

Near-Optimal Non-malleable Codes and Leakage Resilient Secret Sharing Schemes

A THESIS
SUBMITTED FOR THE DEGREE OF
Doctor of Philosophy
IN THE
Faculty of Mathematical Sciences

BY
Sruthi Sekar



Department of Mathematics
Indian Institute of Science
Bangalore – 560 012 (INDIA)

May, 2022

Declaration of Originality

I, **Sruthi Sekar**, with SR No. **10-06-00-10-31-14-1-11378** hereby declare that the material presented in the thesis titled

Near-Optimal Non-malleable Codes and Leakage Resilient Secret Sharing Schemes

represents original work carried out by me in the **Department of Mathematics** at **Indian Institute of Science** during the years **2016-2021**.

With my signature, I certify that:

- I have not manipulated any of the data or results.
- I have not committed any plagiarism of intellectual property. I have clearly indicated and referenced the contributions of others.
- I have explicitly acknowledged all collaborative research and discussions.
- I have understood that any false claim will result in severe disciplinary action.
- I have understood that the work may be screened for any form of academic misconduct.

Date:

Student Signature

In my capacity as supervisor of the above-mentioned work, I certify that the above statements are true to the best of my knowledge, and I have carried out due diligence to ensure the originality of the report.

Advisor Name:

Advisor Signature

© Sruthi Sekar

May, 2022

All rights reserved

DEDICATED TO

My Amma, Appa, and

All the wonderful mentors and teachers I have had in my life.

Acknowledgements

First and foremost, I would like to thank my advisor Dr. Bhavana Kanukurthi for constantly inspiring me, and guiding me through not just my research, but also other aspects of my life. My journey with her began in 2017 when I took the best decision of my life to join her for a Master's project, which soon transformed into a desire to continue for a Ph.D. under her guidance. What began as a mere admiration for Cryptography, after taking a coursera course by Prof. Dan Boneh on "Basic Cryptography", soon transformed into absolute love for the subject and a desire to learn more and more. I will cherish my days with Bhavana at IISc and really miss the long research discussions, the nights before deadlines at her house, the wonderful game nights, the time spent with Avartana (her daughter), the amazing dinner/lunch outings, and the everyday coffee discussions at Prakruthi.

Secondly, I would also like to sincerely thank Prof. Manjunath Krishnapur, who is my co-advisor from the Mathematics Department. The first course that made me want to continue research in mathematics was probability theory, which was taught by him. Without him, my Ph.D. journey at IISc would've even have started. He has been a constant source of inspiration for me, be it his motivating words during times of rejection, or the way he teaches. My desire to not just be a good researcher, but also be a wonderful teacher stems from watching him teach.

Besides my advisors, I have been fortunate enough to have two other great mentors, which include Prof. Gautam Bharali, whose advice at different stages of my academic journey has been invaluable, and Dr. Nishanth Chandran, who has been as much a mentor to me and as inspiring as my other Ph.D. advisors. Nishanth's guidance through the research projects, and his belief that "many papers are only due to relentless pushing for the right answer" has been a great source of motivation for me.

I am extremely grateful to Prof. Sanjam Garg, who hosted me at UC Berkeley for 4 months in summer 2018. It was an enriching experience, and I am looking forward to the next academic journey with him. I am also truly thankful to Prof. Yuval Ishai, for hosting me at Technion-Israel Institute of Technology amidst all the pandemic uncertainties in summer 2021. Further, I owe my deepest gratitude to Prof. Manoj Prabhakaran, for giving me the chance to visit him

Acknowledgements

at IIT Bombay, to Prof. Rafail Ostrovski, Prof. Brent Waters and Dr. Pratyay Mukherjee, for inviting me to UCLA, UT Austin and VISA research, respectively, for a wonderful discussion and an invited talk. Besides this, Microsoft Research, Bangalore has played a great role in providing the conducive environment for wonderful reading groups, and fruitful collaborations. Each of these opportunities have been a big learning experience for me, and I wish to express my heartfelt gratitude to each one of my collaborators in this journey, including Prof. Sanjam Garg, Prof. Mohammad Mahmoody, Dr. Mohammad Hajiabadi, Ahmedreza Rahimi, Prof. Yuval Ishai, Dr. Eshan Chattopadhyay, Dr. Divesh Aggarwal, Dr. Maciej Obremski, Dr. Divya Gupta, Akash Shah, Nishka Dasgupta. I also wish to thank Prof. Vinod Vaikuntanathan for brainstorming with us on a dream problem in Crypto for several weeks.

The biggest support for any researcher comes from their research lab, and it's no different for me. I wish to thank my fellow labmates and my source of strength, from the Cryptography Security and Privacy (CrySP) group, including Marilyn, Nithin, Cressida, Rohit, Abhishek, Jenit, Anirban, Shreyas, and last but not the least, my collaborator and partner in crime throughout my research journey, Sai Lakshmi Bhavana Obbattu. Special thanks to three amazing friends, who were my support system during my visit at Berkeley: Romain, who is not just a great friend, but also a genuine advisor for me about general life decisions, Mohammad, and Xiao, both of whom have been true friends and superfun to hang out with. The support from some great teachers/mentors at the Department of Computer Science and Automation, including Dr. Arpita Patra, Prof. Sanjit Chatterjee, Dr. Siddharth Barman, Dr. Chandan Saha, Prof. Narahari, Dr. Anand Louis, Dr. Arindam Khan, Dr. Rahul Saladi, and Dr. Chaya Ganesh, has been truly encouraging and I wish to sincerely thank each of them. My sincere acknowledgements to TCS for the research grant to support my Ph.D., the institute GARP sponsorship and IACR for supporting my travel to conferences, the Department of Mathematics, IISc, for honoring me with the Prof. K. Venkatchalaenger Scholarship, and finally to Technion-Israel Institute of Technology, for supporting my research visit with the Sandwich Scholarship for International Students.

Finally, I wish to express my love and gratitude, and thank the people who are my family, my friends, and the ones who kept me sane through these years. This includes my biggest cheerleaders– my Amma and Appa, my brother Vishrudh, my sisters Mukthi and Sahithya, and the friends, who are like family to me– Nimisha, Rahul, Saahil, Anupam, Divya, and Swati. I have been fortunate enough to be surrounded by so many friends who have been a part of different stages of my life, including Garima, Vijayluxmi, Megha, Harshita, Sukriti, Pratik, Ramesh, Anamay, Milind, Sandip, and Mukul. Last but not the least, I wish to thank a very special person– my fiance, Pavithran, for standing by me and always inspiring me.

Abstract

A well-studied class of attacks on cryptosystems called “*side-channel attacks*”, stems from the additional access that an adversary can get due to the susceptibility of the hardware on which the cryptosystem (e.g., digital signatures, encryption schemes) is implemented. Particularly, such attacks allow the adversary to get some form of side-channel information about the secret, in addition to the actual allowed input-output access to the device on which the cryptosystem is being implemented. Two such classes of side-channel attacks that have received tremendous attention in the literature are *leakage attacks*, which give the adversary some extra bits of information about the secret, and *tampering attacks*, which allow the adversary to tamper with the device storing the secret and observe additional input-output behaviour on the tampered secret. It is extremely difficult to create hardware that is immune to such side-channel attacks, and hence, a lot of recent research focuses on building algorithmic defenses against such attacks. Two such important well-studied primitives are: *non-malleable codes*, which help against tampering attacks, and *leakage resilient secret sharing schemes*, which help against leakage attacks. In this thesis, we study these two fundamental objects and strive to build them with “optimal” parameters.

Dziembowski, Pietrzak, and Wichs introduced non-malleable codes (NMCs) at ITCS 2010, with the intent to secure against tampering attacks. NMCs give a guarantee that adversarial tampering of the encoding of the secret will lead to a tampered secret, which is either same as the original or completely independent of it, thus giving no additional information to the adversary.

Secret sharing schemes, introduced by Shamir and Blakely in 1979, help a party, called a dealer, to share his secret message amongst N parties in such a way that any t of these parties can combine their shares to recover the secret, but the secret remains hidden from an adversary corrupting $< t$ parties to get their complete shares. To secure against leakage attacks, which allow the adversary to get some bounded bits of leakage from the shares of the remaining parties, in addition to the complete shares, Dziembowski and Pietrzak introduced the notion of leakage resilient secret sharing schemes (LRSS), at FOCS 2007.

Whether we consider using NMCs and storing the non-malleable encoding of the message on a tamper-prone system, or we use an LRSS to generate shares of a secret to be stored on a leakage-prone system, it is important to build schemes that output codewords/shares that are of optimal length and do not introduce too much redundancy into the codewords/shares. This is, in particular, captured by the rate of the schemes, which is the ratio of the message length to the codeword length/largest share length. The research goal of this thesis is to improve the state of art on rates of these schemes and get near-optimal/optimal rates. We elaborate on the results below.

- In the first part of our thesis, we look at NMCs in a natural and well-studied model of tampering called the *2-split-state* model, where the codeword consists of two independently tamperable states. In this model, Cheraghchi and Guruswami (ITCS 2014) showed that one cannot build NMCs with a rate better than $1/2$. Since its inception, this area has witnessed huge strides leading to the construction of a constant-rate NMC in the 2-split state model due to Aggarwal and Obremski at FOCS 2020. However, the rate of this construction – roughly $1/1,500,000$ – is nowhere close to the best achievable rate of $1/2$! In this thesis, we dramatically improve this status quo by building an NMC in the 2-split-state model with a near-optimal rate of $1/3$. As a stepping stone, we also show how one can get a rate of $1/2$, if one relaxes the non-malleability requirement from NMCs to only uniform random messages.
- In the second part of our thesis, we study LRSS schemes, where on the one hand, a huge line of prior works focus on improving the leakage model but suffer from poor rate– in the strongest leakage model, called *adaptive and joint*, the best-known rate of $\mathcal{O}(1/N)$ (where N is the number of parties) is achieved by the LRSS scheme of Chattopadhyay, Goodman, Goyal, Kumar, Li, Meka, Zuckerman from FOCS 2020 – on the other hand, a line of works focus on improving the rate of the LRSS, which come at the expense of the leakage model– in the weakest leakage model, called *non-adaptive and independent*, the work of Srinivasan and Vasudevan from CRYPTO 2019 achieves a rate of $1/3$. In this thesis, we strive to bridge this gap, and build an LRSS scheme in the strong *adaptive and joint* leakage model, while still getting a much better rate of $\mathcal{O}(1)$. Further, we also improve the *non-adaptive and independent* LRSS scheme of Srinivasan and Vasudevan, and show how to get an optimal rate of 1.

Publications based on this Thesis

Accepted Papers:

- [C1] Bhavana Kanukurthi, Sai Lakshmi Bhavana Obbattu, and **Sruthi Sekar**: Non-malleable Randomness Encoders and their Applications.
In: Advances in Cryptology- EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part III, pages 589–617, 2018.
- [C2] Nishanth Chandran, Bhavana Kanukurthi, Sai Lakshmi Bhavana Obbattu, and **Sruthi Sekar**: Adaptive Extractors and their Application to Leakage Resilient Secret Sharing.
In: Advances in Cryptology – CRYPTO 2021.
- [C3] Divesh Aggarwal, Bhavana Kanukurthi, Sai Lakshmi Bhavana Obbattu, Maciej Obremski, and **Sruthi Sekar**: Rate One-Third Non-malleable Codes.
In: Symposium on Theory of Computing – STOC 2022.
- [C4] Nishanth Chandran, Bhavana Kanukurthi, Sai Lakshmi Bhavana Obbattu, and **Sruthi Sekar**: Leakage Resilient Secret Sharing with Optimal Share Size.
In: Advances in Cryptology – CRYPTO 2022.

Contents

Acknowledgements	i
Abstract	iii
Publications based on this Thesis	v
Contents	vi
List of Figures	ix
List of Tables	x
1 Introduction	1
1.1 Non-malleable Codes	3
1.1.1 Our Results	3
1.1.2 Related Works	4
1.2 Leakage Resilient Secret Sharing	6
1.2.1 Our Results	7
1.2.2 Related Works	7
1.3 Connection Between LRSS and NMCs	9
1.4 Organization of the Thesis	10
2 Preliminaries	11
2.1 Notations	11
2.2 Statistical distance and Entropy	12
2.3 Randomness Extractors	14
2.4 One-Time Message Authentication Codes	17
2.5 Non-malleable Codes	18

2.6	Secret Sharing Schemes	20
2.6.1	Adaptive Privacy	21
2.6.2	Consistent Re-sampling	25
2.6.3	Local Uniformity	27
2.6.4	Leakage Resilient Secret Sharing	27
I	Non-malleable Codes	28
3	Near-Optimal Non-malleable Codes	29
3.1	Introduction	29
3.1.1	Technical Overview	30
3.1.1.1	Case 1: Codeword is in $\mathcal{P}_2$ i.e., $\text{NMDec}(\underline{L}, \underline{R}) = \text{NMDec}(L, R)$. .	34
3.1.1.2	Case 2: Codeword is in $\mathcal{P}_3$ i.e., $\text{NMDec}(\underline{L}, \underline{R}) \neq \text{NMDec}(L, R)$. .	35
3.1.2	Organization of the Chapter	39
3.2	Our Construction	39
3.2.1	Building Blocks	39
3.2.2	Construction Overview	39
3.2.3	Security Proof	41
3.3	t -Source Strong Non-malleable Extractors to t -State Augmented NMCs	50
II	Leakage Resilient Secret Sharing Schemes	54
4	Near Optimal/Optimal Leakage Resilient Secret Sharing Schemes	55
4.1	Introduction	55
4.1.1	Technical Overview	57
4.1.2	Organization of the Chapter	60
4.2	Adaptive Extractors	61
4.2.1	Construction	61
4.3	Leakage Model: Joint and Adaptive	66
4.3.1	Adaptive Leakage and Reveal Model $\mathcal{F}_{leak}^{\psi, \mu}$	66
4.3.2	Joint Leakage and Reveal Model $\mathcal{J}^{X, \psi, \mu}$	67
4.4	Our Construction: Leakage Resilient Secret Sharing	68
4.4.1	Building Blocks and Construction	69
4.4.2	Security Proof: Adaptive Leakage and Reveal Model	70
4.4.2.1	Instantiation and Rate Analysis	81

CONTENTS

4.4.3	Security Proof: Joint Leakage and Reveal Model	81
4.4.3.1	Instantiation and Rate Analysis	83
4.4.4	LRSS for General Access Structures	84
4.5	Rate-optimal Non-adaptive LRSS Scheme	84
Conclusion		87
Bibliography		88

List of Figures

2.1	Adaptive Privacy of Secret Sharing	22
2.2	Shamir Secret Sharing Scheme	23
2.3	Benaloh-Leichter Secret Sharing Scheme	24
3.1	Construction of our Non-malleable Randomness Encoder	31
3.2	Construction of our Non-malleable Code	32
3.3	Simple non-malleable code with a great rate. Here h is an affine evasive function. The decoding procedure is analogous: the decoder inverts Steps 3 and 2, obtains keys k_w, k_c , verifies MACs from the Step 5 and proceeds to obtain the message via the Step 4. If in Step 2 the function h outputs $\perp$, then the decoder aborts and outputs $\perp$	38
3.4	Our NMC Construction	40
4.1	The Main Construction	60
4.2	The AdLeak^b Game	64
4.3	Adaptive Leakage and Reveal Model: $\text{Leak}_{\text{Share}}^m$ Distribution	67
4.4	Joint Leakage and Reveal Model: $\text{JLeak}_{\text{Share}}^m$ Distribution	68
4.5	Construction of our LRSS	70
4.6	Hybrid $\text{LeakA}_{\mathbf{q}}^m$	73
4.7	Hybrid $\text{LeakB}_{\mathbf{q}}^m$	74
4.8	Hybrid LeakC^m	75
4.9	Construction of our Optimal Rate LRSS	86

List of Tables

1.1	Prior Work on 2-state NMCs (n is codeword length)	5
1.2	LRSS Prior Work	9

Chapter 1

Introduction

Traditionally, cryptographic security of any system assumes that the adversary only has “black-box” access to an attacked cryptosystem, and can only observe its input-output behaviour, and nothing more. However, such security models fail to capture attacks on the actual physical implementation of the system, which are broadly called “*side-channel attacks*”. There are two very well studied category of such attacks: *leakage attacks*, which are passive in nature and allow the adversary to learn some information about the secret state of a cryptosystem, by means of some physical implementation properties (e.g. power consumption, heat, radiation, etc.), and *tampering attacks*, which are active in nature and allow the adversary to modify the secret state, and observe additional input-output behaviour on the tampered secret state, by means of some physical tampering of the hardware (e.g. heating to introduce random errors, cutting wires in circuit, etc.). For example, the security of RSA signature is shown to fail under random faults introduced into the computations by the work of Boneh, DeMillo and Lipton [25]. This has led to a flurry of research in leakage-resilient and tamper-resilient cryptography [78, 21, 22, 92, 23, 31, 85, 55, 13, 51, 57, 83, 84, 64, 73, 11]. Particularly, two fundamental objects of interest in this study have been “*Non-malleable Codes*” (NMCs) and “*Leakage Resilient Secret Sharing*” (LRSS) schemes.

