limited number of rounds*, while having access to the committed-value oracle. Roughly speaking, $\langle C, R\rangle$ is k-robust if the joint outputs of

every k-round interaction, with an adversary having access to $\mathcal{O}$, can be simulated without the oracle. In other words, having access to the oracle does not help the adversary in participating in any k-round protocol much.

Definition 35 (Robustness) *Let $\langle C, R\rangle$ be a tag based commitment scheme with $t(n)$-bit identities, and $\mathcal{C}$ and $\mathcal{C}'$ two classes of circuits. We say that $\langle C, R\rangle$ is $(\mathcal{C}, \mathcal{C}', k)$-robust w.r.t. the committed-value oracle, if there exists a simulator $S \in \mathcal{C}'$, such that, for every adversary $A \in \mathcal{C}$ that participates with $\mathcal{O}$ in $m = \mathsf{poly}(n)$ interactions and for every $B \in \mathcal{C}$ that participates in a k-round interaction with A the following ensembles are computationally indistinguishable,*

$$\left\{\mathsf{output}_{B,A^{\mathcal{O}}}[B, A^{\mathcal{O}}(1^n)]\right\}_{n\in\mathbb{N}} \; ; \; \left\{\mathsf{output}_{B,S^{A}}[B, S^{A^{\mathcal{O}_S}}(1^n)]\right\}_{n\in\mathbb{N}} \; , \tag{5.29}$$

where $\mathsf{output}_{B,A^{\mathcal{O}}}[B, A^{\mathcal{O}}(1^n)]$ denote the joint outputs of A and B in an interaction between them with uniformly and independently chosen random inputs to each machine and $\mathcal{O}_S$ is the oracle simulated by S for A.

Remark 16 *In the standard definition of robustness [83], the probabilistic poly-time adversaries A and B are given auxiliary inputs – private inputs y and z respectively and common input x. Since, our adversaries our non-uniform we can assume that the values x, y, z are instead hardcoded in A and B.*

5.8.3 Proof of Robust CCA-security of $\langle \widehat{C}, \widehat{R}\rangle$

The commitment scheme $\langle \widehat{C}, \widehat{R}\rangle$ is a result of applying the strengthening technique described in Section 5.5.3 to a 2-round over-extractable $\mathcal{C}^{\wedge}_{S_{\mathsf{NM}},S_{\mathsf{NM}}}$ one-one non-malleable (w.r.t. extraction) commitment scheme $\langle C, R\rangle$. The strengthening technique additionally relies on other basic building blocks described in Section 5.5.2. It was shown in

Theorems 20,21,22 that $\langle \widehat{C}, \widehat{R} \rangle$ is over-extractable w.r.t. extractor $\widehat{o\mathcal{E}}_{\mathsf{NM}}$ and is $\mathcal{C}^{\wedge}_{d_4,d_4}$ concurrent non-malleable w.r.t. extraction and commitment. Next, we will show that $\langle \widehat{C}, \widehat{R} \rangle$ is also $\mathcal{C}^{\wedge}_{d_4,d_4}$-CCA-secure and $(\mathcal{C}^{\wedge}_{d_4,d_4}, \mathcal{C}^{\wedge}_{d_2,S_{\mathsf{CRH}}}, \kappa)$-robust for any polynomial κ.

Theorem 27 *$\langle \widehat{C}, \widehat{R} \rangle$ is $\mathcal{C}^{\wedge}_{d_4,d_4}$-CCA-secure and is $(\mathcal{C}^{\wedge}_{d_4,d_4}, \mathcal{C}^{\wedge}_{d_2,S_{\mathsf{CRH}}}, \kappa(n))$-robust w.r.t. committed-value oracle for any polynomial κ.*

The proof of Theorem 27 consists of two parts: in Section 5.8.3 we first show that $\langle \widehat{C}, \widehat{R} \rangle$ is CCA-secure and in Section 5.8.3 we show that it is also robust.

Proof of CCA-security

Let us consider a fixed family of circuits $A = \{A_n\}_{n\in\mathbb{N}}$ belonging to the circuit class $\mathcal{C}^{\wedge}_{d_4,d_4}$ that in the CCA-experiment $\mathsf{IND}(1^n, b)$ interacts with a honest receiver C and has oracle access to the committed-value oracle to which it makes $m = \mathsf{poly}(n)$ number of queries. For convenience, we will refer to A's interaction with C as the *left* interaction and its interactions with $\mathcal{O}$ as *right* interactions. Then, to prove CCA-security, we need to show that

$$\{\mathsf{IND}^{A}_{\langle \widehat{C},\widehat{R}\rangle}(1^n, 0)\}_{n\in\mathbb{N}} \approx_c \{\mathsf{IND}^{A}_{\langle \widehat{C},\widehat{R}\rangle}(1^n, 1)\}_{n\in\mathbb{N}} \ . \tag{5.30}$$

Proof Overview. At a very high level: the above indistinguishability follows from similar proof as that of one-many non-malleability in Section 5.5.3. The proof goes through similar hybrids $\{H_j\}_j$ as that for proving non-malleabilty in the proof of Theorems 21 and 22, with the following slight modification. In the definition of non-malleability, the man-in-the-middle A interacts with the honest receivers on the right, whereas in that for CCA security, A interacts with the committed-value oracle $\mathcal{O}$ on the right, who additionally returns the value val committed in every right interaction as soon as it ends. Therefore, in the hybrids for proving CCA-security, we need to simulate $\mathcal{O}$ for A. To

do so, we rely on the over-extractability of $\langle \widehat{C}, \widehat{R} \rangle$ by an extractor $\widehat{o\mathcal{E}}_{\mathsf{NM}}$, and simulate the committed-value oracle for A using the following *extracted-value oracle* — $\mathcal{O}_E$ works identically to the committed-value oracle except that at the end of an interaction, it runs $\widehat{o\mathcal{E}}_{\mathsf{NM}}$ to extract a value from the commitment and returns it to A.

With the modified hybrids, to show CCA-security, we need to establish that i) $\mathcal{O}_E$ indeed simulates the committed-value oracle correctly, and ii) the indistinguishability of the hybrids remains. For i), recall that the over-extractability of $\langle \widehat{C}, \widehat{R} \rangle$ only guarantees that the value $\mathcal{O}_E$ extracts is the correct committed value when a commitment is valid, otherwise, $\widehat{o\mathcal{E}}_{\mathsf{NM}}$ might return an arbitrary value, instead of $\perp$. To show that the latter does not happen, (similar to the proof of non-malleability in Theorem 21 and 22,) we maintain throughout all hybrids a "no-fake-witness" invariant, which would guarantee that $\widehat{o\mathcal{E}}_{\mathsf{NM}}$ indeed returns $\perp$ when a right commitment is invalid, except with negligible probability. Hence, $\mathcal{O}_E$ perfectly simulates the committed-value oracle with overwhelming probability.

Next, to show ii) the indistinguishability of the hybrids, recall that the extractor $\widehat{o\mathcal{E}}_{\mathsf{NM}}$ on a commitment c works as follows: It returns $\perp$ if the ZAP proof (in c) is not accepting, and otherwise, it return the value v' extracted from $c1$ using the extractor $o\mathcal{E}_1$ of ECom_1 — the complexity of $\widehat{o\mathcal{E}}_{\mathsf{NM}}$ is roughly the same as that of $o\mathcal{E}_1$. Observe that running the extractor $o\mathcal{E}_1$ of ECom_1, and hence $\widehat{o\mathcal{E}}_{\mathsf{NM}}$, in the hybrids does not hurt the security of any other components, namely, $\mathsf{CRH}, \langle C, R \rangle$ and ECom_i's for $i > 1$, since $\mathsf{ECom}_1 \prec \mathsf{CRH}, \langle C, R \rangle, \mathsf{ECom}_2, \mathsf{ECom}_4, \mathsf{ECom}_3$, as depicted in Figure 5.2 (iii). Therefore, if the indistinguishability of a pair of neighboring hybrids reduces to the security of components other than ECom_1, this indistinguishability remains intact even when running $\widehat{o\mathcal{E}}_{\mathsf{NM}}$ inside. This is the case for all but the last two hybrids, whose indistinguishability reduces to the hiding of ECom_1 itself. To show their indistinguishability, (again similar to the proof of non-malleability,) we simulate $\widehat{o\mathcal{E}}_{\mathsf{NM}}$ by extracting from the commitment

$c3$ using the extractor $o\mathcal{E}_3$ for ECom_3, and rely on the hiding of ECom_1 against $o\mathcal{E}_3$. This concludes the overview of the proof of CCA-security. Next, we provide a more formal analysis.

Proof Sketch We consider a sequence of hybrids $\{G_j(b)\}_{0\leq j\leq 6}$ for $b \in \{0,1\}$ where for every $0 \leq j \leq 6$ and $b \in \{0,1\}$ the hybrid $G_j(b)$ is identical to the hybrid $H_j(b)$ described in the Proof of Theorem 21 in Section 5.5.3 except one slight difference. For its right interactions A in $G_j(b)$ interacts with the extracted-value oracle $\mathcal{O}_E$ instead of the honest receiver as in $H_j(b)$. Note that the hybrid $G_0(b)$ as described above emulates an execution which is identical to the CCA-experiment $\mathsf{IND}(b)$[28] with A except A is given access to the extracted-value oracle $\mathcal{O}_E$ instead of the committed-value oracle. As before, for notational convenience, we use font style x to denote a random variable in the left interaction, and font style $\tilde{x}_i$ the corresponding random variable in the i'th right interaction. Moreover, by $\mathsf{IND}^A_{G_j}(1^n, b)$ we will denote the output of the hybrid $G_j(b)$. Then to show indistinguishability as described in Equation (5.30), we prove in Lemma 36 that the output of the neighbouring hybrids $G_j(b)$ and $G_{j+1}(b)$ are indistinguishable for the same b. Furthermore, we show the output is statistically close in $G_6(1)$ and $G_5(0)$ and the output of $G_0(b)$ is also statistically close to $\mathsf{IND}^A_{\langle\widehat{C},\widehat{R}\rangle}(b)$, this then establishes Equation (5.30).