Non-malleable Codes. Motivated by the need to protect data against tampering attacks, Dziembowski, Peitzak and Wichs, in their pioneering work [57], introduced non-malleable codes. Non-malleability is a widely studied notion in cryptography and strives to defend against *related* tampering. NMCs, in specific, are coding schemes where the adversary cannot tamper the codeword into the codeword of a related message. Informally, they provide the following guarantee: given a codeword C of a message m and a function f chosen from a tampering family $\mathcal{F}$, decoding $f(C)$ gives something that is either equal to m or completely independent

of m . This becomes extremely relevant when NMCs are used to store the secret key (which is tampered). NMCs assure us that the encoding of the secret key may be tampered with, but the tampered codeword decodes to a message that is independent of the original secret key. A consequence of this is that, observing input-output behaviour on this tampered key is now rendered useless and the cryptosystem continues to remain secure.

Leakage Resilient Secret Sharing. Secret sharing schemes [95, 24] are a fundamental cryptographic primitive and have many applications, such as in multi-party computation [18, 39], threshold cryptographic systems [50, 63, 94], and leakage-resilient circuit compilers [72, 60, 93]. These are cryptographic primitives that allow a dealer to distribute a secret to N parties, such that only an authorized subset of parties can reconstruct the original secret and any unauthorized set of parties have no information about the underlying secret (*privacy*). For instance, in a t -out-of- N threshold secret sharing scheme, there are N parties, and any collection of t ($t \leq N$) or more parties would correspond to an authorized set, and any collection of less than t parties would be unauthorized. Note that an implicit assumption is that the unauthorized set of parties has no information about secrets of the remaining shares. A rich study on leakage attacks initiated by Kocher [78] tells us that this is an idealized assumption that may not hold in practice. Such leakage can be dangerous and completely break the security of the underlying primitive¹. To protect against such leakage attacks, Dziembowski and Pietrzak in [55] introduced the problem of leakage resilience in secret sharing schemes. Informally, an LRSS gives a guarantee that the adversary gets no information about the secret, given its shares from an unauthorized set, as well as leakage from the remaining honest shares.

For both NMCs and LRSS schemes, a very important parameter associated with them is the *rate* of the schemes, which captures the redundancy introduced by the coding/sharing procedure. Since, in either scenario, the codeword/shares are stored on some tamper-prone/leakage-prone system, it is of at most importance that the encoding/sharing process does not blow-up the actual message size. For example, in an application of LRSS to leakage-resilient MPC [96], the size of the share affects the overall communication complexity of the MPC protocol, or in an application of NMCs to non-malleable commitments (NMComs) [67], the codeword size directly affects the communication size of the NMCom. Formally, rate of an NMC is the ratio of the message length to the codeword length, while rate of an LRSS is the ratio of the message length to the length of the largest share.

The goal of our thesis is to study these objects in detail and improve the status quo on their rates. Particularly, we build NMCs and LRSS schemes with near-optimal/optimal rates.

¹For example, Guruswami and Wooters [69] show that Shamir’s secret sharing scheme is completely insecure when the adversary gets some $t - 1$ shares and just one-bit of leakage from other shares.

1.1 Non-malleable Codes

Non-malleable codes provide a guarantee that given a codeword C of a message m and a function f chosen from a tampering family $\mathcal{F}$, decoding $f(C)$ gives something that is either equal to m or completely independent of m . It was observed in [57] that it is impossible to build NMCs secure against unrestricted tampering i.e., when $\mathcal{F}$ corresponds to the family of all functions. (To see this, simply consider the function $f(c) = \text{Enc}(\text{Dec}(c) + 1)$.) NMCs are therefore built with respect to specific classes of functions. A well-studied class of tampering functions is the 2-split-state model where the codeword consists of two states L and R , and the adversary tampers with each of these states independently. This is a very natural model and indeed such NMCs have found many applications in securing against physical (leakage and tampering) attacks [57, 84], domain extension of encryption schemes [43, 44], non-malleable commitments [67], non-malleable secret sharing [65, 10, 14, 96] and privacy amplification [37].

As we discussed above, the most important parameter for a non-malleable code is its rate $= \frac{\kappa}{n}$ where κ denotes the message length and n the codeword length. Cheraghchi and Guruswami [41] showed that the optimal achievable rate for the 2-split-state family is $\frac{1}{2}$. The quest for this holy grail has inspired a long line of fascinating research, introduced new pseudo-random objects and intricate proof techniques. Most recently, Aggarwal and Obremski [3] constructed a constant rate 2-split-state NMC (with negligible error). While asymptotically significant, they achieve a constant rate of only around $1/1,500,000$. Thus, there is an embarrassingly large gap between the best achievable rate ($1/2$) and what we currently know ($1/1,500,000$)! *In this thesis, we build 2-split state non-malleable codes with a near-optimal rate $\frac{1}{3}$.*

As a stepping stone towards this quest, we introduce a primitive called “*Non-malleable Randomness Encoder*” (NMRE), which is similar to a 2-split-state NMC, and consists of two independently tamperable states L and R . Contrary to an NMC, where the encoder encodes arbitrary messages, an NMRE’s encoder outputs L and R such that they decode to a random string, and herein lies all the difference: while the problem of building high-rate NMCs has eluded researchers for over a decade, *we show how to build NMREs with rate $\frac{1}{2}$* . At the same time, we emphasize that obtaining a high-rate NMC (instead of an NMRE) is critical for many applications (such as non-malleable commitments).

1.1.1 Our Results

We obtain our construction of a rate- $\frac{1}{3}$ NMC via rate boosters which compile low rate NMCs into high rate ones. A precursor to our construction ideas lead to the construction of a rate- $\frac{1}{2}$ NMRE. We state the informal result below, the details of which are discussed in Chapter 3.

Informal Theorem 1 . *There exists an efficient, information-theoretically secure ε -non-malleable code in the 2-split-state model with rate $1/3$, and an error of $\varepsilon = 2^{-\Omega(\frac{\kappa}{\log^3 \kappa})}$, where κ is the size of the message. We introduce the notion of non-malleable randomness encoders, and a precursor to our construction ideas gives us an information-theoretically secure ε -non-malleable randomness encoder in the 2-split-state model with rate $1/2$, and an error of $\varepsilon = 2^{-\Omega(\frac{\kappa}{\log^{\rho+1} \kappa})}$ (for any $\rho > 0$), where κ is the size of the message.*

1.1.2 Related Works

We now sketch the landscape of the prior results on NMCs, and particularly summarize the results on 2-split-state NMCs in Table 1.1. In [57], in addition to introducing non-malleable codes, Dziembowski *et al.* also introduced a model of tampering called the t -split-state model, where the codeword consists of t independently tamperable states. They give the first NMC constructions in the n -split-state model (where n is the codeword length), with rate ≈ 0.19 , and the 2-split-state model (using random oracles). Dziembowski, Kazana and Obremski [58] provided the first construction of 2-split-state NMCs without any assumptions. Their construction enabled encoding of 1-bit messages and used two source extractors. The first NMC in the 2-split-state model for k -bit messages was given by Aggarwal, Dodis and Lovett [5], which used inner product extractors with tools from additive combinatorics. In [41], Cheraghchi and Guruswami brought focus to the rate ($= \frac{\text{message length}}{\text{codeword length}}$) of non-malleable codes. In particular, they showed that the optimal achievable rate for the t -split-state family is $1 - 1/t$. Note that in the split-state tampering model, having as few states is most desirable, with 2 states being the best achievable. By the above result, the best possible rate for the 2-split-state model is therefore $\frac{1}{2}$. A long series of works² has made partial progress towards achieving this holy grail. We now discuss some of these results. The work of Cheraghchi and Guruswami [40] gave the first optimal rate non-malleable code in the n -split-state model (where n is the codeword length). More importantly, this work introduced non-malleable two-source extractors and demonstrated that these special extractors can be used to generically build 2-split-state NMCs. This connection has led to several fascinating works [35, 36, 80, 81] striving to improve the rate and number of states of non-malleable codes. Most notably, Chattopadhyay and Zuckerman [35] built a 10-split-state NMC with constant rate, making this the first constant rate construction with sublinear number of states. They achieve their result by first building a non-malleable extractor with 10 sources and then using the connection due to [40]. Aggarwal, Dodis, Kazana and Obremski [6] introduced the concept of non-malleable reductions – which would later be used

²Other works have considered non-malleable codes in models other than the 2-split-state model or under computational assumptions [9, 61, 12, 74, 8, 54, 46, 47, 33, 6, 32, 34, 68, 16, 62, 40, 35, 6, 36, 80, 81, 76, 77, 68, 3].

Work	Rate
[57]	1/6 (Existential, Random Oracle Model)
[41]	1/2 (Existential, Lower bound)
[58]	$\Omega(1/n)$ (Only for 1-bit messages)
[5, 1, 2]	$\Omega(1/n^{4/5})$
[6]	$n^{-\Omega(1)}$
[36]	$n^{-\Omega(1)}$
[80]	$\Omega(1/\log(n))$
[81]	$\Omega(\log \log(n)/\log(n))$
[81]	$\Omega(1)$ (with constant error)
[3]	$\approx 1/1,500,000$
Our Result	1/3

Table 1.1: Prior Work on 2-state NMCs (n is codeword length)

to build constant rate NMCs [3]. The work of Kanukurthi, Obbattu and Sekar [76] used seeded extractors to build a compiler that transforms a low rate non-malleable code into one with high rate and, in particular, obtained a rate 1/3, 4-split-state non-malleable code. This was subsequently improved to three states in the works of Kanukurthi, Obbattu and Sekar [77] as well as Gupta, Maji and Wang [68]. Li [81] obtained 2-split-state NMC with rate $\mathcal{O}(\frac{\log \log \log(1/\epsilon)}{\log \log(1/\epsilon)})$ (where ϵ is the error). Particularly, this gave a rate of $\mathcal{O}(\log \log(n)/\log(n))$, for negligible error $\epsilon = 2^{-\Omega(n)}$, and a constant rate for constant error, making this the first constant rate scheme in the 2-split-state model. The most recent work of Aggarwal and Obremski [3] relied on the concept of non-malleable reductions and built the first constant rate 2-split-state NMC with negligible error.

Other Related Works. In the computational setting, the work of [9] shows how to build an optimal rate-1, 2-split-state NMC and the work of [45] build the same for a stronger tampering model, called “continuous tampering”.

Non-malleable codes have also been studied extensively for various other tampering models like the lookahead tampering model [6, 32, 16, 68], NC0 circuits [15], AC0 circuits [34, 16], all polynomial-size tampering with bounded polynomial depth [48], affine functions [34], space-bounded tampering [62], continuous tampering models [61, 74, 8, 54, 45] and tampering functions that permute and perturb the codeword bits [12].

1.2 Leakage Resilient Secret Sharing

Leakage resilient secret sharing schemes give us a guarantee that the adversary gets no information about the secret, given its shares from an unauthorized set, as well as leakage from the remaining honest shares. This problem has received much attention (for example, [49, 84, 6, 65, 20, 14, 96, 79, 10, 59, 27, 38, 28]), wherein researchers have strived to improve various parameters such as its rate (defined as (message length)/(length of longest share)), leakage model (determining the leakage queries allowed to be made by the adversary) as well as leakage rate (defined as (number of bits of total leakage allowed)/(the size of a share)). Leakage model is an important parameter which determines the strength of the leakage adversary. There have been two broad categories of leakage models: *adaptive* vs *non-adaptive*, and *independent* vs *joint*. A non-adaptive and independent leakage model, called as the *local leakage model* [20, 65, 96, 10] allows the adversary to make a single non-adaptive query to obtain a complete unauthorized set of shares, along with independent (bounded) leakage on the remaining shares. LRSS schemes tolerating local leakage have been shown to have applications to leakage-resilient MPC [26, 66, 20, 96], leakage-resilient non-malleable secret sharing [65, 96, 14], and more recently to zero knowledge PCPs [71]. On the other end, a much stronger adaptive and joint leakage model, typically termed as *bounded collusion protocols* (BCP) model [79, 38] allows the adversary to make multiple adaptive (bounded) leakage queries on all shares, with each query depending jointly on multiple shares (joint) and also on the prior query responses (adaptive). In the information-theoretic setting, a majority of the works focus on improving the leakage model to get adaptive and joint leakages, and such strong leakage models come at the expense of poor and sub-optimal rate (typically $\omega(1)$). Particularly in an adaptive joint leakage model, the best known rate is achieved by the construction of [38], which is $\mathcal{O}(1/N)$, where N is the number of parties. On the other hand, there are works which focus on improving the rate of the LRSS scheme, and typically this comes at the expense of the leakage model. Particularly, a near-optimal rate of $1/3$ is achieved by the work of [96], but only in the non-adaptive and independent model.

While on the one hand, we do want to secure against as strong adversaries as possible, we also want the rate to be great, as it determines the efficiency of the application that relies on it (for example, in an application to leakage-resilient MPC, the size of the share affects the overall communication complexity of the MPC protocol). Furthermore, for standard threshold secret sharing schemes, we know constructions [95, 82] with optimal share size (i.e., same as the message size), but the picture is quite different in the presence of leakage. Thus, the main goal that we strive to achieve is to bridge this gap between the rate and leakage model. *In*

this thesis, we build LRSS schemes in a joint and adaptive model, achieving a rate of $\mathcal{O}(1)$. Furthermore, we also improve the status quo on the rate of non-adaptive independent LRSS scheme to build a scheme with an optimal rate³ of 1.

1.2.1 Our Results

For N parties and considering the t -threshold setting for secret sharing, we get the following results, the details of which are discussed in Chapter 4. Our main result is an LRSS scheme in the joint and adaptive leakage model, achieving a constant rate.

Informal Theorem 2 (Main Result). *We build an LRSS scheme, tolerating ψ adaptive queries, each dependent on X shares (with $\psi \cdot X \leq N - t + 1$) and the reveal of the remaining $t - 1$ shares, such that it achieves a rate of $(X^{\Theta(\psi X/t)})^{-1}$, while allowing $\Theta(\ell)$ bits of leakage per query, for threshold access structures. In particular, for a constant X and $N = \Theta(t)$, this gives the first constant-rate adaptive LRSS scheme for the threshold access structure.*

Furthermore, in the non-adaptive and independent leakage model, we improve upon the near-optimal construction of [96] to get the following optimal rate LRSS for general monotone access structures⁴ and N parties.

Informal Theorem 3. *Given any secret sharing scheme for general monotone access structure $\mathcal{A}$ with share size ℓ/R , where ℓ is the message length and $R \leq 1$, one can construct an LRSS for the same access structure $\mathcal{A}$, against the local leakage model allowing μ bits of leakage per share, with a share size of $\ell/R + \mu + o(\ell/R + \mu)$.*

1.2.2 Related Works

Firstly, we look at the relevant works in the local leakage model (non-adaptive and independent). Here, in a restricted setting (where the underlying field is a large characteristic field, the number of parties N is large, the threshold t is at least $N - o(\log N)$, and the adversary can only obtain a constant number of full shares), Benhamouda *et al.* [20] show that the Shamir secret sharing [95] is leakage resilient with share size $\geq (\text{message length} + 4 \cdot \mu)$, where μ is the number of bits of leakage from each share⁵. For the more general case of arbitrary N and t , Srinivasan and Vasudevan [96] constructed an LRSS against the local leakage model with a share size of

³This matches the lower bound for the rate of an information-theoretic LRSS in the threshold setting, against the local leakage model, given in [89].