Lemma 36 *For $b \in \{0,1\}$ and $0 \leq j \leq 5$, the following are computationally indistinguishable,*

$$\mathsf{IND}^A_{G_j}(b) \; ; \; \mathsf{IND}^A_{G_{j+1}}(b) \; ,$$

and $\mathsf{IND}^A_{G_0}(b) \approx_s \mathsf{IND}^A_{\langle\widehat{C},\widehat{R}\rangle}(b)$ and $\mathsf{IND}^A_{G_6}(b) \approx_s \mathsf{IND}^A_{G_5}(0)$.

Towards proving the above lemma, we will also maintain the following "no-fake-witness" invariant (similar to Invariant 1 in Section 5.5.3).

[28] We ignore the security parameter for notational convenience

Invariant 3 (No-fake-witness invariant) *In $G_j(b)$, the probability that there exists a right interaction i that is accepting and A commits to a fake witness in it is negligible.*

Showing the no-fake-witness invariant in every hybrid enforces A to prove the honest statement in every accepting right interaction k. That is, for every accepting right interaction k, A proves that the underlying commitment $\widetilde{c1}_k$ is valid. Then, due to the over-extraction property of the extractor $o\mathcal{E}_1$, it follows that A in its interaction with the extracted-value oracle in fact receives the value actually committed inside the right commitment $\tilde{c}_k$. Therefore A's interaction with the extracted-value oracle is identical to its interaction with the committed-value oracle, except with negligible probability. This fact will come in handy to show Lemma 36.

In fact, as in the proof of Theorems 21 and 22, we maintain the following, easier to prove, invariant which from an argument similar to the one made in the proof of Claim 10, implies Invariant 3.

Invariant 4 *In $G_j(b)$, the probability that there exists a right interaction i that is accepting and the value* extracted *from the non-malleable commitment $(\widetilde{a}_{\mathsf{NM}i}, \widetilde{b}_{\mathsf{NM}i})$ in this interaction is a fake witness is negligible.*

Therefore to establish the proof of CCA-security, we will prove Lemma 36 and show that Invariant 4 holds in all hybrids.

First, we show that Invariant 4 holds in $G_0(b)$. In fact, as in Claim 11, we show that the value extracted from the ECom_2 commitment $\widetilde{c2}_k$ in any right interaction k is not a collision of the hash function $\widetilde{h}_k$ where A interacts with $\mathcal{O}_E$ for its right interactions. This then implies Invariant 4 holds. At a high level this readily follows from the fact that the collision-resistance of the hash function is more secure than both ECom_2 and ECom_1, $h \succ \mathsf{ECom}_2, \mathsf{ECom}_1$ (see Figure 5.2 (iii)). This is because if in some right interaction k, A commits to a collision of $\widetilde{h}_k$ using ECom_2, then we can construct a non-uniform circuit

B' that violates the collision-resistance of $\widetilde{h}_k$ by extracting from $\widetilde{c2}_k$. More precisely, B' behaves identically to the adversary B in the proof of Claim 11 except that for all its $m = \mathsf{poly}(n)$ right interactions with A, B' internally simulates the oracle $\mathcal{O}_E$ by running the extractor $o\mathcal{E}_1$ whereas B just acts as a honest receiver. Therefore, the size of B' blows up by an additive factor of $m \cdot \mathsf{size}(o\mathcal{E}_1)$ over the size of B. Since size of B is at most $\mathsf{poly}(S_{\mathsf{CRH}})$ and the size of $o\mathcal{E}_1$ is also at most $\mathsf{poly}(S_{\mathsf{CRH}})$, we have that B' also has size $\mathsf{poly}(S_{\mathsf{CRH}})$, that is,

$$\begin{aligned}
\mathsf{size}(B') &\leq \mathsf{size}(B) + m \cdot \mathsf{size}(o\mathcal{E}_1) \\
&\leq \mathsf{size}(A) + \mathsf{size}(o\mathcal{E}_2) + \mathsf{poly}(n) + m \cdot \mathsf{size}(o\mathcal{E}_1) \\
&\leq \mathsf{poly}(d_4) + \mathsf{poly}(S_1) + \mathsf{poly}(S_{\mathsf{CRH}}) \\
&< \mathsf{poly}(S_{\mathsf{CRH}}) \qquad (\text{since, } S_{\mathsf{CRH}} >> S_1, d_4 \text{ from Equation (5.9)}).
\end{aligned}$$

Therefore, $B \in \mathcal{C}^{\wedge}_{S_{\mathsf{CRH}}, S_{\mathsf{CRH}}}$ and then due to the $\mathcal{C}_{S_{\mathsf{CRH}}}$-collision-resistance of $\mathcal{H}$ we have that Invariant 4 holds in $G_0(b)$ as formalized in the following claim,

Claim 25 *For $b \in \{0,1\}$ and for every right interaction i in $G_0(b)$, the probability that i is accepting and the value extracted from $(\widetilde{a}_{\mathsf{NM}_i}, \widetilde{b}_{\mathsf{NM}_i})$ is a fake witness, is negligible.*

We recall that the only difference between showing Invariant 4 holds in $G_0(b)$ from the proof of Claim 11 was that the adversary B' (defined similarly to adversary B from Claim 11) additionally runs the extractor $o\mathcal{E}_1$ to simulate the extracted value oracle for A for its right interactions. In fact one can show Invariant 4 holds in hybrids $G_1(b)$ through $G_5(b)$ via the same modification to adversary constructed in the proof of Claims 12, 14, 16, 18 and 20 respectively. Since, $\mathsf{ECom}_1 \prec \langle C, R\rangle, \mathsf{ZAP}, \mathsf{ECom}_i$ $(i > 1)$, it can be observed that running the extractor $o\mathcal{E}_1$ of $(\mathsf{ECom}_1, \mathsf{EOpen}_1)$ does not blow up the size or depth of the modified adversary much still allowing us to reach a contradiction as in the above Claims from Section 5.5.3. Furthermore, $G_6(b)$ is in fact identical to

$G_5(0)$ (as $H_5(0)$ identical to $H_6(b)$), therefore Invariant 4 also holds in $G_6(b)$. Therefore essentially by the same proofs in Section 5.5.3, we conclude that Invariant 4 does hold in all hybrids $G_j(b)$. This is captured in the following Claim and we skip a formal proof.

Claim 26 *For $b \in \{0,1\}, 0 \leq j \leq 6$ and for every right interaction i in $G_j(b)$, the probability that i is accepting and the value extracted from $(\widetilde{a}_{\mathsf{NM}i}, \widetilde{b}_{\mathsf{NM}i})$ is a fake witness, is negligible.*

Next we move onto showing Lemma 36. First, given Claim 25 holds, we show that the output of hybrid $G_0(b)$ is statistically close to the CCA-experiment $\mathsf{IND}(b)$ for any $b \in \{0,1\}$.

Claim 27 *For $b \in \{0,1\}$, the following holds*

$$\mathsf{IND}^{A}_{G_0}(b) \approx_s \mathsf{IND}^{A}_{\langle \widehat{C}, \widehat{R} \rangle}(b) \ .$$

Note that due to Claim 25 and also because Invariant 4 implies Invariant 3, we know that A in each of its (accepting) right interactions, with the oracle $\mathcal{O}_E$, does not commit to a fake witness, except with negligible probability. Then by the soundness of ZAP, in every accepting right interaction k, A proves with overwhelming probability that the underlying commitment $\widetilde{c1}_k$ is well-formed . Therefore by the over-extractability of ECom_1 w.r.t. $o\mathcal{E}_1$, we know that the extracted value oracle $\mathcal{O}_E$ (implemented using $o\mathcal{E}_1$) in fact returns $\mathsf{val}(\widetilde{c1}_k) = \mathsf{val}(\widetilde{c}_k)$. For interactions k that are not accepting, both val and $\mathcal{O}_E$ return $\perp$. Therefore for every right interaction k, the values returned by the extracted-value oracle $\mathcal{O}_E$ agree with $\mathsf{val}(c_k)$, the values returned by the committed-value oracle $\mathcal{O}$ except with negligible probability. Therefore, interaction of A with the extracted-value oracle $\mathcal{O}_E$ is statistically close to its interaction with the committed-value oracle $\mathcal{O}$ implying Claim 27.

Next we show that the output of $G_0(b)$ is indistinguishable from the output of $G_1(b)$, that is, $\mathsf{IND}^A_{G_0}(b)$ and $\mathsf{IND}^A_{G_1}(b)$ are indistinguishable. As in Claim 13, we construct an adversary B' that violates the hiding of ECom_2, that is, B' works identically to the adversary B in the proof of Claim 13 except Step 4 and 5. The adversary B (in Claim 13) waits for A to terminate and then in its Step 4 runs $o\mathcal{E}_1$ to obtain $\tilde{v_i}'$ for every successful right interaction i and sets $\tilde{v_i}' = \perp$ for every unsuccessful right interaction i. Then in Step 5, B runs the distinguisher D on the view of A and right extracted values $\tilde{v_i}'$ and returns the output of D. However, here in the CCA case, A expects to receive the extracted values and that too as soon as a right interaction ends. Therefore, B' runs the extractor $o\mathcal{E}_1$ to obtain $\tilde{v_i}'$ as soon as the i-th interaction ends and returns $\tilde{v_i}'$ to A if i is an accepting right interaction. Otherwise, it returns $\perp$. Then it runs the distinguisher D on the output y of A which is carefully replaced with $\perp$ if any of A's right interactions uses the same identity as its left interaction. The former change of running the extractor $o\mathcal{E}_1$ to obtain $\tilde{v_i}'$ is similar to the modification made while proving that Invariant 4 holds in $G_j(b)$ (Claim 26) and the later change is merely a syntactic change required to be consistent with the IND experiment. Note that this change in the code of B' does not blow up its depth significantly (over B). Since depth of B is at most $\mathsf{poly}(d_2)$ and $o\mathcal{E}_1 \in \mathcal{C}^{\wedge}_{d_2,S_{\mathsf{CRH}}}$, we have that $\mathsf{dep}(B')$ is at most $\mathsf{poly}(d_2)$. Then the $\mathcal{C}^{\vee}_{d_2,S_2}$-hiding of ECom_2 implies that the output of $G_0(b)$ and $G_1(b)$ are indistinguishable as formalized in the following claim,