⁴A monotone access structure $\mathcal{A}$ consists of subsets of parties such that, for any two subsets A, B with $A \subset B \subseteq [N]$ and $A \in \mathcal{A}$, it holds that $B \in \mathcal{A}$.

⁵In [20], under the same restrictions (on N , the field and the number of full shares allowed), the authors also consider the setting with threshold $t \leq \alpha N$, for $\alpha < 1$ for the Shamir secret sharing scheme, but this only allows constant bits of leakage per share.

approximately $(3 \cdot \text{message length} + \mu)$, to tolerate μ bits of leakage from each share, which is the best known prior result in this model.

To summarize the works in the adaptive and joint leakage model, we first list a few parameters that are relevant for LRSS:

- *Rate*: This is defined as $\frac{\text{messagelength}}{\text{sharelength}}$.
- *Global Limit*: This refers to the total number of shares on which the leakage queries can depend on.
- *Per-query Limit*: This refers to the number of shares that a specific query can depend on.
- *Per-query Leakage Rate*: This is the ratio of the total allowable leakage from a single leakage query to the size of a share.

In the information theoretic setting, the only prior LRSS schemes allowing for a joint and adaptive leakage model are [79, 38]. While our model allows adaptive queries on up to $N - t + 1$ shares, each dependent on at most X shares (where X is some value between 1 and $t - 1$), before fully revealing the remaining $t - 1$ shares, [38] allows adaptive queries on all N shares, each dependent on at most $t - 1$ shares before revealing $t - 1$ full shares. Both the schemes require the adaptive queries to be on disjoint sets of shares. However, our scheme/analysis offers a more fine-grained trade-off between the various parameters and allows us to obtain better results for certain settings. In particular, when we consider the instance where X is constant (and $t = \alpha N$, for a constant $\alpha < 1$), we get a constant-rate adaptive LRSS achieving a constant leakage rate, while [38] gets a rate and leakage rate of $\mathcal{O}(1/N)$ each, in all instances. To put this in context, even if [38] makes independent adaptive leakage queries on all shares, their rate is $\mathcal{O}(1/N)$ and the maximum number of bits they can leak is at most a constant fraction of the size of a single share, while we can leak close $(N - t + 1)$ times a constant fraction of the size of a single share!

The work of [38] also consider a variant of joint leakage, allowing overlap of the query sets, the detailed parameters of which are given in Table 1.2. We give a detailed comparison of the parameters achieved by the various schemes in Table 1.2, for the threshold setting with $t = \alpha N$ (for a constant $\alpha < 1$).

Other Related Works. For the case where the adversary is restricted to be computationally bounded, the works of [28, 59, 30] show LRSS schemes achieving optimal rate in the strongest possible adaptive and joint leakage model.

In this thesis, our focus is only on the so-called bounded leakage model [56], where we assume that the leakage is bounded by some length ℓ . However, a more general notion of “noisy leakage”

Work*	Rate	Joint Leakage	Global Limit	Per Query Limit	Leakage Rate (per query)	Adaptive	Full Shares
[96]	1/3	No	N	1	≈ 1	No	Unauth.
[10]	$\mathcal{O}(1/N)$	No	N	1	$\approx 1 - c$	No	Unauth.
[79]	$\mathcal{O}(1/\text{poly}(N))$	Yes (Overlapping)	N	$\log N$	$\approx \Theta(1/\text{poly}(N))$	Yes	$\log N$
[38]	$\mathcal{O}(1/N)$	Yes	N	$t - 1$	$\approx \Theta(1/N)$	Yes	Unauth.
[38] (threshold)	$\mathcal{O}(1/\text{poly}(N))$	Yes (Overlapping)	N	$\mathcal{O}(t/\log t)$	$\approx \Theta(1/\text{poly}(N))$	Yes	Unauth.
[38] (N -out-of- N)	$\mathcal{O}(1/\text{poly}(\ell))$	Yes (Overlapping)	$0.99N$	1	$\approx \Theta(1/\text{poly}(\ell))$	Yes	Unauth.
Our Result 1	$\Theta(1)$	Yes	$N - t + 1$	constant	$\approx \Theta(1)$	Yes	Unauth.**
Our Result 2	1	No	N	1	≈ 1	No	Unauth.

Table 1.2: LRSS Prior Work

- *All works mentioned here are information-theoretic. We write all comparisons for the threshold setting with threshold $t = \alpha N$ (where $\alpha < 1$ is a constant and N denotes the total number of parties).
- ** For our result, the unauthorized queries cannot overlap with the leakage queries.
- c is a small constant and ℓ is the message length.
- All schemes (except the joint overlapping schemes of [38] (threshold and N -out-of- N) actually work for general access structures.
- **Full Shares:** Number of complete shares that an adversary can see (at the end of all leakage queries, in the adaptive schemes). “Unauth.” refers to a complete set of unauthorized shares. Note that all schemes except [79] imply standard privacy property of secret sharing.

has also been considered in literature [87, 88], where the length of the leakage is not bounded, but the only restriction is that the conditional average min-entropy of the shares given the leakage should remain high enough. The recent work of [29] shows that the bounded leakage model can be lifted in a black-box way to a more general noisy leakage model, and further, as mentioned by this paper, this provides a way of converting the bounded leakage protocols discussed above (including ours) into a noisy leakage one.

1.3 Connection Between LRSS and NMCs

While this thesis treats LRSS and NMCs are separate primitives, with the goals highlighted in previous sections, they can be considered as special cases of a stronger more general primitive, called “*leakage resilient non-malleable secret sharing*” (LRNMSS) [79, 59, 27, 28]. Informally, an LRNMSS is an LRSS which additionally ensures that even if the shares are tampered with (each share being tampered arbitrarily and independent of each other), post getting the leakage, the recovered tampered secret looks independent of the original secret that was shared. As pointed in [58, 65], any t -split-state NMC is in fact a t -out-of- t non-malleable secret sharing scheme, where each state of the NMC can be thought of as a share of the message. Furthermore, a 2-split-state NMC that additionally allows leakage resilience (adversary gets independent leakage

from each state before tampering) [84, 7] is a 2-out-of-2 LRNMSS. However, the stronger general LRNMSS schemes in the literature have a poor share size and leakage rate, and hence, our goal is to consider the specialized cases of LRSS and NMCs and optimize their parameters (share size, leakage rate and codeword size).

Adding leakage resilience to our 2-split-state NMC. We wish to note here that NMCs that additionally allow leakage from each of the state would indeed be a stronger primitive to build and, more useful in the real-world scenario. While our NMCs directly do not offer this stronger feature, the work of [7] can be used to convert our NMC construction into one with the added feature of leakage resilience, while still preserving the rate of $1/3$.

Adding non-malleability to our LRSS. Our LRSS scheme is indeed prone to malleability attacks. However, by instantiating the non-malleable secret sharing compiler of [65] with our 2-split-state NMC and our LRSS scheme, we can get an LRNMSS scheme with only a small constant factor blow-up in the share size (i.e., in our joint and adaptive leakage model, this LRNMSS would be of rate $\mathcal{O}(1)$).

In the light of the above discussions, in Chapters 3 and 4, our focus is only on building LRSS and NMC schemes, and not on directly building the stronger LRNMSS schemes.

1.4 Organization of the Thesis

We begin by giving all the necessary preliminary definitions and lemmata in Chapter 2. Further, we divide our thesis into two parts:

Part I will focus on non-malleable codes. In Chapter 3, we will begin by giving a detailed technical overview of our construction ideas, which involves the precursor step of getting non-malleable randomness encoders. Followed by this, we will give our main construction and security proof for the near-optimal 2-split-state NMC.

Part II will focus on leakage resilient secret sharing schemes. In Chapter 4, we will begin by giving a detailed technical overview of our construction ideas. Then, we will describe our main result on building LRSS schemes with constant rate in the joint and adaptive leakage model. Finally, we will sketch the ideas behind the optimal rate LRSS scheme in the non-adaptive and independent leakage model.

Chapter 2

Preliminaries

In this chapter, we give the notations and definitions of various primitives used in our constructions.

2.1 Notations

We denote the security parameter by κ . We use $s \in_R S$ to denote that s is sampled uniformly from the set S , and $x \leftarrow X$ to denote that x is sampled from the probability distribution X . The concatenation of two binary strings x and y is denoted by $x\|y$. We denote the length of a binary string x by $|x|$, and the cardinality of any set S by $|S|$. For any set S , U_S denotes the uniform distribution on S , and we use the shorthand U_ℓ to represent $U_{\{0,1\}^\ell}$, for any integer $\ell > 0$. All logarithms are over base 2. The set $X \setminus Y \stackrel{\text{def}}{=} \{x : x \in X, x \notin Y\}$, is the set of elements in X that are not in Y . For a subset S of $[n]$:

- If $x_1, \dots, x_n$ are some variables or elements, then x_S denotes the set $\{x_i \text{ such that } i \in S\}$.
- For some function f outputting n values $y_1, \dots, y_n$ on input x , $f(x)_S$ denotes $(y_i)_{i \in S}$.
- If $T_1, \dots, T_n$ are sets, then T_S denotes the union $\cup_{i \in S} T_i$.

$\chi(a = b)$ indicates equality of the strings a and b , i.e., $\chi(a = b) = 1$ if and only if a is equal to b . In constructions, where we use field operations, we assume natural one-to-one correspondence between the set $\{0, 1\}^n$ and the field $GF(2^n)$.

For any event E , and any random variable X_1 , $X_1|_E$ denotes the distribution of the random variable X_1 conditioned on the event E . For any random variables X_1, X_2 and an event E , $(X_2, X_1|_{E, X_2})$ denotes the distribution where we sample X_2 according to its marginal distribution, and then sample X_1 conditioned on the choice of X_2 and the event E .

2.2 Statistical distance and Entropy

Let X_1, X_2 be two probability distributions over some set S . Their statistical distance is

$$\Delta(X_1; X_2) \stackrel{\text{def}}{=} \max_{T \subseteq S} \{\Pr[X_1 \in T] - \Pr[X_2 \in T]\} = \frac{1}{2} \sum_{s \in S} \left| \Pr_{X_1}[s] - \Pr_{X_2}[s] \right|$$

(they are said to be ε -close if $\Delta(X_1; X_2) \leq \varepsilon$ and denoted by $X_1 \approx_\varepsilon X_2$). We shorthand $\Delta(X_1, Y; X_2, Y)$ by $\Delta(X_1; X_2 | Y)$ for any random variables X_1, X_2, Y . The *min-entropy* of a random variable W is $\mathbf{H}_\infty(W) = -\log(\max_w \Pr[W = w])$. For a joint distribution (W, E) , the (average) conditional min-entropy of W given E is defined in [52] as

$$\tilde{\mathbf{H}}_\infty(W | E) = -\log(\mathbf{E}_{e \leftarrow E}(\max_w \Pr[W = w | E = e]))$$

(here the expectation is taken over e for which $\Pr[E = e]$ is nonzero). For a random variable W over $\{0, 1\}^n$, $W|E$ is said to be an (n, t) -source if $\tilde{\mathbf{H}}_\infty(W|E) \geq t$.

We now state some fundamental lemmata associated with min-entropy and statistical distance, which would be used in our proofs. Note that the lemmata stated without references are obvious properties. The following lemma states how the conditional min-entropy is related to the min-entropy of the joint distribution.

Lemma 2.1 [52] *If B has at most 2^λ possible values, then $\tilde{\mathbf{H}}_\infty(A | B) \geq \mathbf{H}_\infty(A, B) - \lambda \geq \mathbf{H}_\infty(A) - \lambda$, and more generally, $\tilde{\mathbf{H}}_\infty(A | B, C) \geq \tilde{\mathbf{H}}_\infty(A, B | C) - \lambda \geq \tilde{\mathbf{H}}_\infty(A | C) - \lambda$.*

The following lemma states a fundamental property of statistical distance, that, if two random variables are statistically close, then any function evaluation on them are also close.

Lemma 2.2 *For any random variables A, B , if $A \approx_\varepsilon B$, then for any (possibly randomized) function f , $f(A) \approx_\varepsilon f(B)$.*

Further, the following standard property of statistical distance upper bounds the distance between a pair of joint random variables by the conditional distributions.

Lemma 2.3 *For any random variables A, B over $\mathcal{A}$, and events E, E' with non-zero probabilities,*

$$\Delta(A \wedge E ; B \wedge E') \leq |\Pr[E] - \Pr[E']| + \Pr[E'] \cdot \Delta(A|E ; B|E')$$

where,

$$\Delta(A \wedge E ; B \wedge E') \stackrel{\text{def}}{=} \frac{1}{2} \sum_{a \in \mathcal{A}} |\Pr[A = a \wedge E] - \Pr[B = a \wedge E']|$$

and

$$\Delta(A|E ; B|E') \stackrel{\text{def}}{=} \frac{1}{2} \sum_{a \in \mathcal{A}} |\Pr[A = a|E] - \Pr[B = a|E']|$$

We require another property of the statistical distance, proved in [76], that if for two randomized functions F and G over a set $\mathcal{A}$, the function evaluations on every point of $\mathcal{A}$ are close, then for any two random variables A and B over $\mathcal{A}$ and $\mathcal{B}$, with B being independent of F and G , the function evaluations on the random variable A are close, even given A and B .

Lemma 2.4 [76] *Let A, B be correlated random variables over $\mathcal{A}, \mathcal{B}$. For randomized functions $F : \mathcal{A} \rightarrow \mathcal{X}$, $G : \mathcal{A} \rightarrow \mathcal{X}$ (randomness used is independent of B) if $\forall a \in \mathcal{A}, F(a) \approx_\epsilon G(a)$, then $(B, A, F(A)) \approx_\epsilon (B, A, G(A))$.*

The following lemma states that if a pair of random variables (X_1, X_2) and (Y_1, Y_2) are statistically close, then for any set $\mathcal{A}$, the conditional random variables X_2 conditioned on the event $X_1 \in \mathcal{A}$, i.e., $X_2|_{X_1 \in \mathcal{A}}$, and Y_2 conditioned on the event $Y_1 \in \mathcal{A}$, i.e., $Y_2|_{Y_1 \in \mathcal{A}}$ are also close.

Lemma 2.5 [5, Claim 4] *Let X_1, X_2, Y_1, Y_2 be random variables such that $(X_1, X_2) \approx_\epsilon (Y_1, Y_2)$. Then, for any non-empty set $\mathcal{A}$, we have:*

$$\Delta(X_2|_{X_1 \in \mathcal{A}}; Y_2|_{Y_1 \in \mathcal{A}}) \leq \frac{2\epsilon}{\Pr[X_1 \in \mathcal{A}]}.$$

The following lemma states that if for some random variable S on a set $\mathcal{S}$ and a function ϕ , $\phi(S)$ is close to some random variables on a partition of the set $\mathcal{S}$, then $\phi(S)$ will be close to a convex combination of these random variables on the whole set $\mathcal{S}$.

Lemma 2.6 [3] *Let S be some random variable distributed over a set $\mathcal{S}$, and let $\mathcal{S}_1, \dots, \mathcal{S}_j$ be a partition of $\mathcal{S}$. Let $\phi : \mathcal{S} \rightarrow \mathcal{T}$ be some function, and let $D_1, \dots, D_j$ be some random variables over the set $\mathcal{T}$. Assume that for all $1 \leq i \leq j$,*

$$\Delta(\phi(S)|_{S \in \mathcal{S}_i} ; D_i) \leq \epsilon_i.$$

Then

$$\Delta(\phi(S) ; D) \leq \sum \epsilon_i \Pr[S \in \mathcal{S}_i],$$

for some random variable $D \in \mathcal{T}$ such that for all d $\Pr[D = d] = \sum_i \Pr[S \in \mathcal{S}_i] \cdot \Pr[D_i = d]$.

The following lemma states another fundamental property of the statistical distance that the distance between two random variables is upper bounded by a convex combination of the distance between them, conditioned on a set of mutually exclusive and exhaustive events.

Lemma 2.7 *Let $A_1, \dots, A_n$ be mutually exclusive and exhaustive events. Then, for probability distributions X_1, X_2 over some set S , we have:*

$$\Delta(X_1 ; X_2) \leq \sum_{i=1}^n \Pr[A_i] \cdot \Delta(X_1|A_i ; X_2|A_i)$$

where $X_j|A_i$ is the distribution of X_j conditioned on the event A_i .

The following lemma states a simple property about the statistical distance between conditional distributions, given the distance between a pair of joint distributions.

Lemma 2.8 *For any random variables X, Y and Z and uniform random variable U , with X, Y, U defined over some binary space $\mathcal{X}$, such that Y is independent of X conditioned on Z and also independent of U conditioned on Z , if $(X, Z) \approx_\varepsilon (U, Z)$, then it implies that $(Y, Z)|(X \oplus Y = x) \approx_\varepsilon (Y, Z)|(U \oplus Y = x)$, for any $x \in \mathcal{X}$.*

Proof: Assume to the contrary that there exists an unbounded adversary, $\mathcal{D}$, who can distinguish between $(Y, Z)|(X \oplus Y = x)$ and $(U, Z)|(U \oplus Y = x)$ for some $x \in \mathcal{X}$, with advantage $> \varepsilon$. Then, we claim that we can build an adversary distinguishing (X, Z) and (U, Z) with the same advantage. The reduction is simple: given (a, c) from (X, Z) or (U, Z) , we sample $b \leftarrow Y|a \oplus Y = x$. Given the independence of Y from both $X|Z$ and $U|Z$, (b, c) is correctly simulated to be either $(Y, Z)|(X \oplus Y = x)$ (if (a, c) was from (X, Z)) or $(U, Z)|(U \oplus Y = x)$ (if (a, c) was from (U, Z)), for $\mathcal{D}$. Hence, the lemma is proved. $\square$

Finally, we require the following lemma about independence of two random variables given some sequential function evaluations on them, the proof of which can be found in [7].

Lemma 2.9 [55] *Let $A \in \mathcal{A}$ and $B \in \mathcal{B}$, and V_0 be random variables such that A is independent of B given any fixing of the random variable V_0 . Let $V_1, V_2, \dots$ be random variables defined as functions of A, B satisfying the following property. For all $i \in \mathbb{N}$, if i is even then $V_i = \phi_i(V_0, V_1, \dots, V_{i-1}, A)$ and if i is odd, then $V_i = \phi_i(V_0, V_1, \dots, V_{i-1}, B)$ for some function ϕ_i . Then for all i , A is independent of B given any fixing of $V_0, V_1, \dots, V_i$.*

2.3 Randomness Extractors

Randomness extractors (introduced by Nisan and Zuckerman [90]) output an almost uniform string from a (n, k) -entropic source, using a short uniform string, called *seed*, as a catalyst. Average-case extractors are extractors whose output remains close to uniform, even given the seed and some auxiliary information about the source (but independent of the seed), whenever

the source has enough average entropy given the auxiliary information. We formally define them below.

Definition 2.1 (Seeded Extractors) *We say that a polynomial time function $\text{Ext} : \{0, 1\}^n \times \{0, 1\}^d \rightarrow \{0, 1\}^\ell$, using d bits of randomness, is an $(n, k, d, \ell, \varepsilon)$ -strong seeded extractor if for all random variables $W \in \{0, 1\}^n$, E correlated with W , such that $\mathbf{H}_\infty(W|E) \geq k$, and X independent of W, E and uniform in $\{0, 1\}^d$, we have $(\text{Ext}(W; X), X, E) \approx_\varepsilon (U_\ell, X, E)$, where U is uniform in $\{0, 1\}^\ell$ and independent of X, E .*

Linear Seeded Extractors. *Further, Ext is said to be linear if for every $s \in \{0, 1\}^d$, $\text{Ext}(\cdot, s)$ is a linear function.*

We also define t -source non-malleable extractors, introduced by [40] for 2 sources, and extended to t sources in [80]. Informally, they guarantee that the extractor output $\text{nmExt}(X_1, \dots, X_t)$ on t independent entropic sources looks random, even given any one of the sources and the output on tampered sources, $\text{nmExt}(f_1(X_1), \dots, f_t(X_t))$.

Definition 2.2 [80] *A function $\text{nmExt} : (\{0, 1\}^n)^t \rightarrow \{0, 1\}^\ell$ is a (k, ε) -seedless strong non-malleable extractor for t independent sources w.r.t. family $\mathcal{F} = \{(f_1, \dots, f_t) : f_i : \{0, 1\}^n \rightarrow \{0, 1\}^n\}$, if it satisfies the following property: Let $X_1, \dots, X_t$ be t independent (n, k) -sources and $(f_1, \dots, f_t) \in \mathcal{F}$ be t arbitrary functions such that there exists an f_j with no fixed points, then for every i :*

$$(\text{nmExt}(X_1, \dots, X_t), \text{nmExt}(f_1(X_1), \dots, f_t(X_t)), X_i) \approx_\varepsilon (U_\ell, \text{nmExt}(f_1(X_1), \dots, f_t(X_t)), X_i)$$

Further, we lose the adjective “strong” if the source X_i is not given.

Instantiations. We require the following instantiation of a seeded extractor with short seed length, in our constructions.

Lemma 2.10 [70] *For every constant $\nu > 0$, all integers $n \geq k$ and all $\varepsilon \geq 0$, there is an explicit (efficient) $(n, k, d, \ell, \varepsilon)$ -strong seeded extractor with $\ell = k - \mathcal{O}\left(\log n + \log \frac{1}{\varepsilon}\right)$ and $d = \mathcal{O}\left(\log n + \log \frac{1}{\varepsilon}\right)$.*

Further, we instantiate linear extractors with extractors due to Raz et.al [91], which extracts almost all the randomness and is an improvement of Trevisan’s extractor [97]. Particularly, we use the following instantiation of the same given in [82].

Lemma 2.11 [82, Lemma 6] *There is an explicit $(n, k, d, \ell, \epsilon)$ -strong linear extractor with $d = \mathcal{O}(\log^3(\frac{n}{\epsilon}))$ and $\ell = k - \mathcal{O}(d)$.*

In our application of linear extractors, we will often require to uniformly sample an extractor source such that the extractor output on this source and a given seed s takes a given value y . Basically, given a seed s and some $y \in \{0, 1\}^\ell$, the inverting function needs to sample an element uniformly from the set $\text{Ext}(\cdot, s)^{-1}(y)$ (which is $\{w : \text{Ext}(w; s) = y\}$). We formalize this procedure⁶ as **InvExt** and show that linear extractors allow such sampling in the following lemma.

Lemma 2.12 *For every efficient linear extractor **Ext**, there exists an efficient randomized function **InvExt** : $\{0, 1\}^\ell \times \{0, 1\}^d \rightarrow \{0, 1\}^n \cup \{\perp\}$ (termed inverter) such that*

1. $U_n, U_d, \text{Ext}(U_n; U_d) \equiv \text{InvExt}(\text{Ext}(U_n; U_d), U_d), U_d, \text{Ext}(U_n; U_d)$
2. For each $(s, y) \in \{0, 1\}^d \times \{0, 1\}^\ell$,
 - (a) $\Pr[\text{InvExt}(y, s) = \perp] = 1$, if and only if there exists no $w \in \{0, 1\}^n$ such that $\text{Ext}(w; s) = y$.
 - (b) $\Pr[\text{Ext}(\text{InvExt}(y, s); s) = y] = 1$, if there exists some $w \in \{0, 1\}^n$ such that $\text{Ext}(w; s) = y$.

Proof: Recall that for a linear extractor, for any seed $s \in \{0, 1\}^d$, $\text{Ext}(\cdot, s)$ is a linear map from the vector space $\{0, 1\}^n$ to the vector space $\{0, 1\}^\ell$. Let $\mathcal{I}_s$ and $\mathcal{K}_s$ denote the image and kernel of this linear map $\text{Ext}(\cdot, s)$. We now define **InvExt** as follows. Fix any arbitrary input y, s to **InvExt**.

InvExt(y, s):

- If $y \in \mathcal{I}_s$
 - Let w be such that $\text{Ext}(w; s) = y$
 - Sample z uniformly from $\mathcal{K}_s$
 - Output $w + z$

Else output $\perp$

⁶In literature invertible (seeded) extractors (see [42] for an exposition on the same) are well-studied which allow efficient sampling of a source w and a seed s such that the extractor output on w and s equals a given value y . Note that our requirement to sample a source w given a seed s and a value y is stronger than the guarantee provided by invertible extractors. Hence, we explicitly show that certain extractors allow such sampling.

InvExt is efficient because the bases for the linear sub-spaces $\mathcal{K}_s$, $\mathcal{J}_s$ and the preimage space on the value y (corresponding to the linear map $\text{Ext}(\cdot, s)$) can be determined efficiently. By the definition, it is easy to see that InvExt satisfies property (2) of the Lemma statement. We now proceed to prove property (1) about statistical distance. Consider the set $\mathcal{S} = \{(w, s, y) : \text{Ext}(w; s) = y\}$.

For any $(w, s, y) \in \mathcal{S}$,

$$\begin{aligned} & \Pr[(\text{InvExt}(\text{Ext}(U_n; U_d), U_d), U_d, \text{Ext}(U_n; U_d)) = (w, s, y)] \\ &= \sum_{w' \in \{0,1\}^n} \Pr[U_n = w', U_d = s] \cdot \Pr[\text{InvExt}(y, s) = w] \cdot \chi(\text{Ext}(w'; s) = y) \end{aligned}$$

Since $\text{Ext}(w; s) = y$ by definition of $\mathcal{S}$, we know that w lies in the set of $|\mathcal{K}_s|$ elements from which $\text{InvExt}(y, s)$ chooses its output uniformly. Therefore $\Pr[\text{InvExt}(y, s) = w] = \frac{1}{|\mathcal{K}_s|}$. Further, since $\text{Ext}(\cdot; s)$ is a linear map and $y \in \mathcal{J}_s$, we know that there are exactly $|\mathcal{K}_s|$ number of values $w' \in \{0,1\}^n$ such that $\text{Ext}(w'; s) = y$. With these observations, we conclude that

$$\Pr[(\text{InvExt}(\text{Ext}(U_n; U_d), U_d), U_d, \text{Ext}(U_n; U_d)) = (w, s, y)] = \frac{1}{2^{n+d}}$$

Also for any $(w, s, y) \in \mathcal{S}$, $\Pr[(U_n, U_d, \text{Ext}(U_n; U_d)) = (w, s, y)] = \frac{1}{2^{n+d}}$.

For any $(w, s, y) \notin \mathcal{S}$, it holds that $\text{Ext}(w; s) \neq y$. With this we have

$$\Pr[(U_n, U_d, \text{Ext}(U_n; U_d)) = (w, s, y)] = 0$$

and

$$\Pr[(\text{InvExt}(\text{Ext}(U_n; U_d), U_d), U_d, \text{Ext}(U_n; U_d)) = (w, s, y)] = 0$$

The last equation is true because a) if $y \in \mathcal{J}_s$, then $\Pr[\text{InvExt}(y, s) = w] = 0$ b) if $y \notin \mathcal{J}_s$, then $\Pr[(U_d, \text{Ext}(U_n; U_d)) = (s, y)] = 0$. Further note that $\Pr[\text{InvExt}(\text{Ext}(U_n; U_d), U_d) = \perp] = 0$ as $\text{Ext}(U_n; U_d) \in \mathcal{J}_{U_d}$ with probability 1. Combining these observations, it follows that the statistical distance between the distributions $(\text{InvExt}(\text{Ext}(U_n; U_d), U_d), U_d, \text{Ext}(U_n; U_d))$ and $(U_n, U_d, \text{Ext}(U_n; U_d))$ is zero, which concludes the proof. $\square$

2.4 One-Time Message Authentication Codes

Information theoretic one-time message authentication codes give a guarantee that, given a message-tag pair, (m, t) , where t is generated using a tag generation algorithm, using a random authentication key, the probability with which an (unbounded) adversary can come up with a

valid message-tag pair (m', t') , for $m' \neq m$, is negligible.

Definition 2.3 A family of functions $\{\text{Tag}_{k_a} : \{0, 1\}^\gamma \rightarrow \{0, 1\}^\delta, \text{Vrfy}_{k_a} : \{0, 1\}^\gamma \times \{0, 1\}^\delta \rightarrow \{0, 1\}\}_{k_a \in \{0, 1\}^\tau}$ is said to be a μ -secure one time message authentication code if

1. For $k_a \in_R \{0, 1\}^\tau$, $\forall m \in \{0, 1\}^\gamma$, $\Pr[\text{Vrfy}_{k_a}(m, \text{Tag}_{k_a}(m)) = 1] = 1$,

$$\text{where for any } (m, t), \text{Vrfy}_{k_a}(m, t) \stackrel{\text{def}}{=} \begin{cases} 1 & \text{if } \text{Tag}_{k_a}(m) = t \\ 0 & \text{otherwise} \end{cases}$$

2. For any $m \neq m', t, t'$, $\Pr_{k_a}[\text{Tag}_{k_a}(m) = t | \text{Tag}_{k_a}(m') = t'] \leq \mu$, where $k_a \in_R \{0, 1\}^\tau$.

For simplicity, we consider Tag to be a deterministic function⁷.

Instantiation. The following lemma guarantees that there exists efficient one-time message authentication codes where the key and tag can be much shorter than the message to be authenticated. We refer the reader to [53] for a construction satisfying these parameters.

Lemma 2.13 [75] For any $\gamma, \varepsilon > 0$ there is an efficient ε -secure one time MAC with $\delta \leq (\log(\gamma) + \log(\frac{1}{\varepsilon}))$, $\tau \leq 2\delta$, where τ, γ, δ are key, message, tag lengths respectively.

2.5 Non-malleable Codes

Non-malleable codes (NMCs) were introduced in [57]. We begin by formally defining a coding scheme and its rate.

Definition 2.4 A (possibly randomised) function pair $(\text{Enc} : \mathcal{M} \rightarrow \mathcal{C}, \text{Dec} : \mathcal{C} \rightarrow \mathcal{M} \cup \{\perp\})$ is said to be a **coding scheme** from $\mathcal{M}$ to $\mathcal{C}$ if $\forall m \in \mathcal{M}$, $\Pr[\text{Dec}(\text{Enc}(m)) = m] = 1$ (the probability is over the randomness of Enc, Dec). The rate of the coding scheme is given by $\frac{\log |\mathcal{M}|}{\log |\mathcal{C}|}$.

Non-malleable codes are specific coding schemes, which give a guarantee that if an adversary tampers the non-malleable encoding, c , of a message m , then the recovered tampered message is either m or something independent of it. We formalize this in the following definition from [40].

Definition 2.5 A coding scheme (Enc, Dec) from $\mathcal{M}$ to $\mathcal{C}$, is said to be ε -**non-malleable** with respect to a tampering function family $\mathcal{F} \subseteq \{f : \mathcal{C} \rightarrow \mathcal{C}\}$ if $\forall f \in \mathcal{F}$, there exists a distribution Sim_f over $\mathcal{M} \cup \{\text{same}^*, \perp\}$ such that $\forall m \in \mathcal{M}$

$$\text{Tamper}_f^m \approx_\varepsilon \text{Copy}(m, \text{Sim}_f)$$

⁷We note here that since the Tag is deterministic, the Vrfy algorithm just executes the Tag tag again to check validity and hence it need not be considered as a separate algorithm. However, for notational convenience, to indicate the verification procedure in our constructions, we retain the Vrfy algorithm separately.

where Tamper_f^m denotes the distribution $\text{Dec}(f(\text{Enc}(m)))$ and for $\tilde{m} \leftarrow \text{Sim}_f$, $\text{Copy}(m, \text{Sim}_f)$ is defined as

$$\text{Copy}(m, \text{Sim}_f) = \begin{cases} m & \text{if } \tilde{m} = \text{same}^* \\ \tilde{m} & \text{otherwise} \end{cases}$$

Sim_f should be efficiently samplable given oracle access to $f(\cdot)$.

Split-state Tampering Family. Specifically, we consider NMCs with respect to the two split-state tampering family defined below, for message space $\{0,1\}^\alpha$ and codeword space $\{0,1\}^{\beta_1} \times \{0,1\}^{\beta_2}$, comprising two states:

$$\mathcal{F}_{\text{split}} \stackrel{\text{def}}{=} \{(f, g) : f : \{0,1\}^{\beta_1} \rightarrow \{0,1\}^{\beta_1}, g : \{0,1\}^{\beta_2} \rightarrow \{0,1\}^{\beta_2}\}$$

Our construction requires NMCs with respect to the split-state tampering family satisfying an additional property called “augmented security”, which was introduced in [9], and guarantees that in addition to the tampered message, one of the states, say L , of the codeword is also simulatable independent of the message.