Claim 28 *For $b \in \{0,1\}$, the following are indistinguishable,*

$$\mathsf{IND}^A_{G_0}(b); \mathsf{IND}^A_{G_1}(b) \ .$$

We recall that the only difference in showing indistinguishability of $\mathsf{IND}_{G_0}(b)$ and $\mathsf{IND}_{G_1}(b)$ from the proof of Claim 13 was that the adversary B' (defined similarly to adversary B from Claim 13) additionally runs the extractor $o\mathcal{E}_1$ to simulate the extracted

value oracle for A for its right interactions and runs the distinguisher D on the output of A instead of running D on the view of A and the values extracted from its right interactions. In fact one can show that $\mathsf{IND}_{G_1}(b)$ through $\mathsf{IND}_{G_5}(b)$ are all indistinguishable via the same modification to adversary constructed in the proof of Claims 15, 17, 19, 21 respectively. Since, $\mathsf{ECom}_1 \prec \langle C, R\rangle, \mathsf{ZAP}, \mathsf{ECom}_i$ $(i > 1)$, it can be observed that running the extractor $o\mathcal{E}_1$ of $(\mathsf{ECom}_1, \mathsf{EOpen}_1)$ does not blow up the size or depth of the modified adversary much still allowing us to reach a contradiction as in the above Claims from Section 5.5.3.

Claim 29 *For $j \in [4], b \in \{0,1\}$ the following holds*

$$\mathsf{IND}^A_{G_i}(b) \approx_c \mathsf{IND}^A_{G_{i+1}}(b) \ .$$

One would like to extend the above arugment to even argue the indistinguishability of $\mathsf{IND}_{G_5}(b)$ and $\mathsf{IND}_{G_6}(b)$ based on the proof of Claim 23 which reduces to the hiding of ECom_1. However, running $o\mathcal{E}_1 \in \mathcal{C}^{\wedge}_{d_2, S_{\mathsf{CRH}}}$ on the right blow up the size and depth of B' significantly, that is, $\mathsf{size}(B') \geq \mathsf{size}(o\mathcal{E}_1) = \mathsf{poly}(S_{\mathsf{CRH}}) >> S_1$ and similarly $\mathsf{dep}(B') >> d_1$. Since $B' \notin \mathcal{C}^{\vee}_{d_1, S_1}$, it does not violate the $\mathcal{C}^{\vee}_{d_1, S_1}$-hiding of ECom_1. To fix this issue we consider two intermediate hybrids $G'_5(b)$ and $G'_6(b)$ which are statistically close to $G_5(b)$ and $G_6(b)$ respectively and then argue how proof of Claim 23 can be extended to argue the indistinguishability of $\mathsf{IND}_{G_5}(b)$ and $\mathsf{IND}_{G_6}(b)$.

Hybrid $G'_j(b)$ for $5 \leq j \leq 6$: The hybrids $G'_j(b)$ are identical to $G_j(b)$ for $5 \leq j \leq 6$ except the implementation of the extracted-value oracle. Here, the extracted-value oracle behaves as before except that for an accepting right interaction i, the extractor $o\mathcal{E}_3$ is run on the underlying $\widetilde{c3}_i$ commitment to extract the value $(\tilde{v_i}', \widetilde{d1_i}')$ and the value $\tilde{v_i}'$ is returned.

Given that Invariant 4 holds in $G_j(b)$ (Claim 26), A in each of its (accepting) interaction with the oracle $\mathcal{O}_E$ does not commit to the fake witness, except with negligible

probability. Therefore, by the soundness of ZAP, in every accepting right interaction k, A proves that the underlying commitments $\widetilde{c1}_k$ and $\widetilde{c3}_k$ are well-formed and $\widetilde{c3}_k$ commits to a decommitment of $\widetilde{c1}_k$. Then due to the over-extractability of ECom_3 w.r.t. $o\mathcal{E}_3$ we know that the value $(\tilde{v_k}', \widetilde{d1}_k{}')$ extracted by $o\mathcal{E}_3$ is in fact the decommitment of $\widetilde{c1}_k$ which implies that $\tilde{v_k}'$ is in fact the value committed inside $\tilde{c1}_k$. Since the commitment $\tilde{c1}_k$ is also well-formed (as described above), the over-extractability of ECom_1 w.r.t. $o\mathcal{E}_1$ implies that the value extracted from $\tilde{c1}_k$ is also equal to $\mathsf{val}\tilde{c1}_k$ except with negligible probabilty. Therefore, the value $\tilde{v_k}'$ (extracted by $o\mathcal{E}_3$) is equal to the value extracted by $o\mathcal{E}_1$ in every right interaction which implies that the view of A in hybrids $G_j(b)$ and $G'_j(b)$ remains identical except with negligible probability. Therefore the following follow,

Claim 30 *For $b \in \{0,1\}$ and $5 \leq j \leq 6$, the following holds,*

$$\mathsf{IND}^A_{G_j}(b) \approx_s \mathsf{IND}^A_{G'_j}(b) \ .$$

Next we show that $\mathsf{IND}^A_{G'_5}(b)$ and $\mathsf{IND}^A_{G'_6}(b)$ are indistinguishable. This follows from the fact that ECom_1 is more secure than ECom_3, $\mathsf{ECom}_1 \succ \mathsf{ECom}_3$ (see Figure 5.2 (iii)). More precisely we construct an adversary B' that violates the hiding of ECom_1 where B' works identically to the adversary B in the proof of Claim 23 except a slight difference. Here, B' to simulate the extracted-value oracle runs the extractor $o\mathcal{E}_3$ to obtain $\tilde{v_i}'$ as soon as the i-th interaction ends unlike B which runs the extractor $o\mathcal{E}_3$ after all the right interactions end. This change in the code of B' does not blow up its size and depth significantly (over B) and therefore B' (like B) falls in the class $\mathcal{C}^{\vee}_{d_1,S_1}$. Then the $\mathcal{C}^{\vee}_{d_1,S_1}$-hiding of ECom_1 implies that the output of $G'_5(b)$ and $G'_6(b)$ are indistinguishable.

Claim 31 *For $b \in \{0,1\}$, the following are indistinguishable,*

$$\mathsf{IND}^A_{G'_5}(b); \mathsf{IND}^A_{G'_6}(b) \ .$$

Then combining Claims 27, 28,29,30 and 31 and observing that $\mathsf{IND}_{G_5}(0)$ is identical to $\mathsf{IND}_{G_6}(b)$ (as $G_5(0)$ is identical to $G_6(b)$ concludes the proof of Lemma 36 and hence the proof of CCA-security of $\langle \widehat{C}, \widehat{R} \rangle$.

Proof of Robustness

To show that $\langle \widehat{C}, \widehat{R} \rangle$ is $(\mathcal{C}^{\wedge}_{d_4,d_4}, \mathcal{C}^{\wedge}_{d_2,S_{\mathsf{CRH}}}, \kappa(n))$-robust, we need to show that for every $k \leq \kappa(n)$ there exists a simulator $S \in \mathcal{C}^{\wedge}_{d_2,S_{\mathsf{CRH}}}$ such that for any $A \in \mathcal{C}^{\wedge}_{d_4,d_4}$ and for any $B \in \mathcal{C}^{\wedge}_{d_4,d_4}$ that participates in a k-round interaction, interaction between B and A where A has access to the committed value oracle $\mathcal{O}$ is indistinguishable from that between B and S. In other words, S is able to simulate the committed value oracle $\mathcal{O}$ for A when its interacting with an arbitrary B. The construction of the simulator S is very similar to the hybrid $G_0(b)$ as described in the proof of CCA-security in Section 5.8.3. More precisely, given k and a circuit $A \in \mathcal{C}^{\wedge}_{d_4,d_4}$, S externally interacts with an arbitrary k-round circuit B and internally simulates an execution between B and A by forwarding messages from B to A. For the right interactions, S internally simulates the extracted value oracle $\mathcal{O}_E$ for A as described in Section 5.8.3.

To conclude the proof of robustness we need to show two things: (1) $S \in \mathcal{C}^{\wedge}_{d_2,S_{\mathsf{CRH}}}$ and (2) S indeed is able to simulate the committed-value oracle $\mathcal{O}$ for A. First, it is easy to see that S runs $A \in \mathcal{C}^{\wedge}_{d_4,d_4}$ and simulates the extracted-value oracle $\mathcal{O}_E$ for $\mathsf{poly}(m)$ right interactions. As $\mathcal{O}_E$ can be simulated by a circuit in $\mathcal{C}^{\wedge}_{d_2,S_{\mathsf{CRH}}}$ we have that $S \in \mathcal{C}^{\wedge}_{d_2,S_{\mathsf{CRH}}}$. Second, by an argument simular to the one made in the Proof of Claim 25 one can show that due to collision resistance of $\mathcal{H}$ A does not commit to a fake witness in any of its right interactions. Then, as in Claim 27 we can conclude that the view of A with the committed-value oracle is statistically close to the view of A with the extracted-value

oracle (as simulated by S). This then concludes the proof of robustness and the proof of Theorem 27.