Definition 2.6 (Augmented Non-malleable Codes) A coding scheme (Enc, Dec) from message space $\{0,1\}^\alpha$ to codeword space $\{0,1\}^{\beta_1} \times \{0,1\}^{\beta_2}$ is called an ε -augmented non-malleable code with respect to $\mathcal{F}_{\text{split}}$, if the following holds. For any (possibly randomized) $(f, g) \in \mathcal{F}_{\text{split}}$, let $\text{Tamper}_{f,g}^m$ denote the distribution $\text{Dec}(f(L), g(R))$, for $(L, R) \leftarrow \text{Enc}(m)$. There exists a distribution $\text{Sim}_{f,g}$ over $(\{0,1\}^\alpha \cup \{\text{same}^*, \perp\}) \times \{0,1\}^{\beta_1}$ that is given oracle access to $(f, g)(\cdot)$, such that, for all $m \in \{0,1\}^\alpha$

$$\text{Tamper}_{f,g}^m, L \approx_\varepsilon \text{Copy}(m, \text{Sim}_{f,g}),$$

where $(\tilde{m}, L^{\text{sim}}) \leftarrow \text{Sim}_{f,g}$, and $\text{Copy}(m, \text{Sim}_{f,g})$ is defined as

$$\text{Copy}(m, \text{Sim}_{f,g}) \stackrel{\text{def}}{=} \begin{cases} (m, L^{\text{sim}}) & \text{if } \tilde{m} = \text{same}^* \\ (\tilde{m}, L^{\text{sim}}) & \text{otherwise} \end{cases}$$

We use the following secret sharing property of non-malleable codes in the 2-split-state model $\mathcal{F}_{\text{split}}$, in our proofs, which states that for any two messages m_0 and m_1 , any one state of their codewords, say the right states, are statistically close.

Lemma 2.14 [6] Let $\text{Enc} : \{0,1\}^\alpha \rightarrow \{0,1\}^{\beta_1} \times \{0,1\}^{\beta_2}$ and $\text{Dec} : \{0,1\}^{\beta_1} \times \{0,1\}^{\beta_2} \rightarrow \{0,1\}^\alpha \cup \{\perp\}$ be an ε -non-malleable code in the 2-split-state model for some $\varepsilon < 1/2$. For

any pair of messages $m_0, m_1 \in \{0, 1\}^\alpha$, $R^{m_0} \approx_{2\varepsilon} R^{m_1}$, where $(L^{m_0}, R^{m_0}) \leftarrow \text{Enc}(m_0)$ and $(L^{m_1}, R^{m_1}) \leftarrow \text{Enc}(m_1)$.

Instantiations. In our constructions, we use the following two instantiations of NMCs in the 2-split-state model, one due to [5, 1, 2] and another due to [81], which are both stated below in Theorems 2.1 and 2.2, respectively. Further, the NMC in Theorem 2.1 is shown to be augmented secure in [9], which we incorporate in the theorem statement.

Theorem 2.1 [5, 1, 2] *There exists an efficient construction of an ε -augmented non-malleable code in the split state model with message space $\{0, 1\}^\kappa$ and codeword space $\mathbb{F}_p^q \times \mathbb{F}_p^q$, for a prime $p \leq 2^{O(\kappa)}$, $q = O(\kappa^4)$ and $\varepsilon = 2^{-\Omega(\kappa)}$.*

Theorem 2.2 [81, Theorem 1.15] *There are constants $0 < c_1, c_2 < 1$ such that for any $\beta \in \mathbb{N}$ and $2^{-\frac{c_2\beta}{\log \beta}} \leq \varepsilon \leq c_1$, there exists an explicit non-malleable code in the 2-split-state model with block length 2β , rate $\Omega(\frac{\log \log \log(1/\varepsilon)}{\log \log(1/\varepsilon)})$ and error ε .*

Our constructions require an augmented NMC, and while it is not shown in [81], that their code is augmented secure, we prove in Section 3.3 that, it is indeed, an augmented NMC.

2.6 Secret Sharing Schemes

Secret sharing schemes provide a mechanism to distribute a secret into shares such that only an authorized subset of shares can reconstruct the secret and any unauthorized subset of shares has “almost” no information about the secret. We now define secret sharing schemes formally.

Definition 2.7 *Let $\mathcal{M}$ be a finite set of secrets, where $|\mathcal{M}| \geq 2$. Let $[N]$ be a set of identities (indices) of N parties. A sharing function $\text{Share} : \mathcal{M} \rightarrow (\{0, 1\}^\ell)^N$ is a $(\mathcal{A}, N, \epsilon_s)$ - **secret sharing scheme** with respect to a monotone access structure⁸ $\mathcal{A}$ if the following two properties hold :*

1. **Correctness:** *The secret can be reconstructed by any set of parties that are part of the access structure $\mathcal{A}$. That is, for any set $T \in \mathcal{A}$, there exists a deterministic reconstruction function $\text{Rec} : (\{0, 1\}^\ell)^{|T|} \rightarrow \mathcal{M}$ such that for every $m \in \mathcal{M}$,*

$$\Pr[\text{Rec}(\text{Share}(m)_T) = m] = 1$$

where the probability is over the randomness of the Share function and if $(sh_1, \dots, sh_N) \leftarrow \text{Share}(m)$, then $\text{Share}(m)_T$ denotes $\{sh_i\}_{i \in T}$. We will slightly abuse the notation and

⁸ $\mathcal{A}$ is a monotone access structure if for all A, B such that $A \subset B \subseteq [N]$ and $A \in \mathcal{A}$, it holds that $B \in \mathcal{A}$. Throughout this paper whenever we consider a general access structure, we mean a monotone access structure.

denote Rec as the reconstruction procedure that takes in $T \in \mathcal{A}$ and $\text{Share}(m)_T$ as input and outputs the secret. In one of our constructions, we will consider an ϵ_c -correct secret sharing scheme, where the $\Pr[\text{Rec}(\text{Share}(m)_T) = m]$ is $\geq 1 - \epsilon_c$.

2. **Statistical Privacy:** Any collusion of parties not part of the access structure should have “almost” no information about the underlying secret. More formally, for any unauthorized set $U \notin \mathcal{A}$, and for every pair of secrets $m, m' \in \mathcal{M}$,

$$\Delta((\text{Share}(m))_U; (\text{Share}(m'))_U) \leq \epsilon_s$$

An access structure $\mathcal{A}$ is said to be an (N, t) -**threshold access structure** if and only if $\mathcal{A}$ contains all subsets of $[N]$ of size at least t .

Rate of a secret sharing scheme is defined as $\frac{\text{message size}}{\text{share size}}$ (which would be equal to $\frac{\log |\mathcal{M}|}{\ell}$).

We now define three properties of secret sharing schemes, which we use in our constructions. First is a stronger privacy requirement from the secret sharing, called *adaptive privacy* (introduced by Bellare and Rogaway [17]⁹), second is the property of *consistent resampling*, given some shares of a message, and third is the property of *local uniformity*, which requires each share to look uniform. We formalize all these properties below.

2.6.1 Adaptive Privacy

Statistical privacy captures privacy against any non-adaptively chosen unauthorized set U . *Adaptive privacy* preserves privacy even when the choice of U to be adaptive, which means the following. Let $U = \{i_1, \dots, i_q\}$. We say i_j is chosen adaptively, if its choice depended on $\{\text{share}_j\}_{j \in \{i_1, \dots, i_{j-1}\}}$. The choice of which share to query next depends on all the previously observed shares. We give the formal definition below.

We say a $(\mathcal{A}, N, \epsilon_s)$ -secret sharing scheme satisfies adaptive privacy with error ϵ_{adp} if, for any distinguisher $\mathcal{D}$, the advantage in the game in Figure 2.1 is at most ϵ_{adp} .

Instantiations. While generally, any secret sharing scheme may not be adaptively private, we show that for the threshold setting, the scheme of [95] and for the general access structures, the scheme of [19] are both adaptively private. We use them to instantiate our constructions.

Adaptive privacy of Shamir Secret Sharing. We begin by looking at the Shamir secret sharing scheme [95] for threshold access structures and show that it is, in fact, adaptively private. Consider the field $\mathbb{Z}_q$ for prime q such that $N < q$. The Shamir secret sharing scheme is described in Figure 2.2.

⁹In [17], the authors refer to adaptive privacy as privacy against dynamic adversaries.

Game_{Ad-Privacy} : For any arbitrary distinct messages $m_0, m_1 \in \mathcal{M}$

1. $(share_1, \dots, share_N) \leftarrow \text{Share}(m_b)$ where $b \in_R \{0, 1\}$
2. For $j = 1$ to q ^a
 - $\mathcal{D}$ queries on a distinct index i_j (such that $i_{[j]} \notin \mathcal{A}$) and receives $share_{i_j}$
3. $\mathcal{D}$ outputs the guess b' for b and wins if $b = b'$

^a q is arbitrary and chosen by $\mathcal{D}$. It need not be chosen a-priori. We only use it to denote the total number queries made by $\mathcal{D}$

Figure 2.1: Adaptive Privacy of Secret Sharing

Lemma 2.15 *The Shamir secret sharing scheme in Figure 2.2 is adaptively private against the (N, t) -threshold access structure.*

Proof: To prove the adaptively privacy, it is sufficient to prove the following claim.

Claim 1 *For any $U \subseteq [N]$ such that $|U| \leq t - 1$ (unauthorized set), the following two distributions are equivalent.*

$\mathcal{X} :$ • $a_0 = m, a_1, \dots, a_{t-1} \in_R \mathbb{Z}_q.$ • $p(x) = \sum_{i=0}^{t-1} a_i x^i.$ • Output $(p(i))_{i \in U}.$	$\mathcal{Y} :$ • $y_i \in_R \mathbb{Z}_q, \text{ for each } i \in U.$ • Output $(y_i)_{i \in U}.$
---	---

Proof: Let $U = \{i_1, \dots, i_{t-1}\}$, vandermonde matrix V be defined as: $V_{kj} = i_k^j$, for each $k, j \in [t - 1]$, A be the column matrix: $(a_1 \dots a_{t-1})^T$, Z be the column matrix $(z_{i_1} \dots z_{i_{t-1}})^T$ and A_0 be the $(t - 1) \times 1$ column matrix whose all entries are a_0 . For any $z_{i_1}, \dots, z_{i_{t-1}}$ from the field $\mathbb{Z}_q$,

$$\begin{aligned}
\Pr[\mathcal{X} = (z_i)_{i \in U}] &= \Pr[VA = Z - A_0] \\
&= \Pr[A = V^{-1}(Z - A_0)] \\
&= 1/q^{t-1} = \Pr[\mathcal{Y} = (z_i)_{i \in U}]
\end{aligned}$$

□

Given the above claim, clearly, each adaptive unauthorized query can be responded with a random field element. Hence, the scheme is adaptively private. □

ShamirShare $_N^t(m)$:

- Set $a_0 = m$ and pick $a_1, \dots, a_{t-1} \in_R \mathbb{Z}_q$.
- Define the polynomial $p(x) = a_0 + a_1x + \dots + a_{t-1}x^{t-1}$.
- Output: $(p(1), \dots, p(N))$.

ShamirRec $_N^t(\{y_i\}_{i \in T})$: (where $|T| = t$ is some reconstruction set. For simplicity in writing, say $T = \{1, \dots, t\}$)

- Use Lagrange interpolation to reconstruct the polynomial:
 - For $1 \leq i \leq t$: the Lagrange polynomial L_i is defined as the degree t polynomial for which $L_i(i) = 1$ and $L_i(j) = 0$, for each $j \neq i, 1 \leq j \leq t$.
 - $p(x) = \sum_{i=1}^t y_i L_i(x)$.
- Output $m = p(0)$.

Figure 2.2: Shamir Secret Sharing Scheme

Adaptive Privacy of Benaloh-Leichter Secret Sharing. We begin by describing the N -party Benaloh-Leichter secret sharing scheme [19] for a general monotone access structure, $\mathcal{A}$. Any monotone access structure $\mathcal{A}$ can be associated with an equivalent monotone formula $\mathcal{F}$, on n variables (the access structure $\mathcal{A}$ defined by F will be the set of subsets of parties, A , for which F is true when the variables indexed by A are set to true, and vice versa). Further, any monotone formula can be implemented using only *AND* and *OR* operators. We denote the variables of the monotone formulae by $\{v_i : i \in [N]\}$. We describe the sharing procedure, $\text{BLShare}(s, F)$, for secret $s \in \mathbb{Z}_q$ (where $\mathbb{Z}_q$ is a field of prime order q such that $n < q$) and a monotone formula F , in a recursive manner in Figure 2.3.

Lemma 2.16 *For every monotone formula F (with corresponding access structure $\mathcal{A}$), the above Benaloh-Leichter scheme for N parties in Figure 2.3 is adaptively private.*

Proof: To prove the adaptive privacy of the scheme, it is sufficient to prove the following claim.

Claim 2 *For each $A \notin \mathcal{A}$ (unauthorized set), the following distributions are equivalent.*

$\text{BLShare}(s, F) :$

- $\text{BLShare}(s, v_i)$ assigns the share s to party P_i .
- $\text{BLShare}(s, A \vee B) = \text{BLShare}(s, A) \cup \text{BLShare}(s, B)$, for every clause A and B in F .
- $\text{BLShare}(s, A \wedge B) = \text{BLShare}(s_1, A) \cup \text{BLShare}(s_2, B)$, where s_1 and s_2 are uniformly chosen from $\mathbb{Z}_q$ such that $s = (s_1 + s_2) \bmod q$, for every clause A and B in F .

In general for operators with more than 2 arguments and if THRESHOLD_k operators (which are true if and only if at least k of its clauses are true) are used, do the following:

- $\text{BLShare}(s, \vee(F_1, \dots, F_m)) = \cup_{1 \leq i \leq m} \text{BLShare}(s, F_i)$.
- $\text{BLShare}(s, \wedge(F_1, \dots, F_m)) = \cup_{1 \leq i \leq m} \text{BLShare}(s_i, F_i)$, where s_i 's are chosen uniformly from $\mathbb{Z}_q$ such that $s = \sum_{i=1}^m s_i \bmod q$.
- $\text{BLShare}(s, \text{THRESHOLD}_k(F_1, \dots, F_m)) = \cup_{1 \leq i \leq m} \text{BLShare}(s_i, F_i)$, where $(s_1, \dots, s_m) \leftarrow \text{ShamirShare}_m^k(s)$.

Figure 2.3: Benaloh-Leichter Secret Sharing Scheme

$\mathcal{X} :$ • $Sh_1, \dots, Sh_N \leftarrow \text{BLShare}(s, F)$. • <i>Output</i> $(Sh_i)_{i \in A}$.	$\mathcal{Y} :$ • $Sh_i \in_R \mathbb{Z}_q$, for each $i \in A$. • <i>Output</i> $(Sh_i)_{i \in A}$.
---	--

Proof: We prove the claim by induction on the number of operators in the formula F .

- *Base Case:* A formula with no operators consists of one variable v_i , for some $i \in [N]$ and the corresponding access structure will have all subsets of parties containing the i -th one. $\text{BLShare}(s, v_i)$ gives s to P_i alone. For each $A \notin \mathcal{A}$, $i \notin A$, and hence, parties in A do not get anything in this case.
- *Induction Hypothesis:* Suppose the claim holds true for F with $< d$ operators ($d > 0$).
- Now, let F be formula containing d operators, written as, $o(F_1, \dots, F_m)$, where o is one of $\vee$, $\wedge$ or THRESHOLD_k , and each of $F_1, \dots, F_m$ is a monotone formula with less than d operators. The following cases arise depending on which operator o is considered.
 - *Case 1:* For $o = \vee$, $\text{BLShare}(s, \vee(F_1, \dots, F_m)) = \cup_{1 \leq i \leq m} \text{BLShare}(s, F_i)$. By induction hypothesis, for each $A \notin \mathcal{A}$, for each $1 \leq i \leq m$, the shares assigned to parties in A by $\text{BLShare}(s, F_i)$ are identical to random field elements. Moreover, for $i \neq j$, $\text{BLShare}(s, F_i)$

and $\text{BLShare}(s, F_j)$ assign shares independently. Hence, for all parties in $\mathcal{A}$, the shares held by them are identical to uniform.