On the robustness of the scheme $\langle C^*, R^* \rangle$. We claim that the final commitment scheme $\langle C^*, R^* \rangle$ is $(\mathcal{P}/\text{poly}, \mathcal{C}', \kappa(n))$-robust w.r.t the committed-value oracle where $\mathcal{C}'$ is the set of all non-uniform circuits whose size is upperbounded by $\mathsf{poly}(2^{(\log n)^c})$ for a sufficiently large constant c. In other words, $\langle C^*, R^* \rangle$ is robust w.r.t. quasi-polynomial time simulation. Recall that the commitment $\langle C^*, R^* \rangle$ is constructed by repeatedly applying the transformations presented in Sections 5.5.3 and 5.6 relying on a $L = O(\log^* n)$ hierarchy $p_1 << \ldots << p_L$ of non-decreasing functions as discussed in Section 5.7.2 where each level $p_i = 2^{n_i^\varepsilon}$ for an appropriate security parameter n_i. Furthermore recall that the final step in the construction of $\langle C^*, R^* \rangle$ is applying the strengthening technique which relies on a hierarchy of functions as described in Equation (5.9). For this last step the corresponding functions $d_4, \ldots, d_2, \ldots, S_{\mathsf{CRH}}$ are instantiated from the first few functions in the hierarchy namely $p_1, \ldots, p_4, \ldots, p_7$. Setting n_is as discussed in Section 5.7.2 will ensure that d_2 and S_{CRH} are both than $\mathsf{poly}(2^{(\log n)^c})$[29] for some sufficiently large constant c. Hence, the simulator for $\langle C^*, R^* \rangle$ belongs to the class $\mathcal{C}'$ as described above.

5.9 Non-interactive Concurrent Non-Malleable and CCA-secure Commitment against Uniform Adversaries

In this section, we show that when restricting attention to uniform attackers, the first message in our 2-round concurrent non-malleable commitment scheme constructed

[29] Setting n_0 to, say, $(\log n)^2$ in Section 5.7.2.

in Section 5.7 can be removed (Theorem **??**). Recall that these 2-round protocols are obtained by iteratively applying the amplification transformation in Section 5.5 to the basic schemes for short identities in Section 5.4. While the basic schemes are in fact non-interactive, the amplification technique, however, produces schemes with 2 rounds. Our amplification technique involves two steps: Applying the DDN $\log n$ trick, which is actually round preserving, and the security strengthening step that lifts one-one non-malleability w.r.t. extraction to concurrent non-malleability w.r.t. extraction and commitment, while preserving the length of identities. In the security strengthening step, the output scheme has two rounds, where the first message is sent by the receiver and contains the index of a randomly sampled function h from a family of non-uniform CRHFs, the first message of a ZAP proof, and the first message of the input non-malleable commitment scheme (if there is any). Therefore, to remove the first message, our idea is to simply replace h for a fixed uniform CRHF, and replace ZAP with a NIWI, so that the transformation when applied to a non-interactive input commitment scheme, produces a non-interactive output scheme. The only drawback is that with the use of uniform CRHF, the output scheme is only secure against uniform adversaries. We also show that the output scheme of the modified strengthening technique also satisfies stronger notions of CCA-security and robustness when adversaries are restricted to be uniform Turning machines.

Below, we first adapt the notions of non-malleability w.r.t. extraction and commitment, and robust CCA-security to the setting of uniform attackers, and then describe the new amplification step.

5.9.1 Non-malleability against Uniform Adversaries

The notion of non-malleability w.r.t. commitment (or w.r.t. extraction) against *uniform* attackers is defined identically to that against non-uniform attackers as in Definition 29 (or Definition 30 resp.) in Section 5.2.5. To make the distinction explict between uniform and non-uniform we let $\mathsf{uMIM}^{A}_{\langle C,R\rangle}(1^n, b)$ represent the MIM experiment with a uniform adversary A (hence uMIM). We further denote by $\mathsf{umim}^{A}_{\langle C,R\rangle}(1^n, b)$ (or $\mathsf{uemim}^{A}_{\langle C,R\rangle}(1^n, b)$ resp.) the random variable describing the view of A together with the values committed in (or extracted from resp.) the right interactions.

Definition 36 (Non-malleability) *A tag-based commitment scheme $\langle C, R\rangle$ is said to be concurrent T-non-malleable against uniform attackers if for every $\mathsf{poly}(T)$-time uniform Turing machine A participating in $m = \mathsf{poly}(n)$ while sending/receiving commitments to $\alpha = \mathsf{poly}(n)$-bit values, concurrent interactions, the following ensembles are computationally indistinguishable:*

$$\left\{\mathsf{umim}^{A}_{\langle C,R\rangle}(1^n, 0)\right\}_{n\in\mathbb{N}} \;;\; \left\{\mathsf{umim}^{A}_{\langle C,R\rangle}(1^n, 1)\right\}_{n\in\mathbb{N}} .$$

Moreover, it is said to be concurrent T-non-malleable w.r.t. extraction against uniform attackers, if the above indistinguishability holds between $\mathsf{uemim}^{A}_{\langle C,R\rangle}(1^n, 0)$ and $\mathsf{uemim}^{A}_{\langle C,R\rangle}(1^n, 1)$.

5.9.2 Robust CCA-security against Uniform Adversaries

Next we define the notions of CCA-security and Robustness against uniform adversaries. The definition are identical to the non-uniform case as defined in Definitions 34 and 35 except that A in the CCA definition is a uniform Turing machine and all A, B and S in the robustness definition are uniform Turing machines. For completness we define them below.

Definition 37 *We say that $\langle C, R\rangle$ is T-CCA-secure w.r.t. the committed-value oracle against uniform attackers if Equation (5.28) holds for all $\mathsf{poly}(T)$-time uniform Turing machines A that participate in $m = \mathsf{poly}(n)$ queries to the oracle $\mathcal{O}$.*

Definition 38 *We say that $\langle C, R\rangle$ is (T, T', k)-robust w.r.t. the committed-value oracle against uniform attackers if there exists $\mathsf{poly}(T')$-time uniform Turing machine S such that Equation (5.29) holds for all $\mathsf{poly}(T)$-time uniform Turning machines A and B which participates in a k-round interaction.*

5.9.3 1-Message Security Strengthening Technique

We now present our one-message transformation for security strengthening. For some hierarchy of non-decreasing functions on $\mathbb{N}$ satisfying,

$$\begin{aligned} &n << d_4 << d_3 << d_1 << d_2 << S_2 << S_1 << S_{\mathsf{CRH}} << \\ &S'_{\mathsf{CRH}} << S_{\mathsf{NM}} << S'_{\mathsf{NM}} << S_3 << S_4 << S'_4 << S^* \ , \end{aligned} \tag{5.31}$$

the transformation relies on the following building blocks:

1. $(\mathsf{oNICom}, \mathsf{oNIOpen})$ is a non-interactive, tag-based commitment scheme for $t(n)$-bit identities that is S'_{NM}-over-extractable by extractor $o\mathcal{E}_{\mathsf{NI}}$. Furthermore, $\langle C, R\rangle$ is one-one S_{NM}-non-malleable w.r.t. extraction by $o\mathcal{E}_{\mathsf{NI}}$ against uniform adversaries.

2. $\{(\mathsf{ECom}_i, \mathsf{EOpen}_i)\}_{1\leq i\leq 4}$ are identical to that in Section 5.5.3.

3. NIWI is a non-interactive $\mathcal{C}_{S^*}$-witness-indistinguishable proof.

4. $\mathcal{H} = \{h_n\}_n$ is a S_{CRH}-uniform-collision resistant hash function such that there exists a $\mathsf{poly}(S'_{\mathsf{CRH}})$-time TM which finds collisions for $\mathcal{H}$ with probability 1.

Using the above mentioned building blocks, the transformation produces the scheme $(\mathsf{cNICom}, \mathsf{cNIOpen})$ which is non-interactive, tag-based commitment scheme for $t(n)$-bit

identities that is S_{CRH}-over-extractable w.r.t. an extractor $\widehat{o\mathcal{E}}_{\mathsf{NI}}$. Furthermore, $(\mathsf{cNICom}, \mathsf{cNIOpen})$ is concurrent d_4-non-malleable w.r.t. extraction by $\widehat{o\mathcal{E}}_{\mathsf{NI}}$ and concurrent d_4-non-malleable (w.r.t. commitment) against uniform attackers.

The committer $\widehat{C}$ and the receiver $\widehat{R}$ receive the security parameter 1^n and identity $\mathsf{id} \in \{0,1\}^{t(n)}$ as common input. Furthermore, $\widehat{C}$ gets a private input $v \in \{0,1\}^n$ which is the value to be committed.

- Commit stage:

1. $\widehat{C}$ computes a commitment $c1$ to the value v using ECom_1. Let $d1$ be the corresponding decommitment string.
2. $\widehat{C}$ computes a commitment $c3$ to the decommitment $(v, d1)$ of $c1$ using ECom_3.
3. $\widehat{C}$ computes a commitment $c2$ to a random string $r1$ using ECom_2.
4. $\widehat{C}$ computes a commitment $c\mathsf{NM}$ to a random string $r3$ using oNICom using identity id.
5. $\widehat{C}$ computes a commitment $c4$ to a random string $r3$ using ECom_4.
6. $\widehat{C}$ computes the NIWI proof π to prove the following OR-statement:
   (a) *either* there exists a string $\bar{v}$ such that $c1$ is a commitment to $\bar{v}$ and $c3$ commits to a decommitment of $c1$.
   (b) *or* there exists a string $\bar{s} = (x_1, x_2)$ such that $c2$ is a commitment to $\bar{s}$, $c4$ commits to a decommitment of $c2$, $c\mathsf{NM}$ commits to a decommitment of $c4$ and $H_n(x_1) = H_n(x_2)$.

   $\widehat{C}$ proves the statement (a) by using a decommitment of $c3$ to $(v, d1)$ — decommitment of $c1$ to v — as the witness.
7. $\widehat{C}$ sends $(c1, c2, c3, c4, c\mathsf{NM}, \pi)$ as commitment to $\widehat{R}$ and keeps the decommitment $(v, d1)$ private.

- Reveal stage:

  On receiving $(v, d1)$ from $\widehat{C}$, $\widehat{R}$ accepts the decommitment if the NIWI proof is accepting and if $\mathsf{EOpen}_1(c1, v, d1) = 1$. Otherwise, it rejects.