- *Case 2:* For $o = \wedge$, $\text{BLShare}(s, \wedge(F_1, \dots, F_m)) = \cup_{1 \leq i \leq m} \text{BLShare}(s_i, F_i)$, where $s_1, \dots, s_m$ are chosen at random from $\mathbb{Z}_q$ such that $s = \sum_{i=1}^m s_i \pmod q$. Now, for any $A \notin \mathcal{A}$, $\exists i$, such that, the shares of $\text{BLShare}(s_i, F_i)$ held by parties in A have no information about s_i . Further, by induction hypothesis, since the parties in A corresponding to F_i will be unauthorized, shares of $\text{BLShare}(s_i, F_i)$ corresponding to them will look uniform. Since the following two distributions are identical:

* $s_1, \dots, s_n \in_R \mathbb{Z}_q (s = \sum_{i=1}^m s_i \pmod q)$. * Output $(s_j)_{j \neq i}$.	* $s_j \in_R \mathbb{Z}_q$, for each $j \neq i$. * Output $(s_j)_{j \neq i}$.
--	---

therefore, the shares of parties in A corresponding to $\text{BLShare}(s_j, F_j)$, for each $j \neq i$ are also identical to uniform.

- For $o = \text{THRESHOLD}_k$, $\text{BLShare}(s, \text{THRESHOLD}_k(F_1, \dots, F_m)) = \cup_{1 \leq i \leq m} \text{BLShare}(s_i, F_i)$, where $(s_1, \dots, s_m) \leftarrow \text{ShamirShare}_m^k(s)$. For $A \notin \mathcal{A}$, parties in A can only get $< k$ of the s_i 's (else A will be in $\mathcal{A}$), say $s_{i_1}, \dots, s_{i_{k-1}}$ are known to parties in A . By the property of k -threshold Shamir secret sharing (shown in the proof of Lemma 2.15), $(s_{i_1}, \dots, s_{i_{k-1}}) \equiv U_{\mathbb{Z}_q^{k-1}}$. Hence, the shares obtained by parties in A corresponding to $\text{BLShare}(s_{i_j}, F_{i_j})$, for each $j \in [k-1]$, are identical to uniform. Further, for each $j \notin \{i_1, \dots, i_{k-1}\}$, since the parties in A are unauthorized w.r.t. $\text{BLShare}(s_j, F_j)$, by induction hypothesis, their shares are identical to uniform.

Hence, by induction, the claim is proved. $\square$

Given the above claim, clearly each adaptive unauthorized query can be responded with a random field element. Hence, the scheme is adaptively private. $\square$

2.6.2 Consistent Re-sampling

For any $(\mathcal{A}, N, \epsilon_s)$ -secret sharing scheme $(\text{Share}, \text{Rec})$, for any message m and a subset $\mathcal{L} \subseteq [N]$, when we say “ $(sh_1, \dots, sh_N) \leftarrow \text{Share}(m)$ consistent with $sh_{\mathcal{L}}^*$ on $\mathcal{L}$ ” or “ $(sh_1, \dots, sh_N) \leftarrow \text{Share}(m | sh_{\mathcal{L}}^*)$ ” we mean the following procedure:

- Sample and output $(sh_1, \dots, sh_N)$ uniformly from the distribution $\text{Share}(m)$ conditioned on the event that $sh_{\mathcal{L}} = sh_{\mathcal{L}}^*$
- If the above event is a zero probability event then output a string of all zeroes (of appropriate length).

We require the following consistent re-sampling feature¹⁰, which informally states that for any $(\mathcal{A}, N, \epsilon_s)$ -secret sharing scheme and any message m , the distribution of shares which are re-sampled as shares of m , conditioned on some set T of shares (which are also generated as shares of m) chosen adaptively, is identical to the distribution of shares of m generated directly.

Lemma 2.17 *For any $(\mathcal{A}, N, \epsilon_s)$ -secret sharing scheme $(\text{Share}, \text{Rec})$ and for any message m , the following two distributions are identical.*

$\mathcal{D}_1 :$ • $(sh'_1, \dots, sh'_N) \leftarrow \text{Share}(m)$ • $(sh_1, \dots, sh_N) \leftarrow \text{Share}(m sh'_T)$ • <i>Output</i> $(sh_1, \dots, sh_N)$	$\mathcal{D}_2 :$ • $(sh_1, \dots, sh_N) \leftarrow \text{Share}(m)$ • <i>Output</i> $(sh_1, \dots, sh_N)$
--	---

Here, $T \subseteq [N]$ can be any subset chosen as: every index (except the first) depends arbitrarily on the shares corresponding to all the previous indices.

Proof: Let message m be fixed. We prove the lemma inductively (on size of T). First, suppose $T = \{i_1\} \subseteq [N]$. Let $\text{Share}(m; \mathcal{R}) \equiv (\mathcal{H}_j^m)_{j \in [N]}$ denote the distribution on the shares of m , where $\mathcal{R}$ is the uniform distribution on space of randomness R . Consider for any $(sh_1, \dots, sh_N) \in \text{Share}(m; R)$ and for $(sh'_j)_{j \in [N]} \leftarrow \text{Share}(m; \mathcal{R})$:

$$\begin{aligned}
\Pr[\mathcal{D}_1 = (sh_1, \dots, sh_N)] &= \Pr[\mathcal{H}_j^m = sh_j \ \forall j \in [N] | sh_{i_1} = sh'_{i_1}] \\
&= \Pr[\mathcal{H}_{i_1}^m = sh'_{i_1}, \mathcal{H}_j^m = sh_j \ \forall j \neq i_1 \in [N]] \\
&= \Pr[\mathcal{D}_2 = (sh_1, \dots, sh_N)]
\end{aligned} \tag{2.1}$$

The last equality follows since $sh'_{i_1} = sh_{i_1}$.

Now, suppose that the distributions are equivalent for $|T| = k - 1$ (for any $2 \leq k \leq N - 1$), then we show that for $|T| = k$ as well, they are equivalent. Consider for any $(sh_1, \dots, sh_N) \in \text{Share}(m; R)$, for $(sh'_j)_{j \in [N]} \leftarrow \text{Share}(m; \mathcal{R})$ and for $T = \{i_1, \dots, i_k\} \subseteq [N]$, where, for each $j \in [k] \setminus \{1\}$, i_j is generated as an arbitrary function of $\{sh'_m : m \in \{i_1, \dots, i_{j-1}\}\}$:

$$\begin{aligned}
\Pr[\mathcal{D}_1 = (sh_1, \dots, sh_N)] &= \Pr[\mathcal{H}_j^m = sh_j \ \forall j \in [N] | sh_{i_j} = sh'_{i_j} \ \forall j \in [k]] \\
&= \Pr[\mathcal{H}_{i_1}^m = sh'_{i_1}, \mathcal{H}_j^m = sh_j \ \forall j \neq i_1 \in [N] | sh_{i_j} = sh'_{i_j} \ \forall j \in [k] \setminus \{1\}] \\
&= \Pr[\mathcal{H}_{i_1}^m = sh_{i_1}, \mathcal{H}_j^m = sh_j \ \forall j \neq i_1 \in [N] | sh_{i_j} = sh'_{i_j} \ \forall j \in [k] \setminus \{1\}]
\end{aligned} \tag{2.2}$$

¹⁰Note that we only use the re-sampling in proofs and do not require the procedure to be efficient.

which follows by equation 2.1 (the first case). Now, since, fixing sh'_{i_1} , $i_2 = f(sh'_{i_1})$ is a fixed index, and since the equivalence is true for $|T| = k - 1$ by the induction hypothesis, the above equation 2.2 is equal to:

$$\Pr[\mathcal{D}_2 = (sh_1, \dots, sh_N)]$$

Hence, we proved the lemma for $T \subseteq [N]$, for all $1 \leq |T| \leq N - 1$. $\square$

2.6.3 Local Uniformity

We say a secret sharing scheme $(\text{Share}, \text{Rec})$ satisfies *local uniformity* if the distribution of each individual share given out by the Share function is ϵ_u -statistically close to the uniform distribution in its share space. Formally, any sharing function $\text{Share} : \mathcal{M} \rightarrow \{\{0, 1\}^\ell\}^N$ is ϵ_u -locally uniform if for each message $m \in \mathcal{M}$ it holds that

$$\text{Share}(m)_{\{i\}} \approx_{\epsilon_u} U_\ell, \forall i \in [N]$$

Note that Shamir secret sharing scheme (Figure 2.2) is an instantiation of a locally uniform secret sharing scheme (for threshold access structures).

2.6.4 Leakage Resilient Secret Sharing

We now define a special form of secret sharing scheme, called leakage resilient secret sharing (LRSS), introduced in [55], which allow the adversary to not only get an unauthorized set of shares, but also additional bounded leakage from the remaining shares. The leakage allowed is captured by some leakage family/model. We formally define an LRSS with respect to a leakage family $\mathcal{F}$, below.

Definition 2.8 *An $(\mathcal{A}, N, \epsilon_s)$ -secret sharing scheme $(\text{Share}, \text{Rec})$, is said to be ϵ_{lr} -leakage resilient against a leakage family $\mathcal{F}$ if for all functions $f \in \mathcal{F}$, and for any two messages m, m' ,*

$$\Delta(f(\text{Share}(m)); f(\text{Share}(m'))) \leq \epsilon_{lr}.$$

We consider different leakage models $\mathcal{F}$ in our constructions, which will be formalized later.

Part I

Non-malleable Codes

Chapter 3

Near-Optimal Non-malleable Codes

We begin by diving into the first primitive of our interest, *non-malleable codes*, used to secure against tampering attacks. We will show how to build them with a near-optimal rate.

3.1 Introduction

Recall that non-malleable codes, introduced in [57], provide a guarantee that the encoding of a message, when tampered will either decode to the same message as the original, or to an independent message. One of the most studied tampering model for NMCs is the t -split-state model, which allows the adversary to tamper with t parts of the codeword independently and arbitrarily. The holy grail in the realm of NMCs has been to build optimal rate non-malleable codes in the 2-split-state tampering model (the strongest). As shown in [41], the optimal achievable rate for 2-split-state NMCs is $1/2$. This problem has eluded researchers for a decade now, and an impressive line of works [9, 61, 12, 74, 8, 54, 46, 47, 33, 6, 32, 34, 68, 16, 62, 40, 35, 6, 36, 80, 81, 76, 3] have made partial progress towards achieving this holy grail, as discussed in Section 1.1.2, with the best known prior result [3] achieving a rate of roughly $1/1,500,000$ in the 2-split-state model. In this chapter, we dramatically improve this status quo by building a rate booster that converts any augmented¹¹ non-malleable code into an augmented non-malleable code with a rate of $1/3$. As a stepping stone towards our goal, we see how relaxing the non-malleability conditions on NMCs to just uniformly random messages, which we call as non-malleable randomness encoders (NMREs), gives us a rate- $1/2$ NMRE in the 2-split-state model.

¹¹Augmented NMCs give an additional guarantee that one of the states is also simulatable along with the tampered message, and is hence independent of the original message.

Our Results. We obtain our construction of a rate- $\frac{1}{3}$ NMC via rate boosters which compile low rate NMCs into high rate ones. More specifically, we get the following result.

Informal Theorem. *There exists an efficient, information-theoretically secure ε -right-augmented¹² non-malleable code in the 2-split-state model with rate $1/3$. We show two instantiations of the scheme: the first gives us a strikingly simple construction (as we describe in Section 3.1.1) and achieves an error of $2^{-\Omega(\kappa^{1/5}/\text{polylog}(\kappa))}$; the second instantiation loses out on the simplicity but achieves an error of $\varepsilon = 2^{-\Omega(\frac{\kappa}{\log^3 \kappa})}$, where κ is the size of the message.*

We now give a detailed technical overview of our construction ideas, which starts with the precursor construction of rate- $1/2$ NMREs. Since our main goal is to build the near-optimal NMCs, we discuss the NMRE construction in our technical overview (Section 3.1.1) and proceed to look at the main NMC construction (Section 3.2)

3.1.1 Technical Overview

Our construction uses information-theoretic *one-time message authentication codes* (MACs) as well as *randomness extractors*. A randomness extractor Ext allows us to generate randomness from a source W with a min-entropy guarantee using a short *seed* (s) of true randomness. Message authentication codes $\text{MAC} = (\text{Tag}, \text{Vrfy})$ are secret key primitives which guarantee that even given $\text{Tag}_k(m)$ (using a secret key k), an adversary cannot generate m', t' such that $m' \neq m$ and $\text{Vrfy}_k(m', t') = 1$.

In order to highlight the challenges in building high-rate, 2-split-state NMCs, we start by looking at a related primitive—Non-malleable Randomness Encoders—which provide non-malleability guarantees for uniform random messages only. Similar to a 2-split-state NMC, a 2-split-state NMRE consists of two independently tamperable states L and R . Contrary to an NMC, where the encoder encodes arbitrary messages, an NMRE’s encoder outputs L and R such that they decode to a random string, and herein lies all the difference: while the problem of building high-rate NMCs has eluded researchers for over a decade, we show how to build NMREs with rate $\frac{1}{2}$.

To build the NMRE (see Figure 3.1), we pick the source w and seed s to a strong seeded extractor (Ext) as well as a key k_w to a message authentication code (MAC). The code consists of two states, left: $\ell \| w \| t$, and right: r , where ℓ, r are a low-rate non-malleable encoding of s, k_w and σ_w is a tag evaluated on w with k_w as the key. The codeword, if valid, decodes to $\text{Ext}(w; s)$. The high level idea behind security is that if the codeword was tampered and leads to a different $\underline{s}$, then it is independent of s and extractor security applies. This suffices to argue

¹²Right-augmented property guarantees that the right state of the NMC is simulatable independent of the message, along with the tampered message.

independence of the tampered message $\underline{m}$ from m . (In the case where $\underline{s}, \underline{k}_w = s, k_w$, MAC security comes into play.)

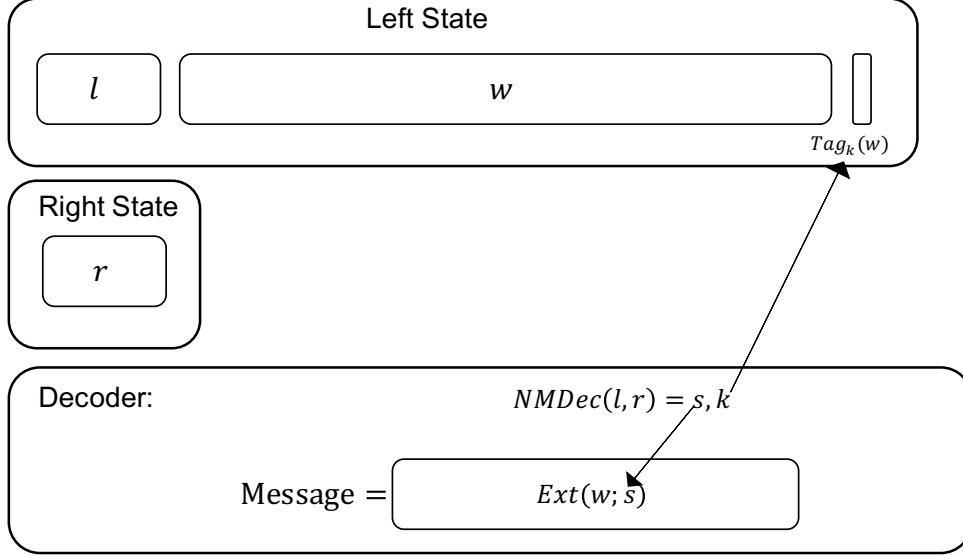


Figure 3.1: Construction of our Non-malleable Randomness Encoder

In order to extend this construction to encode an arbitrary message m , one option would be to reverse sample w and s such that $\text{Ext}(w; s) = m$. Unfortunately, this won't work because, on the one hand, we require the seed s to be short (as it is encoded using a poor-rate NMC) and, on the other hand, given a source w , there will be at most $2^{|s|}$ possible messages that could have been encoded. In other words, to obtain any meaningful security, s needs to be as long as the message. However, if s is long, the above approach will not yield rate gains. These orthogonal constraints are the main stumbling block which we overcome in this work. To do so, we need to somehow use $\text{Ext}(w; s)$ to “mask” the message m . Clearly, since $\text{Ext}(w; s)$ needs to be stored in a state different from the one storing w , we will consider a target code which has the form $(\ell \| w \| \sigma_w), (r \| c)$, where c would depend on the message as well as the extractor output $\text{Ext}(w; s)$ i.e., w, s, c “shares” m . While there are many candidates for c , the solution to our puzzle lies in figuring out a c with which we can prove non-malleability. Here's the tricky part: when the adversary learns the tampered message, she learns information that depends on $\text{Ext}(w; s)$. Furthermore, her tampering of r , which influences the tampered seed $\underline{s}$ and therefore $\text{Ext}(\underline{w}; \underline{s})$, also depends on $\text{Ext}(w; s)$. In a nutshell, regardless of what c is, it is clear that leveraging the security of the strong randomness extractor to argue non-malleability is not straight-forward.