- Extraction - Extractor $\widehat{o\mathcal{E}}_{\mathsf{NI}}$:

  On receiving a commitment c and identity id, $\widehat{o\mathcal{E}}_{\mathsf{NI}}$ first verifies the NIWI proof and outputs $\perp$ if the proof is not accepting. Otherwise, it runs the extractor $o\mathcal{E}_1$ on $c1$ and outputs the extracted value v'.

Theorem 28 *$\langle \widehat{C}, \widehat{R} \rangle$ is a non-interactive, (d_2, S_{CRH})-over-extractable, perfectly binding commitment scheme for identities of length $t(n)$. Furthermore, it is concurrent d_4-non-malleable (w.r.t. commitment) and non-malleable w.r.t. extraction by extractor $\widehat{o\mathcal{E}}_{\mathsf{NM}}$ against uniform adversaries.*

It is easy to see that $\langle \widehat{C}, \widehat{R} \rangle$ is perfectly binding and (d_2, S_{CRH})-over-extractable. The non-malleability properties follow syntactically from the same proof as that of Theorem 21 and 22 w.r.t. the 2-round security strengthening technique in Section 5.5.3. The only slight difference is that when reducing to the collision resistance of the hash function, and the non-malleability w.r.t. extraction of the input commitment scheme, we need to ensure that the reduction is a uniform Turing machine, which can be done easily. More specifically, in Section 5.5.3,

- we rely on the collision resistance of hash functions in order to show that Invariant 2 holds in hybrid $H_0(b)$ (Claim 11), and

- we rely on the non-malleability w.r.t. extraction of the input commitment scheme in order to show that Invariant 2 holds in $H_3(b)$ (Claim 16) and that the emim random variable is indistinguishable in $H_2(b)$ and $H_3(b)$ (Claim 17).

We now observe that the reductions presented in the proof of Claim 11, 16 and 17 can be made uniform. First, these reductions run internally 1) the adversary, 2) the extractors for different commitment schemes, 3) possibly a strategy for finding collisions (for the second bullet point), and some other computations, all of which can be implemented using uniform Turing machines. Furthermore, these reductions have one value hardwired in — the index k of a "special" right interaction. When adapting to the uniform setting, since there are only $m = \mathsf{poly}(n)$ number of right interactions, instead of hard-wiring k, the reduction can simply guess k at random, at the cost of losing a factor of m in its advantage. Therefore, by essentially the same proof, we can show the same in the uniform setting. We hence omit the complete proof.

Robust CCA-security. We next show that the commitment scheme $\langle \widehat{C}, \widehat{R} \rangle$ is also robust-CCA secure against uniform adversaries.

Theorem 29 *$\langle \widehat{C}, \widehat{R} \rangle$ is d_4-CCA-secure and $(d_4, S_{\mathsf{CRH}}, \kappa(n))$-robust w.r.t. committed value oracle against uniform adversaries.*

The proof of d_4-CCA security is identical to the proof of the CCA-security w.r.t. the 2-round strengthening technique as described in Section 5.8.3, except a slight difference. The difference is identical as in the above proof of non-malleability against uniform adversaries, that is, to deal with the uniform collision resistance of hash function and uniform one-one non-malleability w.r.t. extraction of the input commitment scheme As observed earlier, we need to ensure that the reductions are uniform Turning machines which can be easily done as described above. The proof of $(d_4, S_{\mathsf{CRH}}, \kappa(n))$-robustness also follows from the proof of robustness described in Section 5.8.3 except that the simulator S also needs to be a uniform Turing machine which also by the same argument can be

made uniform. Therefore, by essentially the same proof as of Theorem 27, we can show that $\langle \widehat{C}, \widehat{R} \rangle$ is robust-CCA secure and omit a full proof.

5.10 Proof of Theorem 23

Proof: Recall that we want to show the following,

1. If $\langle \widehat{C}, \widehat{R} \rangle$ is $\mathcal{C}$-one-many non-malleable then it is $\mathcal{C}$-concurrent non-malleable.
2. If $\langle \widehat{C}, \widehat{R} \rangle$ is $\mathcal{C}$-one-many non-malleable w.r.t. extraction (by extractor $\widehat{o\mathcal{E}_{\mathsf{NM}}}$) then it is $\mathcal{C}$-concurrent non-malleable w.r.t. extraction (by $\widehat{o\mathcal{E}_{\mathsf{NM}}}$).

We begin by proving the second implication, that is, $\mathcal{C}$-one-many non-malleability w.r.t. extraction implies $\mathcal{C}$-concurrent non-malleability w.r.t. extraction. Let us assume for contradiction that there exists a non-uniform adversary $A = \{A_n\}_{n\in\mathbb{N}} \in \mathcal{C}$ that participates in $m = \mathsf{poly}(n)$ concurrent interactions while sending/receiving commitments to $\alpha = \mathsf{poly}(n)$-bit values, a non-uniform distinugisher $D = \{D_n\}_{n\in\mathbb{N}} \in \mathcal{P}/\mathrm{poly}$, and a polynomial $p(\cdot)$ such that for infinitely many $n \in \mathbb{N}$,

$$\left|\mathsf{Pr}[D_n(\mathsf{emim}^A_{\langle \widehat{C},\widehat{R}\rangle}(1^n, 0))] - \mathsf{Pr}[D_n(\mathsf{emim}^A_{\langle \widehat{C},\widehat{R}\rangle}(1^n, 1))]\right| > \frac{1}{p(n)} \ . \tag{5.32}$$

Fix some generic n for which this happens. We next consider a sequence of hybrid MIM experiments $\{H_i\}_{0\le i\le m-1}$. In the honest MIM experiment $\mathsf{MIM}^A_{\langle \widehat{C},\widehat{R}\rangle}(b)$ (for $b \in \{0,1\}$), A participates in m right interactions with $\widehat{R}$ and m left interactions with $\widehat{C}$. Recall that in all left interactions $i \in [m]$, A first chooses the identity id_i and challenge values (v_i^0, v_i^1), and interacts with $\widehat{C}$ to receive a commitment to value v_i^b with identity id_i. The hybrids H_i's we consider are identical to the MIM experiment $\mathsf{MIM}^A_{\langle \widehat{C},\widehat{R}\rangle}(0)$ except that for all the left interactions $j \le i$ in H_i, A receives a commitment to the value v_j^1 instead of commitments to v_j^0. We let $\mathsf{emim}^A_{H_i}$ denote the random variable that describes

the view of A and the values extracted from the right interactions in H_i by extractor $\widehat{o\mathcal{E}}_{\mathsf{NM}}$. It is easy to see that H_0 is identical to MIM experiment $\mathsf{MIM}^{A}_{\langle\widehat{C},\widehat{R}\rangle}(0)$ (hence $\mathsf{emim}^{A}_{\langle\widehat{C},\widehat{R}\rangle}(0) = \mathsf{emim}^{A}_{H_0}$) and H_m is identical to the MIM experiment $\mathsf{MIM}^{A}_{\langle\widehat{C},\widehat{R}\rangle}(1)$ (hence $\mathsf{emim}^{A}_{\langle\widehat{C},\widehat{R}\rangle}(1) = \mathsf{emim}^{A}_{H_m}$). By a standard hybrid argument, following Equation 5.32, there exists some $i \in \{0,\ldots,m-1\}$ such that,

$$\left|\mathsf{Pr}[D_n(\mathsf{emim}^{A}_{H_i})] - \mathsf{Pr}[D_n(\mathsf{emim}^{A}_{H_{i+1}})]\right| > \frac{1}{p(n)\cdot m} \ . \tag{5.33}$$

Given this, we construct a one-many non-uniform adversary $\widetilde{A} = \{\widetilde{A}_n\}_{n\in\mathbb{N}}$ for $\langle\widehat{C},\widehat{R}\rangle$ and a distinguisher $\widetilde{D} = \{\widetilde{D}_n\}_{n\in\mathbb{N}}$ that violate one-many non-malleability w.r.t. extraction of $\langle\widehat{C},\widehat{R}\rangle$ with advantage $1/(p(n)\cdot m(n))$. For $n\in\mathbb{N}$, $\widetilde{A}_n$ with index i (as defined above) hard-wired in it, participates in one left interaction with $\widehat{C}$ and m right interactions with $\widehat{R}$ and internally emulates an execution of H_i for A_n as follows: all right interactions of A_n are externally forwarded to $\widehat{R}$, the i-th left interaction of A_n is externally forwarded to $\widehat{C}$, and for all remaining left interactions $\widetilde{A}$ internally acts as a honest committer emulating hybrid H_i. More precisely, for the i-th left interaction, A_n forwards the identity id_i and values (v_i^0, v_i^1) sent by A_n to $\widehat{C}$ and receives a commitment to either v_i^0 or v_i^1, which $\widetilde{A}_n$ forwards to A_n as its i-th left commitment. The distinguisher $\widetilde{D}_n$ on input $\mathsf{emim}^{\widetilde{A}}_{\langle\widehat{C},\widehat{R}\rangle}(b)$, that is, the view $view$ of $\widetilde{A}_n$ and the values $u'_1,\ldots,u'_m$ extracted from the right interactions, runs the function $\mathsf{reconstruct}$ that reconstructs the view $view'$ of A in emulation by $\widetilde{A}$ and sets $\tilde{u}_k = u'_k$ iff A did not copy the identity of any of the m left interactions, and $\perp$ otherwise. $\mathsf{reconstruct}$ finally outputs $\tilde{u_1},\ldots,\tilde{u_m}, view'$. By construction it follows that,

$$\mathsf{reconstruct}(\mathsf{emim}^{\widetilde{A}}_{\langle\widehat{C},\widehat{R}\rangle}(0)) = \mathsf{emim}^{A}_{H_i} \ ; \ \mathsf{reconstruct}(\mathsf{emim}^{\widetilde{A}}_{\langle\widehat{C},\widehat{R}\rangle}(1)) = \mathsf{emim}^{A}_{H_{i+1}} \ .$$

The distinguisher $\widetilde{D}_n$ runs the distinguisher D_n on $\tilde{u}_1,\ldots,\tilde{u}_m, view'$ and outputs whatever D_n outputs. Then by Equation 5.33 it follows that the pair $(\widetilde{A},\widetilde{D})$ breaks the one-many

non-malleability of $\langle \widehat{C}, \widehat{R} \rangle$ w.r.t. extraction with advantage $1/(p(n) \cdot m(n))$. To arrive at a contradiction, we need to show that $\widetilde{A}$ and $\widetilde{D}$ belong to appropriate circuit classes. Firstly, note that $\widetilde{A}$ internally runs A and the rest of the computation can be done in $\mathsf{poly}(n)$-time. Therefore, the size/depth of $\widetilde{A}$ blows up only by an additive $\mathsf{poly}(n)$ factor over the size/depth of A. Secondly, $\widetilde{D}$ computes the $\mathsf{reconstruct}$ function, runs D and the rest of the computation can be done in poly-time. Note that the $\mathsf{reconstruct}$ function is in fact computable in poly-time. Therefore, the size/depth of $\widetilde{A}$ blows up only by an additive $\mathsf{poly}(n)$ factor over the size/depth of A. Since $A \in \mathcal{C}$ and $D \in \mathcal{P}/\text{poly}$ and both $\mathcal{C}$ and $\mathcal{P}/\text{poly}$ are closed under composition with $\mathcal{P}/\text{poly}$, we conclude that $\widetilde{A} \in \mathcal{C}$ and $\widetilde{D} \in \mathcal{P}/\text{poly}$. This contradicts the one-many non-malleability w.r.t. extraction of $\langle \widehat{C}, \widehat{R} \rangle$.

The proof of concurrent non-malleability w.r.t. commitment follows syntactically from the proof of non-malleability w.r.t. extraction except that we consider the random variable $\mathsf{mim}^{A}_{\langle \widehat{C}, \widehat{R} \rangle}$ instead of $\mathsf{emim}^{A}_{\langle \widehat{C}, \widehat{R} \rangle}$. We skip the formal proof. ■

Bibliography

[1] S. Myers, *Black-box composition does not imply adaptive security*, in *EUROCRYPT 2004* (C. Cachin and J. Camenisch, eds.), vol. 3027 of *LNCS*, pp. 189–206, Springer, Heidelberg, May, 2004.

[2] U. M. Maurer and K. Pietrzak, *Composition of random systems: When two weak make one strong*, in *TCC 2004* (M. Naor, ed.), vol. 2951 of *LNCS*, pp. 410–427, Springer, Heidelberg, Feb., 2004.

[3] K. Pietrzak, *Composition implies adaptive security in minicrypt*, in *EUROCRYPT 2006* (S. Vaudenay, ed.), vol. 4004 of *LNCS*, pp. 328–338, Springer, Heidelberg, May / June, 2006.

[4] K. Pietrzak, *Composition does not imply adaptive security*, in *CRYPTO 2005* (V. Shoup, ed.), vol. 3621 of *LNCS*, pp. 55–65, Springer, Heidelberg, Aug., 2005.

[5] U. M. Maurer, K. Pietrzak, and R. Renner, *Indistinguishability amplification*, in *CRYPTO 2007* (A. Menezes, ed.), vol. 4622 of *LNCS*, pp. 130–149, Springer, Heidelberg, Aug., 2007.

[6] C. Cho, C.-K. Lee, and R. Ostrovsky, *Equivalence of uniform key agreement and composition insecurity*, in *CRYPTO 2010* (T. Rabin, ed.), vol. 6223 of *LNCS*, pp. 447–464, Springer, Heidelberg, Aug., 2010.

[7] I. Berman and I. Haitner, *From non-adaptive to adaptive pseudorandom functions*, *Journal of Cryptology* **28** (Apr., 2015) 297–311.

[8] I. Berman, I. Haitner, I. Komargodski, and M. Naor, *Hardness-preserving reductions via cuckoo hashing*, *Journal of Cryptology* **32** (Apr., 2019) 361–392.

[9] B. Applebaum and P. Raykov, *Fast pseudorandom functions based on expander graphs*, in *TCC 2016-B, Part I* (M. Hirt and A. D. Smith, eds.), vol. 9985 of *LNCS*, pp. 27–56, Springer, Heidelberg, Oct. / Nov., 2016.

[10] S. Vaudenay, *Decorrelation: A theory for block cipher security*, *Journal of Cryptology* **16** (Sept., 2003) 249–286.

[11] M. N. Wegman and L. Carter, *New hash functions and their use in authentication and set equality*, *Journal of Computer and System Sciences* **22** (1981) 265–279.

[12] D. R. Stinson, *Universal hashing and authentication codes*, in *CRYPTO'91* (J. Feigenbaum, ed.), vol. 576 of *LNCS*, pp. 74–85, Springer, Heidelberg, Aug., 1992.

[13] M. Bellare, R. Canetti, and H. Krawczyk, *Pseudorandom functions revisited: The cascade construction and its concrete security*, in *37th FOCS*, pp. 514–523, IEEE Computer Society Press, Oct., 1996.

[14] T. Holenstein and M. Sinha, *Constructing a pseudorandom generator requires an almost linear number of calls*, in *53rd FOCS*, pp. 698–707, IEEE Computer Society Press, Oct., 2012.

[15] B. Barak and M. Mahmoody-Ghidary, *Lower bounds on signatures from symmetric primitives*, in *48th FOCS*, pp. 680–688, IEEE Computer Society Press, Oct., 2007.

[16] R. Gennaro and L. Trevisan, *Lower bounds on the efficiency of generic cryptographic constructions*, in *41st FOCS*, pp. 305–313, IEEE Computer Society Press, Nov., 2000.

[17] R. Gennaro, Y. Gertner, and J. Katz, *Lower bounds on the efficiency of encryption and digital signature schemes*, in *35th ACM STOC*, pp. 417–425, ACM Press, June, 2003.

[18] J. Bronson, A. Juma, and P. A. Papakonstantinou, *Limits on the stretch of non-adaptive constructions of pseudo-random generators*, in *TCC 2011* (Y. Ishai, ed.), vol. 6597 of *LNCS*, pp. 504–521, Springer, Heidelberg, Mar., 2011.

[19] E. Viola, "On constructing parallel pseudorandom generators from one-way functions." Cryptology ePrint Archive, Report 2005/159, 2005. http://eprint.iacr.org/2005/159.

[20] E. Miles and E. Viola, *On the complexity of non-adaptively increasing the stretch of pseudorandom generators*, in *TCC 2011* (Y. Ishai, ed.), vol. 6597 of *LNCS*, pp. 522–539, Springer, Heidelberg, Mar., 2011.

[21] M. Luby and C. Rackoff, *How to construct pseudorandom permutations from pseudorandom functions*, *SIAM Journal on Computing* **17** (1988), no. 2.

[22] M. Naor and O. Reingold, *On the construction of pseudo-random permutations: Luby-Rackoff revisited (extended abstract)*, in *29th ACM STOC*, pp. 189–199, ACM Press, May, 1997.

[23] V. T. Hoang and P. Rogaway, *On generalized Feistel networks*, in *CRYPTO 2010* (T. Rabin, ed.), vol. 6223 of *LNCS*, pp. 613–630, Springer, Heidelberg, Aug., 2010.

[24] G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche, *On the indifferentiability of the sponge construction*, in *EUROCRYPT 2008* (N. P. Smart, ed.), vol. 4965 of *LNCS*, pp. 181–197, Springer, Heidelberg, Apr., 2008.

[25] P. Rogaway and J. P. Steinberger, *Security/efficiency tradeoffs for permutation-based hashing*, in *EUROCRYPT 2008* (N. P. Smart, ed.), vol. 4965 of *LNCS*, pp. 220–236, Springer, Heidelberg, Apr., 2008.

[26] J.-P. Aumasson, P. Jovanovic, and S. Neves, "NORX8 and NORX16: Authenticated encryption for low-end systems." Cryptology ePrint Archive, Report 2015/1154, 2015. http://eprint.iacr.org/2015/1154.

[27] G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche, *Sponge-based pseudo-random number generators*, in *CHES 2010* (S. Mangard and F.-X. Standaert, eds.), vol. 6225 of *LNCS*, pp. 33–47, Springer, Heidelberg, Aug., 2010.

[28] P. Gazi and S. Tessaro, *Provably robust sponge-based PRNGs and KDFs*, in *EUROCRYPT 2016, Part I* (M. Fischlin and J.-S. Coron, eds.), vol. 9665 of *LNCS*, pp. 87–116, Springer, Heidelberg, May, 2016.

[29] M. Bellare, V. T. Hoang, S. Keelveedhi, and P. Rogaway, *Efficient garbling from a fixed-key blockcipher*, in *2013 IEEE Symposium on Security and Privacy*, pp. 478–492, IEEE Computer Society Press, May, 2013.

[30] J.-S. Coron, J. Patarin, and Y. Seurin, *The random oracle model and the ideal cipher model are equivalent*, in *CRYPTO 2008* (D. Wagner, ed.), vol. 5157 of *LNCS*, pp. 1–20, Springer, Heidelberg, Aug., 2008.

[31] T. Holenstein, R. Künzler, and S. Tessaro, *The equivalence of the random oracle model and the ideal cipher model, revisited*, in *43rd ACM STOC* (L. Fortnow and S. P. Vadhan, eds.), pp. 89–98, ACM Press, June, 2011.

[32] E. Andreeva, A. Bogdanov, Y. Dodis, B. Mennink, and J. P. Steinberger, *On the indifferentiability of key-alternating ciphers*, in *CRYPTO 2013, Part I* (R. Canetti and J. A. Garay, eds.), vol. 8042 of *LNCS*, pp. 531–550, Springer, Heidelberg, Aug., 2013.

[33] Y. Dodis, M. Stam, J. P. Steinberger, and T. Liu, *Indifferentiability of confusion-diffusion networks*, in *EUROCRYPT 2016, Part II* (M. Fischlin and J.-S. Coron, eds.), vol. 9666 of *LNCS*, pp. 679–704, Springer, Heidelberg, May, 2016.