Our first idea to overcome this challenge is to allow c to be such that $2\text{Ext}(\text{Ext}(w; s), c) = m$ where 2Ext is a 2-source extractor. As it turns out, this construction is secure for the following reason: even if the tampered output leaks some information about $\text{Ext}(w; s)$, the two source

extractor¹³ will ensure that the “masking” remains secure. This leads to an NMC with rate $\frac{1}{9}$ in the split-state model.

The main idea behind our final construction is to tackle the information leakage on $\text{Ext}(w; s)$ head-on. In particular, though the tampered message may leak information about $\text{Ext}(w; s)$, we can ensure that this information is useless as far as breaking non-malleability is concerned. In fact, we do this while further simplifying our construction: We set the second half of the right state to be $(c = \text{Ext}(w; s) \oplus m) \parallel \sigma_c$ where σ_c is a MAC tag of c evaluated on key k_c . In retrospect, our encoding scheme is nearly identical to that due to Kanukurthi, Obbattu and Sekar [76]. While [76] gave a four state construction, we merge states to obtain a two state construction.

We now offer a more detailed overview of the construction as well as the proof.

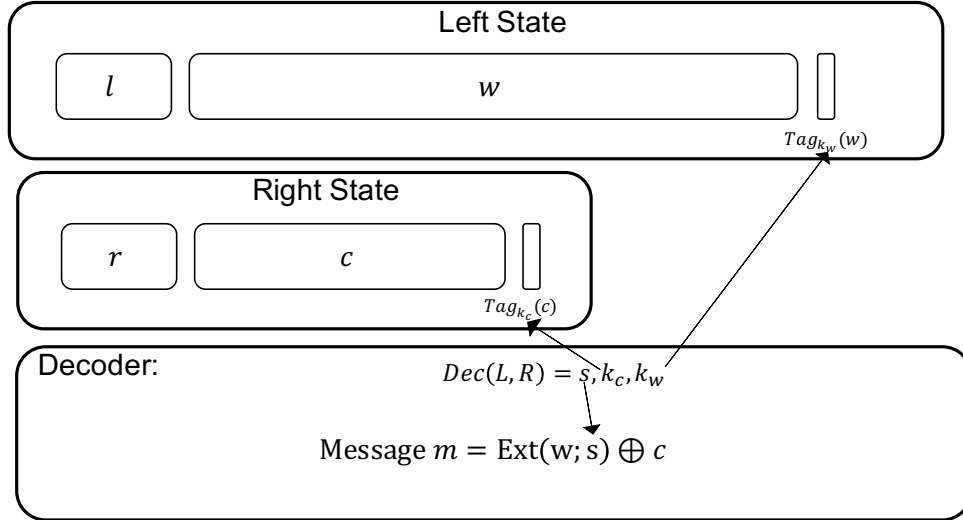


Figure 3.2: Construction of our Non-malleable Code

Overview of the construction: Blocks ℓ, r come from augmented non-malleable code. The encoder proceeds in steps: first, we randomly sample s, k_w, k_c, w (all independently of the message we are encoding), then we encode s, k_w, k_c using NMC into ℓ, r . We then set $c = \text{Ext}(w; s) \oplus m$, and evaluate σ_c as a MAC tag of c on key k_c , and σ_w as MAC tag of w on key k_w .

As building blocks, in addition to a message authentication code, and a strong seeded extractor, our construction uses a low-rate non-malleable code, (NMEnc, NMDec), which we shall use to encode the keys of the message authentication code and the seed for the seeded extractor. The overview of the construction can be found in the Figure 3.2. We will use the notation from the above-mentioned figure. Also, for a variable X , $\underline{X}$ will denote its tampering. We proceed with a slightly simplified sketch of the proof.

¹³A two source extractor takes two entropic sources and outputs perfect randomness.

Proof Overview In order to prove non-malleability we need to demonstrate the existence of a simulator whose outputs is indistinguishable from the output of the tampering experiment. The simulator doesn't use the message; however, it outputs a special symbol to indicate that the tampered message is unchanged. The simulator's output is run through a special wrapper function (typically called “Copy” function) that, in this case, outputs the original message. Our proof proceeds by partitioning our codeword space. Our simulator will pick a codeword at random, checks which partition it lies in to determine its output. We describe the partitions below:

- $\mathcal{P}_1 = \{(\ell, w, \sigma_w, r, c, \sigma_c) : (w, \sigma_w, c, \sigma_c) = (\underline{w}, \underline{\sigma_w}, \underline{c}, \underline{\sigma_c}) \wedge \text{NMDec}(\ell, r) = \text{NMDec}(\underline{\ell}, \underline{r})\}$.
This partition captures the scenario when, even after tampering, the inner codeword (ℓ, r) decodes to the same message, and W, σ_w, C, σ_c remain unchanged. In this case, the final codeword must decode to the same message.
- $\mathcal{P}_2 = \{(\ell, w, \sigma_w, r, c, \sigma_c) : (w, \sigma_w, c, \sigma_c) \neq (\underline{w}, \underline{\sigma_w}, \underline{c}, \underline{\sigma_c}) \wedge \text{NMDec}(\ell, r) = \text{NMDec}(\underline{\ell}, \underline{r})\}$.
 $\mathcal{P}_2$ captures the scenario when, the decoding of the inner code remains unchanged after tampering, while one of the pairs (W, σ_w) or (C, σ_c) are changed. We will show that if this event occurs then, using the security of MACs, the tampering is detected with overwhelming probability.
- $\mathcal{P}_3 = \{(\ell, w, \sigma_w, r, c, \sigma_c) : \text{NMDec}(\ell, r) \neq \text{NMDec}(\underline{\ell}, \underline{r})\}$. $\mathcal{P}_3$ captures the scenario that the inner code is non-trivially tampered and does not decode to $\text{Dec}(L, R)$. In this case, we will show that the tampered codeword is independent of the original message m .

The simulator generates the codeword $((L, W, \sigma_w), (R, \tilde{C}, \tilde{\sigma}_c))$ of a random message. If this simulated codeword is in $\mathcal{P}_1$, it outputs **same***. Recall that the wrapper function will then output the original message. If the simulated codeword is in $\mathcal{P}_2$, the simulator outputs $\perp$, else the simulator outputs $\text{Dec}\left((L, W, \sigma_w), (R, \tilde{C}, \tilde{\sigma}_c)\right)$. (Note that our code is right-augmented i.e., it satisfies a stronger notion of security where the right state of the codeword can be revealed without breaking non-malleability.)

To prove non-malleability, we need to show that this behaviour of the simulator is indistinguishable from that of the tampering experiment. To do this, we first show that the probability of a codeword being in any given partition is independent of the message¹⁴. Next, we show that the output of the tampering experiment is, in each case, indistinguishable from the simulator's output.

¹⁴This proof relies on the secret sharing property of the non-malleable code as well as the security of the strong randomness extractor.

For the case where the codeword is in partition $\mathcal{P}_1$, it is clear that the simulator output is identical to that of the tampering experiment. We therefore focus on the other two cases.

3.1.1.1 Case 1: Codeword is in $\mathcal{P}_2$ i.e., $\text{NMDec}(\underline{L}, \underline{R}) = \text{NMDec}(L, R)$.

Intuitively, we would like to argue that the tag keys K_w, K_c will remain securely hidden from the adversary, and if he decides to tamper with W or C he will not be able to fake tags σ_w, σ_c . Thus either the whole codeword remains untampered (in which case, we are in $\mathcal{P}_1$) or the new codeword will not be valid.

The standard approach would be to argue that if $\Pr(\text{NMDec}(\underline{L}, \underline{R}) = \text{NMDec}(L, R))$ is not too small then

$$\Pr[\text{tampered codeword is valid} \wedge (\underline{W}, \underline{C}) \neq (W, C) \mid \text{Dec}(\underline{L}, \underline{R}) = \text{Dec}(L, R)]$$

is negligible. However we have to be delicate here. For example if adversary wants to tamper with W he has access to L and knows that $\text{NMDec}(\underline{L}, \underline{R}) = \text{NMDec}(L, R)$. This reveals some information about R and thus adversary potentially might get hold of some partial information about the encoded data (and K_w, K_c in particular). This is why it is actually easier to directly argue that

$$\Pr[\text{Tampered codeword is valid} \wedge (\underline{W}, \underline{C}) \neq (W, C) \wedge \text{Dec}(\underline{L}, \underline{R}) = \text{Dec}(L, R)] \quad (3.1)$$

is negligible. Notice that the codeword will not be valid in only one of three cases: if $\text{NMDec}(\underline{L}, \underline{R}) = \perp$ or if one of the MACs on W or C does not verify correctly. Since $\text{NMDec}(\underline{L}, \underline{R}) = \text{NMDec}(L, R)$ we know that the only options left are the failures to verify MACs. Moreover we know that $(\underline{K}_w, \underline{K}_c) = (K_w, K_c)$, thus Inequality 3.1 can be rewritten:

$$\Pr[\text{Vrfy}_{K_w}(W, \sigma_w) = \text{Vrfy}_{K_c}(C, \sigma_c) = 1 \wedge (\underline{W}, \underline{C}) \neq (W, C) \wedge \text{NMDec}(\underline{L}, \underline{R}) = \text{NMDec}(L, R)] \quad (3.2)$$

is negligible. Now we can upper-bound the term in the Inequality 3.2 by the following

$$\Pr[\text{Vrfy}_{K_w}(W, \sigma_w) = \text{Vrfy}_{K_c}(C, \sigma_c) = 1 \wedge (\underline{W}, \underline{C}) \neq (W, C)].$$

Which by the union bound can be upper-bounded with

$$\Pr[\text{Vrfy}_{K_w}(W, \sigma_w) = 1 \wedge \underline{W} \neq W] + \Pr[\text{Vrfy}_{K_c}(C, \sigma_c) = 1 \wedge \underline{C} \neq C].$$

Finally we can argue that each of elements of the sum is negligible. Notice that when tampering with W adversary has access to L but that can not reveal any information about K_w since every non-malleable code is a secret sharing scheme. The rest follows from the security of MACs.

3.1.1.2 Case 2: Codeword is in $\mathcal{P}_3$ i.e., $\text{NMDec}(\underline{L}, \underline{R}) \neq \text{NMDec}(L, R)$.

In this case, we will follow the adventures of the seed S ; the MACs and keys do not play any role here. In fact, for the purposes of this proof sketch we will ignore the MAC keys and tags. We will also assume that this case (i.e., codeword $\in \mathcal{P}_3$) occurs with substantial probability (else we don't have to worry about it). In such a case, we will argue that the final message is independent of the original message.

To provide a proof sketch for this case, we consider the following modified construction: Let $(\ell, r) \leftarrow \text{NMEnc}(s)$, where $(\text{NMEnc}, \text{NMDec})$ is the low-rate non-malleable code. Pick a random W , the source for a strong extractor. Then the left state of the final non-malleable code is $\ell \| w$. The right state is $r \| (c = \text{Ext}(w; s))$. In other words, we've simply removed the MAC keys and tags. While this is obviously not a secure non-malleable code, we use this construction for the sake of ease of exposition.

By the properties of the augmented NMC¹⁵ we know that there is a simulator NMSim that outputs $S^{\text{sim}}, L^{\text{sim}}$. Given that we are in the case where $\text{NMDec}(\underline{L}, \underline{R}) \neq \text{NMDec}(L, R)$, S^{sim} is independent of S . As discussed earlier, this knowledge alone doesn't suffice for us. The tampering of R (which influences $\underline{S}$) depends on C which in turn uses the extractor output. In order to output the tampered message, we need to be able to compute $\text{Ext}(\underline{W}; \underline{S})$. We handle this by using a Markov style argument, as we shall see shortly.

Non-malleability of the underlying NMC: To use the non-malleability of the inner code, $(\text{NMEnc}, \text{NMDec})$, consider the tampering functions f^*, g^* on (L, R) that sample $W, \tilde{C}$ uniformly and then compute $f^*(L) = f_1(L, W)$ and $g^*(R) = g_1(R, \tilde{C})$. Let $\underline{S} = \text{NMDec}(f^*(L), g^*(R))$. By the augmented non-malleability of $(\text{NMEnc}, \text{NMDec})$, there exists a simulator NMSim such that

$$\underline{S}, L, S, W, \tilde{C} \approx \text{Copy}(\mu, \text{NMSim}), S, W, \tilde{C}, \quad (3.3)$$

where the output of NMSim depends on $W, \tilde{C}$ and the functions f_1, g_1 . Recall (from definition 2.6) that NMSim is the joint distribution consisting of $(\text{NMSim}_1, L^{\text{sim}})$. We further denote NMSim_1 by $\underline{S}^{\text{sim}}$, when NMSim_1 is conditioned to not output **same*** or $\perp$. We denote by E_1 the event that $(L, W, \sigma_w, R, \tilde{C}, \tilde{\sigma}_c) \in \mathcal{P}_3$ and E_2 the event that $S \neq \underline{S}$. Note that E_1 is the same as E_2 . We denote by E_3 the event that $\text{NMSim}_1 \notin \{(\text{same}^*, S)\}$. We first show that $\Pr[E_3]$ is

¹⁵Augmented NMC is a natural extension of NMC (and equivalent of strong extraction property). We require that $\text{NMDec}(\underline{L}, \underline{R})$ is equal or independent of $\text{NMDec}(L, R)$ and revealing L or R doesn't ruin this independence.

high.

From here, using properties of statistical distance, and expanding out the definition of **Copy**, we show that:

$$(\underline{S}, L, S, W, \tilde{C})|_{E_2} \approx (\underline{S}^{\text{Sim}}, L^{\text{Sim}}, S, W, \tilde{C})|_{E_3} \quad (3.4)$$

We next show that the (average) conditional entropy of W given $\underline{S}^{\text{Sim}}, \underline{L}^{\text{Sim}}, \text{Ext}(W; \underline{S})$ is high; here $\underline{L}^{\text{Sim}}$ and $\underline{W}$ are tamperings of L^{Sim}, W and can be computed as their deterministic function. In particular, we have that

$$\tilde{\mathbf{H}}_{\infty}(W | \underline{S}^{\text{Sim}}, \underline{L}^{\text{Sim}}, \text{Ext}(W; \underline{S}^{\text{Sim}}))$$

is high.

Further since S is independent of $\underline{S}^{\text{Sim}}$, we use the strong extractor property of **Ext**, to show the following:

$$\text{Ext}(W; S) \approx U | S, L^{\text{Sim}}, \tilde{C}, \underline{L}^{\text{Sim}}, \text{Ext}(W; \underline{S}^{\text{Sim}}), \underline{S}^{\text{Sim}} \quad (3.5)$$

In order to proceed with the proof, we will prove a statement similar to the one above; except that instead of the tampered seed being simulated, we will let it be the tampered seed (in the underlying non-malleable code). (Eventually we would need to replace $\tilde{C}$ by $C := \text{Ext}(W; S) \oplus m$.)