[34] D. Dachman-Soled, J. Katz, and A. Thiruvengadam, *10-round Feistel is indifferentiable from an ideal cipher*, in *EUROCRYPT 2016, Part II* (M. Fischlin and J.-S. Coron, eds.), vol. 9666 of *LNCS*, pp. 649–678, Springer, Heidelberg, May, 2016.

[35] Y. Dai and J. P. Steinberger, *Indifferentiability of 8-round Feistel networks*, in *CRYPTO 2016, Part I* (M. Robshaw and J. Katz, eds.), vol. 9814 of *LNCS*, pp. 95–120, Springer, Heidelberg, Aug., 2016.

[36] U. M. Maurer, R. Renner, and C. Holenstein, *Indifferentiability, impossibility results on reductions, and applications to the random oracle methodology*, in *TCC 2004* (M. Naor, ed.), vol. 2951 of *LNCS*, pp. 21–39, Springer, Heidelberg, Feb., 2004.

[37] M. Bellare, V. T. Hoang, and S. Keelveedhi, *Instantiating random oracles via UCEs*, in *CRYPTO 2013, Part II* (R. Canetti and J. A. Garay, eds.), vol. 8043 of *LNCS*, pp. 398–415, Springer, Heidelberg, Aug., 2013.

[38] P. Soni and S. Tessaro, *Public-seed pseudorandom permutations*, in *EUROCRYPT 2017, Part II* (J.-S. Coron and J. B. Nielsen, eds.), vol. 10211 of *LNCS*, pp. 412–441, Springer, Heidelberg, Apr. / May, 2017.

[39] C. Brzuska, P. Farshim, and A. Mittelbach, *Indistinguishability obfuscation and UCEs: The case of computationally unpredictable sources*, in *CRYPTO 2014, Part I* (J. A. Garay and R. Gennaro, eds.), vol. 8616 of *LNCS*, pp. 188–205, Springer, Heidelberg, Aug., 2014.

[40] Y. Dodis, C. Ganesh, A. Golovnev, A. Juels, and T. Ristenpart, *A formal treatment of backdoored pseudorandom generators*, in *EUROCRYPT 2015, Part I* (E. Oswald and M. Fischlin, eds.), vol. 9056 of *LNCS*, pp. 101–126, Springer, Heidelberg, Apr., 2015.

[41] T. Matsuda and G. Hanaoka, *Chosen ciphertext security via UCE*, in *PKC 2014* (H. Krawczyk, ed.), vol. 8383 of *LNCS*, pp. 56–76, Springer, Heidelberg, Mar., 2014.

[42] M. Bellare and V. T. Hoang, *Resisting randomness subversion: Fast deterministic and hedged public-key encryption in the standard model*, in *EUROCRYPT 2015, Part II* (E. Oswald and M. Fischlin, eds.), vol. 9057 of *LNCS*, pp. 627–656, Springer, Heidelberg, Apr., 2015.

[43] M. Bellare and I. Stepanovs, *Point-function obfuscation: A framework and generic constructions*, in *TCC 2016-A, Part II* (E. Kushilevitz and T. Malkin, eds.), vol. 9563 of *LNCS*, pp. 565–594, Springer, Heidelberg, Jan., 2016.

[44] M. Bellare, V. T. Hoang, and S. Keelveedhi, *Cryptography from compression functions: The UCE bridge to the ROM*, in *CRYPTO 2014, Part I* (J. A. Garay and R. Gennaro, eds.), vol. 8616 of *LNCS*, pp. 169–187, Springer, Heidelberg, Aug., 2014.

[45] J.-S. Coron, Y. Dodis, C. Malinaud, and P. Puniya, *Merkle-Damgård revisited: How to construct a hash function*, in *CRYPTO 2005* (V. Shoup, ed.), vol. 3621 of *LNCS*, pp. 430–448, Springer, Heidelberg, Aug., 2005.

[46] A. Mandal, J. Patarin, and Y. Seurin, *On the public indifferentiability and correlation intractability of the 6-round Feistel construction*, in *TCC 2012* (R. Cramer, ed.), vol. 7194 of *LNCS*, pp. 285–302, Springer, Heidelberg, Mar., 2012.

[47] J.-S. Coron, T. Holenstein, R. Künzler, J. Patarin, Y. Seurin, and S. Tessaro, *How to build an ideal cipher: The indifferentiability of the Feistel construction*, *Journal of Cryptology* **29** (Jan., 2016) 61–114.

[48] C. Brzuska and A. Mittelbach, *Using indistinguishability obfuscation via UCEs*, in *ASIACRYPT 2014, Part II* (P. Sarkar and T. Iwata, eds.), vol. 8874 of *LNCS*, pp. 122–141, Springer, Heidelberg, Dec., 2014.

[49] M. Zhandry, *The magic of ELFs*, in *CRYPTO 2016, Part I* (M. Robshaw and J. Katz, eds.), vol. 9814 of *LNCS*, pp. 479–508, Springer, Heidelberg, Aug., 2016.

[50] A. Mittelbach, *Salvaging indifferentiability in a multi-stage setting*, in *EUROCRYPT 2014* (P. Q. Nguyen and E. Oswald, eds.), vol. 8441 of *LNCS*, pp. 603–621, Springer, Heidelberg, May, 2014.

[51] D. Dolev, C. Dwork, and M. Naor, *Nonmalleable cryptography*, *SIAM Journal on Computing* **30** (2000), no. 2 391–437.

[52] R. Pass and A. Rosen, *Concurrent non-malleable commitments*, in *46th FOCS*, pp. 563–572, IEEE Computer Society Press, Oct., 2005.

[53] B. Barak, *Constant-round coin-tossing with a man in the middle or realizing the shared random string model*, in *43rd FOCS*, pp. 345–355, IEEE Computer Society Press, Nov., 2002.

[54] R. Pass and A. Rosen, *New and improved constructions of non-malleable cryptographic protocols*, in *37th ACM STOC* (H. N. Gabow and R. Fagin, eds.), pp. 533–542, ACM Press, May, 2005.

[55] H. Lin, R. Pass, and M. Venkitasubramaniam, *Concurrent non-malleable commitments from any one-way function*, in *TCC 2008* (R. Canetti, ed.), vol. 4948 of *LNCS*, pp. 571–588, Springer, Heidelberg, Mar., 2008.

[56] H. Lin and R. Pass, *Non-malleability amplification*, in *41st ACM STOC* (M. Mitzenmacher, ed.), pp. 189–198, ACM Press, May / June, 2009.

[57] O. Pandey, R. Pass, and V. Vaikuntanathan, *Adaptive one-way functions and applications*, in *CRYPTO 2008* (D. Wagner, ed.), vol. 5157 of *LNCS*, pp. 57–74, Springer, Heidelberg, Aug., 2008.

[58] R. Pass and H. Wee, *Constant-round non-malleable commitments from sub-exponential one-way functions*, in *EUROCRYPT 2010* (H. Gilbert, ed.), vol. 6110 of *LNCS*, pp. 638–655, Springer, Heidelberg, May / June, 2010.

[59] H. Wee, *Black-box, round-efficient secure computation via non-malleability amplification*, in *51st FOCS*, pp. 531–540, IEEE Computer Society Press, Oct., 2010.

[60] V. Goyal, *Constant round non-malleable protocols using one way functions*, in *43rd ACM STOC* (L. Fortnow and S. P. Vadhan, eds.), pp. 695–704, ACM Press, June, 2011.

[61] H. Lin and R. Pass, *Constant-round non-malleable commitments from any one-way function*, in *43rd ACM STOC* (L. Fortnow and S. P. Vadhan, eds.), pp. 705–714, ACM Press, June, 2011.

[62] V. Goyal, C.-K. Lee, R. Ostrovsky, and I. Visconti, *Constructing non-malleable commitments: A black-box approach*, in *53rd FOCS*, pp. 51–60, IEEE Computer Society Press, Oct., 2012.

[63] M. Ciampi, R. Ostrovsky, L. Siniscalchi, and I. Visconti, *Four-round concurrent non-malleable commitments from one-way functions*, in *CRYPTO 2017, Part II* (J. Katz and H. Shacham, eds.), vol. 10402 of *LNCS*, pp. 127–157, Springer, Heidelberg, Aug., 2017.

[64] M. Ciampi, R. Ostrovsky, L. Siniscalchi, and I. Visconti, *Concurrent non-malleable commitments (and more) in 3 rounds*, in *CRYPTO 2016, Part III* (M. Robshaw and J. Katz, eds.), vol. 9816 of *LNCS*, pp. 270–299, Springer, Heidelberg, Aug., 2016.

[65] V. Goyal, O. Pandey, and S. Richelson, *Textbook non-malleable commitments*, in *48th ACM STOC* (D. Wichs and Y. Mansour, eds.), pp. 1128–1141, ACM Press, June, 2016.

[66] D. Khurana, *Round optimal concurrent non-malleability from polynomial hardness*, in *TCC 2017, Part II* (Y. Kalai and L. Reyzin, eds.), vol. 10678 of *LNCS*, pp. 139–171, Springer, Heidelberg, Nov., 2017.

[67] R. Pass, *Unprovable security of perfect NIZK and non-interactive non-malleable commitments*, in *TCC 2013* (A. Sahai, ed.), vol. 7785 of *LNCS*, pp. 334–354, Springer, Heidelberg, Mar., 2013.

[68] V. Goyal, D. Khurana, and A. Sahai, *Breaking the three round barrier for non-malleable commitments*, in *57th FOCS* (I. Dinur, ed.), pp. 21–30, IEEE Computer Society Press, Oct., 2016.

[69] C. Dwork and M. Naor, *Zaps and their applications*, in *41st FOCS*, pp. 283–293, IEEE Computer Society Press, Nov., 2000.

[70] R. L. Rivest, A. Shamir, and D. A. Wagner, *Time-lock puzzles and timed-release crypto*, tech. rep., Massachusetts Institute of Technology, Cambridge, MA, USA, 1996.

[71] T. May, *Timed-release crypto*, .

[72] C. Dwork and M. Naor, *Pricing via processing or combatting junk mail*, in *CRYPTO'92* (E. F. Brickell, ed.), vol. 740 of *LNCS*, pp. 139–147, Springer, Heidelberg, Aug., 1993.

[73] M. Jakobsson and A. Juels, *Proofs of work and bread pudding protocols*, in *Proceedings of the IFIP TC6/TC11 Joint Working Conference on Secure Information Networks: Communications and Multimedia Security*, CMS '99, (Deventer, The Netherlands, The Netherlands), pp. 258–272, Kluwer, B.V., 1999.

[74] S. Nakamoto, *Bitcoin: A peer-to-peer electronic cash system, 2008*, .

[75] D. Boneh and M. Naor, *Timed commitments*, in *CRYPTO 2000* (M. Bellare, ed.), vol. 1880 of *LNCS*, pp. 236–254, Springer, Heidelberg, Aug., 2000.

[76] N. Bitansky, S. Goldwasser, A. Jain, O. Paneth, V. Vaikuntanathan, and B. Waters, *Time-lock puzzles from randomized encodings*, in *ITCS 2016* (M. Sudan, ed.), pp. 345–356, ACM, Jan., 2016.

[77] B. Barak, S. J. Ong, and S. P. Vadhan, *Derandomization in cryptography*, in *CRYPTO 2003* (D. Boneh, ed.), vol. 2729 of *LNCS*, pp. 299–315, Springer, Heidelberg, Aug., 2003.

[78] J. Groth, R. Ostrovsky, and A. Sahai, *Non-interactive zaps and new techniques for NIZK*, in *CRYPTO 2006* (C. Dwork, ed.), vol. 4117 of *LNCS*, pp. 97–111, Springer, Heidelberg, Aug., 2006.

[79] N. Bitansky and O. Paneth, *ZAPs and non-interactive witness indistinguishability from indistinguishability obfuscation*, in *TCC 2015, Part II* (Y. Dodis and J. B. Nielsen, eds.), vol. 9015 of *LNCS*, pp. 401–427, Springer, Heidelberg, Mar., 2015.

[80] B. Barak and R. Pass, *On the possibility of one-message weak zero-knowledge*, in *TCC 2004* (M. Naor, ed.), vol. 2951 of *LNCS*, pp. 121–132, Springer, Heidelberg, Feb., 2004.

[81] P. Rogaway, *Formalizing human ignorance*, in *Progress in Cryptology - VIETCRYPT 06* (P. Q. Nguyen, ed.), vol. 4341 of *LNCS*, pp. 211–228, Springer, Heidelberg, Sept., 2006.

[82] R. Canetti, H. Lin, and R. Pass, *Adaptive hardness and composable security in the plain model from standard assumptions*, in *51st FOCS*, pp. 541–550, IEEE Computer Society Press, Oct., 2010.

[83] H. Lin and R. Pass, *Black-box constructions of composable protocols without set-up*, in *CRYPTO 2012* (R. Safavi-Naini and R. Canetti, eds.), vol. 7417 of *LNCS*, pp. 461–478, Springer, Heidelberg, Aug., 2012.

[84] S. Kiyoshima, *Round-efficient black-box construction of composable multi-party computation*, in *CRYPTO 2014, Part II* (J. A. Garay and R. Gennaro, eds.), vol. 8617 of *LNCS*, pp. 351–368, Springer, Heidelberg, Aug., 2014.

[85] V. Goyal, H. Lin, O. Pandey, R. Pass, and A. Sahai, *Round-efficient concurrently composable secure computation via a robust extraction lemma*, in *TCC 2015, Part I* (Y. Dodis and J. B. Nielsen, eds.), vol. 9014 of *LNCS*, pp. 260–289, Springer, Heidelberg, Mar., 2015.

[86] Z. Brakerski, S. Halevi, and A. Polychroniadou, *Four round secure computation without setup*, in *TCC 2017, Part I* (Y. Kalai and L. Reyzin, eds.), vol. 10677 of *LNCS*, pp. 645–677, Springer, Heidelberg, Nov., 2017.

[87] P. Soni and S. Tessaro, *On the query complexity of constructing prfs from non-adaptive prfs*, 2020.

[88] P. Soni and S. Tessaro, *Naor-Reingold goes public: The complexity of known-key security*, in *EUROCRYPT 2018, Part III* (J. B. Nielsen and V. Rijmen, eds.), vol. 10822 of *LNCS*, pp. 653–684, Springer, Heidelberg, Apr. / May, 2018.

[89] H. Lin, R. Pass, and P. Soni, *Two-round and non-interactive concurrent non-malleable commitments from time-lock puzzles*, in *58th FOCS* (C. Umans, ed.), pp. 576–587, IEEE Computer Society Press, Oct., 2017.

[90] H. Lin, R. Pass, and P. Soni, *Two-round and non-interactive concurrent non-malleable commitments from time-lock puzzles*, *SIAM Journal on Computing* **49** (Jan., 2020) 196–279. Special issue for focs 2017.

[91] R. Impagliazzo and S. Rudich, *Limits on the provable consequences of one-way permutations*, in *CRYPTO'88* (S. Goldwasser, ed.), vol. 403 of *LNCS*, pp. 8–26, Springer, Heidelberg, Aug., 1990.

[92] O. Reingold, L. Trevisan, and S. P. Vadhan, *Notions of reducibility between cryptographic primitives*, in *TCC 2004* (M. Naor, ed.), vol. 2951 of *LNCS*, pp. 1–20, Springer, Heidelberg, Feb., 2004.

[93] Y. Gertner, T. Malkin, and O. Reingold, *On the impossibility of basing trapdoor functions on trapdoor predicates*, in *42nd FOCS*, pp. 126–135, IEEE Computer Society Press, Oct., 2001.

[94] D. R. Simon, *Finding collisions on a one-way street: Can secure hash functions be based on general assumptions?*, in *EUROCRYPT'98* (K. Nyberg, ed.), vol. 1403 of *LNCS*, pp. 334–345, Springer, Heidelberg, May / June, 1998.

[95] R. Impagliazzo and S. Rudich, *Limits on the provable consequences of one-way permutations*, in *21st ACM STOC*, pp. 44–61, ACM Press, May, 1989.

[96] A. Buldas, S. Laur, and M. Niitsoo, *Oracle separation in the non-uniform model*, in *ProvSec 2009* (J. Pieprzyk and F. Zhang, eds.), vol. 5848 of *LNCS*, pp. 230–244, Springer, Heidelberg, Nov., 2009.

[97] M. Bellare and P. Rogaway, *The security of triple encryption and a framework for code-based game-playing proofs*, in *EUROCRYPT 2006* (S. Vaudenay, ed.), vol. 4004 of *LNCS*, pp. 409–426, Springer, Heidelberg, May / June, 2006.

[98] P. Gazi and S. Tessaro, *Secret-key cryptography from ideal primitives: A systematic overview*, in *2015 IEEE Information Theory Workshop, ITW 2015, Jerusalem, Israel, April 26 - May 1, 2015*, pp. 1–5, IEEE, 2015.

[99] M. Bellare, D. J. Bernstein, and S. Tessaro, *Hash-function based PRFs: AMAC and its multi-user security*, in *EUROCRYPT 2016, Part I* (M. Fischlin and J.-S. Coron, eds.), vol. 9665 of *LNCS*, pp. 566–595, Springer, Heidelberg, May, 2016.

[100] B. Barak and M. Mahmoody-Ghidary, *Merkle puzzles are optimal - an $O(n^2)$-query attack on any key exchange from a random oracle*, in *CRYPTO 2009* (S. Halevi, ed.), vol. 5677 of *LNCS*, pp. 374–390, Springer, Heidelberg, Aug., 2009.

[101] J. Patarin, *New results on pseudorandom permutation generators based on the DES scheme*, in *CRYPTO'91* (J. Feigenbaum, ed.), vol. 576 of *LNCS*, pp. 301–312, Springer, Heidelberg, Aug., 1992.

[102] S. Chen and J. P. Steinberger, *Tight security bounds for key-alternating ciphers*, in *EUROCRYPT 2014* (P. Q. Nguyen and E. Oswald, eds.), vol. 8441 of *LNCS*, pp. 327–350, Springer, Heidelberg, May, 2014.

[103] U. Feige, D. Lapidot, and A. Shamir, *Multiple non-interactive zero knowledge proofs based on a single random string (extended abstract)*, in *31st FOCS*, pp. 308–317, IEEE Computer Society Press, Oct., 1990.

[104] D. Khurana and A. Sahai, *How to achieve non-malleability in one or two rounds*, in *58th FOCS* (C. Umans, ed.), pp. 564–575, IEEE Computer Society Press, Oct., 2017.

[105] N. Bitansky and H. Lin, *One-message zero knowledge and non-malleable commitments*, in *Theory of Cryptography* (A. Beimel and S. Dziembowski, eds.), (Cham), pp. 209–234, Springer International Publishing, 2018.

[106] M. Ball, D. Dachman-Soled, M. Kulkarni, H. Lin, and T. Malkin, "Non-malleable codes against bounded polynomial time tampering." Cryptology ePrint Archive, Report 2018/1015, 2018. https://eprint.iacr.org/2018/1015.

[107] O. Goldreich and L. A. Levin, *A hard-core predicate for all one-way functions*, in *Proceedings of the Twenty-first Annual ACM Symposium on Theory of Computing*, STOC '89, (New York, NY, USA), pp. 25–32, ACM, 1989.

[108] M. Bellare and M. Yung, *Certifying permutations: Noninteractive zero-knowledge based on any trapdoor permutation*, *Journal of Cryptology* **9** (June, 1996) 149–166.

[109] D. Boneh and M. K. Franklin, *Identity based encryption from the Weil pairing*, *SIAM Journal on Computing* **32** (2003), no. 3 586–615.

[110] J. A. Garay, P. D. MacKenzie, M. Prabhakaran, and K. Yang, *Resource fairness and composability of cryptographic protocols*, *Journal of Cryptology* **24** (Oct., 2011) 615–658.