For the remainder of the proof, let $L, W, R, \tilde{C}$ be distributed as $L, W, \sigma_w, R, \tilde{C}, \tilde{\sigma}_c|_{E_1}$. Also, let $\underline{S}, \underline{L}, \underline{R}, \underline{W}$ be the corresponding tampered random variables. After tampering conditioned on the event that $(L, W, \sigma_w, R, \tilde{C}, \tilde{\sigma}_c) \in \mathcal{P}_3$. We shorthand the distribution in the LHS and RHS of every equation I by IA and IB , respectively.

$$\text{Ext}(W; \underline{S}), S, L, \tilde{C}, \underline{L}, \text{Ext}(W; \underline{S}) \approx \text{Ext}(W; \underline{S}), S^{\text{sim}}, L, \tilde{C}, \underline{L}, \text{Ext}(W; \underline{S}^{\text{sim}}) \quad (3.6)$$

$$\mathcal{U}, S, L, \tilde{C}, \underline{L}, \text{Ext}(W; \underline{S}) \approx \mathcal{U}, S^{\text{sim}}, L, \tilde{C}, \underline{L}, \text{Ext}(W; \underline{S}^{\text{sim}}) \quad (3.7)$$

Applying triangle inequality on Inequalities (3.5), (3.6), (3.7), we get that

$$\text{Ext}(W; S) \approx U | S, L, \tilde{C}, \underline{L}, \text{Ext}(W; \underline{S}), \underline{S} \quad (3.8)$$

Note that in the equation above, there is no dependence on m on either side as $\tilde{C}$ is independent of m . Ultimately, we would like to say that the output of the tampering experiment is

indistinguishable from the simulated output. We accomplish this in three steps:

1. Adding R . In equation 3.8, the only information correlated to W and R is $\underline{S}$. Since $\text{Ext}(W; S) \approx \mathcal{U}$ even given $\underline{S}$, we can safely add R to Equation 3.8.

$$\text{Ext}(W; S) \approx U|S, L, \tilde{C}, \underline{L}, \text{Ext}(\underline{W}; \underline{S}), \underline{S}, R, g_1(R, \tilde{C}, \tilde{\sigma}_c) .$$

From here, we would ideally like to drop $\tilde{C}$ and somehow bring back the dependence on m via C . For now, we drop $\tilde{C}$

$$\text{Ext}(W; S) \approx U|S, L, \underline{L}, \text{Ext}(\underline{W}; \underline{S}), \underline{S}, R, \underline{R} . \quad (3.9)$$

The way we'll bring C is to condition $\tilde{C}$ on being a "cipher of m ". For that we first need to prove that $\tilde{C}$ is independent of W given appropriate auxiliary information.

2. Capturing $\tilde{C}$'s correlation with W . In this step, we will prove that $\tilde{C}$ is independent of W given $S, L, \underline{L}, \text{Ext}(\underline{W}; \underline{S}), \underline{S}, R, \underline{R}$. We first observe that $\tilde{C}$ is independent of W given (L, R, S) . Now we would like to add the other random variables in the auxiliary information. We use the a Lemma due to Dziembowski and Pietrzak which states that independence in the presence of additional auxiliary information is indeed possible, provided it satisfies a few properties:

- The auxiliary information may be computed in multiple steps.
- Computation in all of the steps can use (L, R, S) and the part of the auxiliary information generated in previous steps.
- Computation in a given step can either depend on $\tilde{C}$ or W but not both.

By computing auxiliary information in the order $\underline{L}$ followed by $\underline{R}$ followed by $\text{Ext}(\underline{W}; \underline{S})$, we can easily prove that $\tilde{C}$ is independent of W given $S, L, \underline{L}, \text{Ext}(\underline{W}; \underline{S}), \underline{S}, R, \underline{R}$.

3. Conditioning $\tilde{C}$ appropriately Since W is independent of $\tilde{C}$ given appropriate auxiliary information, in Equation 3.9, we can condition $\tilde{C}$ to either be $m \oplus \text{Ext}(W, S)$ or $m \oplus U$. (Note that the latter is identical to C .) By doing so, Equation 3.9 will lead to the following $C, S, L, \underline{L}, \text{Ext}(W; S), \underline{S}, R, \underline{R} \approx \mathcal{U}, S, L, \underline{L}, \text{Ext}(W; S), \underline{S}, R, \underline{R}$, where $\underline{R}, \underline{S}$ are appropriately computed.

The desired result follows by observing that the tampered codeword is a function of

$$\underline{L}, \underline{R}, \text{Ext}(\underline{W}; \underline{S}), C, R .$$

Putting it Together. So far, we’ve described the simulator and sketched the proof for showing that the simulated output is indistinguishable from the tampered output in each of the cases. To complete the proof, we need to combine all three cases and, in particular, the probability that the codewords (tampered vs simulated) lie in each of the partitions needs to be analysed.

Candidate Instantiation While we can turn any augmented non-malleable code (or randomness encoder) into a good rate non-malleable code, the most strikingly simple result can be obtained using [5]. To encode a message m all we will need is *an affine evasive function* h . It is a function $h : \mathbb{F}_p \rightarrow \mathcal{M} \cup \perp$ such that $\Pr(h(aU + b) \neq \perp \mid h(U) = m)$ is negligible for all a, b, m , and $U \mid h(U) = m$ should be efficiently samplable, the construction of the said function can be found in [5, 1]. The encoding procedure is described in Figure 3.3.

Short and Simple: The Encoding Procedure:

1. Sample s, k_w, k_c, w uniformly at random.
2. Sample x uniformly random, such that $h(x) = s, k_w, k_c$.
3. Sample $\ell, r \in \mathbb{F}_p^n$ uniformly random, such that $\langle \ell, r \rangle = x$.
4. Evaluate $c = \text{Ext}(w; s) \oplus m$.
5. Calculate MACs $\sigma_c = \text{Tag}_{k_c}(c)$ and $\sigma_w = \text{Tag}'_{k_w}(w)$

The final output is: on the left: ℓ, w, σ_w , and on the right: r, c, σ_c .

Figure 3.3: Simple non-malleable code with a great rate. Here h is an affine evasive function. The decoding procedure is analogous: the decoder inverts Steps 3 and 2, obtains keys k_w, k_c , verifies MACs from the Step 5 and proceeds to obtain the message via the Step 4. If in Step 2 the function h outputs $\perp$, then the decoder aborts and outputs $\perp$.

3.1.2 Organization of the Chapter

We discuss our construction of the near-optimal rate NMC in Section 3.2. In Appendix 3.3, we give a proof that the underlying NMC instantiation used by our scheme is indeed augmented secure.

3.2 Our Construction

We begin by giving the necessary building blocks for our construction.

3.2.1 Building Blocks

We set $N = 2\kappa^* + 8n$, and consider the following building blocks.

- Let $(\text{NMEnc} : \{0, 1\}^\kappa \rightarrow \{0, 1\}^n \times \{0, 1\}^n, \text{NMDec} : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}^\kappa)$ be an ε_1 -augmented non-malleable code in the 2-split-state model (as in Definition 2.6).
- Let Ext be the $(N, k, d, \kappa^*, \varepsilon_2)$ -strong seeded extractor from Lemma 2.10 with $k = \kappa^* + 4n$, and $\varepsilon_2 = 2^{-\Omega(\kappa/\log \kappa)}$.
- Let $(\text{Tag}, \text{Vrfy})$ be the one time ε_3 -message authentication codes from Lemma 2.13 with key length $\tau = \Theta(\kappa)$, messages of size at most N , tag length $t = \Theta(\kappa)$, and $\varepsilon_3 = 2^{-\Omega(\kappa)}$.

3.2.2 Construction Overview

Overview. The encoder chooses a source w and a seed s corresponding to a seeded extractor Ext and then computes $c := m \oplus \text{Ext}(w; s)$, where $\oplus$ is the bitwise XOR. Further, it authenticates the source w and the ciphertext c to get the tags σ_w and σ_c respectively, generates the non-malleable encoding (using the underlying NMC) of the seed and the authentication keys to get (ℓ, r) and finally outputs (ℓ, w, σ_w) as one state and (r, c, σ_c) as the other. The construction is as described in Figure 3.4.

Enc(m): • $s \in_R \{0, 1\}^d$, $k_w, k_c \in_R \{0, 1\}^\tau$, $w \in_R \{0, 1\}^N$ • $c = m \oplus \text{Ext}(w; s)$. • $\sigma_w = \text{Tag}_{k_w}(w)$ and $\sigma_c = \text{Tag}_{k_c}(c)$ • $(\ell, r) \leftarrow \text{NMEnc}(\mu)$ where $\mu = (s, k_w, k_c)$ • Output $(\ell, w, \sigma_w), (r, c, \sigma_c)$	Dec($(\ell, w, \sigma_w), (r, c, \sigma_c)$): • $(s, k_w, k_c) = \text{NMDec}(\ell, r)$. • If $\text{Vrfy}_{k_w}(w, \sigma_w) = 1$ and $\text{Vrfy}_{k_c}(c, \sigma_c) = 1$, Output $m = \text{Ext}(w; s) \oplus c$. - Else, Output $\perp$.
---	--

Figure 3.4: Our NMC Construction

Theorem 3.1 *Let $(\text{NMEnc}, \text{NMDec})$, Ext and $(\text{Tag}, \text{Vrfy})$ be the 2-split-state NMC, strong seeded randomness extractor and message authentication codes, as in Section 3.2.1. Then, for any $\kappa^* \in \omega(n)$, the construction in Figure 3.4 gives an efficient $O(\sqrt{\varepsilon_1} + 2^{-\Omega(\kappa/\log \kappa)})$ -augmented non-malleable code (specifically, right-augmented, i.e., the simulator outputs the right state along with the tampered message) from κ^* -bit messages to at most $(3 + o(1))\kappa^*$ -bit codewords.*

Specifically, when we instantiate the NMC, $(\text{NMEnc}, \text{NMDec})$ with the construction from Theorem 2.1, we get the following parameters, and a construction which is conceptually very simple, as shown in Figure 3.3.

Corollary 3.1 *For any integer $\kappa^* > 0$, $\varepsilon^* \in 2^{-\Omega((\kappa^*)^{1/5}/\text{polylog}(\kappa^*))}$, there is an efficient ε^* -right-augmented non-malleable code from κ^* -bit messages to $(3 + o(1))\kappa^*$ -bit codewords in the 2-split-state model.*

Further, using $(\text{NMEnc}, \text{NMDec})$ from Corollary 3.3, and setting $\kappa^* = n \log n$ we can get the following instantiation, with a better error at the expense of being more complicated.

Corollary 3.2 *For any integer $\kappa^* > 0$, $\varepsilon^* \in 2^{-\Omega(\kappa^*/\log^3 \kappa^*)}$, there is an efficient ε^* -right-augmented non-malleable code from κ^* -bit messages to $(3 + o(1))\kappa^*$ -bit codewords in the 2-split-state model.*

The proofs of Corollaries 3.1 and 3.2 follow from a direct substitution of parameters from Theorems 2.1 and 2.2, respectively, in Theorem 3.1. Hence, we proceed to prove Theorem 3.1 below.

3.2.3 Security Proof

The correctness of the construction is immediate by definition. We now prove the desired non-malleability.

Fix any $m \in \{0, 1\}^{\kappa^*}$, tampering functions $(f, g) \in \mathcal{F}_{\text{split}}$. Throughout this proof, we use the notation $\underline{X}$ to denote any part X of the message/codeword after tampering. Let $f = (f_1, f_2, f_3)$ and $g = (g_1, g_2, g_3)$, where

$$f_1 : \{0, 1\}^{n+N+t} \rightarrow \{0, 1\}^n, \quad f_2 : \{0, 1\}^{n+N+t} \rightarrow \{0, 1\}^N, \quad f_3 : \{0, 1\}^{n+N+t} \rightarrow \{0, 1\}^t,$$

and

$$g_1 : \{0, 1\}^{n+\kappa^*+t} \rightarrow \{0, 1\}^n, \quad g_2 : \{0, 1\}^{n+\kappa^*+t} \rightarrow \{0, 1\}^{\kappa^*}, \quad g_3 : \{0, 1\}^{n+\kappa^*+t} \rightarrow \{0, 1\}^t.$$

Let $W, S, K_w, K_c, C, \sigma_w, \sigma_c, L, R$ be the distributions of $w, s, k_w, k_c, c, \sigma_w, \sigma_c, \ell, r$ ¹⁶ sampled in $\text{Enc}(m)$. Now, apply the tampering to get $(\underline{L}, \underline{W}, \underline{\sigma}_w) = f(L, W, \sigma_w)$, $(\underline{R}, \underline{C}, \underline{\sigma}_c) = g(R, C, \sigma_c)$, decode to get $\underline{\mu} = \text{NMDec}(\underline{L}, \underline{R})$ and $\underline{M} = \text{Dec}((\underline{L}, \underline{W}, \underline{\sigma}_w), (\underline{R}, \underline{C}, \underline{\sigma}_c))$. Further, note that the distribution $\underline{M}$ is the distribution $\text{Tamper}_{f,g}^m$ itself.

Our proof proceeds by partitioning the sample space of the codeword, $\{0, 1\}^n \times \{0, 1\}^N \times \{0, 1\}^t \times \{0, 1\}^n \times \{0, 1\}^{\kappa^*} \times \{0, 1\}^t$ and proving non-malleability in each of the partitions. Consider the following three partitions.

$$\mathcal{P}_1 = \{(\ell, w, \sigma_w, r, c, \sigma_c) : (f_2(\ell, w, \sigma_w), f_3(\ell, w, \sigma_w), g_2(r, c, \sigma_c), g_3(\ell, c, \sigma_c)) = (w, \sigma_w, c, \sigma_c), \\ \text{NMDec}(f_1(\ell, w, \sigma_w), g_1(r, c, \sigma_c)) = \text{NMDec}(\ell, r)\},$$

$$\mathcal{P}_2 = \{(\ell, w, \sigma_w, r, c, \sigma_c) : (f_2(\ell, w, \sigma_w), f_3(\ell, w, \sigma_w), g_2(r, c, \sigma_c), g_3(\ell, c, \sigma_c)) \neq (w, \sigma_w, c, \sigma_c), \\ \text{NMDec}(f_1(\ell, w, \sigma_w), g_1(r, c, \sigma_c)) = \text{NMDec}(\ell, r)\},$$

and

$$\mathcal{P}_3 := \{(\ell, w, \sigma_w, r, c, \sigma_c) : \text{NMDec}(f_1(\ell, w, \sigma_w), g_1(r, c, \sigma_c)) \neq \text{NMDec}(\ell, r)\},$$

The event that the codeword is in partition $\mathcal{P}_1$ corresponds to the event when, even after tampering, the decoding of the inner codeword (L, R) remains unchanged, and also W, σ_w, C, σ_c remain unchanged, and hence the final tampered codeword must decode to the original message.

¹⁶We slightly abuse notation and let σ_w and σ_c denote both the distribution and the sample of the authentication tags σ_w and σ_c respectively.

The event that the codeword is in partition $\mathcal{P}_2$ corresponds to the event that the decoding of the inner code remains unchanged after tampering, while one of the pairs (W, σ_w) or (C, σ_c) are changed. We will show that if this event occurs then, using the security of MACs, the tampering is detected with overwhelming probability. The event that the codeword is in partition $\mathcal{P}_3$ corresponds to the event that the inner code is non-trivially tampered and does not decode to $\text{Dec}(L, R)$. In this case, we will show that the tampered codeword (and hence its decoding) is independent of the original message m .

We begin by showing that the event that the codeword $(L, W, \sigma_w, R, C, \sigma_c) \in \mathcal{P}_i$ for some $i \in \{1, 2, 3\}$ is independent of the message m even given R, C, σ_c .

Let $\tilde{C}$ be uniform in $\{0, 1\}^{\kappa^*}$ and independent of S, K_w, K_c, W, σ_w , and hence also of L, R . Let $\tilde{\sigma}_c = \text{Tag}_{K_c}(\tilde{C})$.

Let $b_1, b_2, \tilde{b}_2$ be boolean random variables defined as follows. The random variable b_1 indicates whether W, σ_w changes after tampering, i.e., $b_1 = 1$ if and only if $(f_2(L, W, \sigma_w), f_3(L, W, \sigma_w)) = (W, \sigma_w)$. Similarly, the random variable b_2 (respectively, $\tilde{b}_2$) indicates whether C, σ_c (respectively, $\tilde{C}, \tilde{\sigma}_c$) changes after tampering, i.e., $b_2 = 1$ (respectively, $\tilde{b}_2 = 1$) if and only if $(g_2(R, C, \sigma_c), g_3(R, C, \sigma_c)) = (C, \sigma_c)$ (respectively, $(g_2(R, \tilde{C}, \tilde{\sigma}_c), g_3(R, \tilde{C}, \tilde{\sigma}_c)) = (\tilde{C}, \tilde{\sigma}_c)$). Notice that for any i , whether the event that $(L, W, \sigma_w, R, C, \sigma_c) \in \mathcal{P}_i$ (respectively, $(L, W, \sigma_w, R, \tilde{C}, \tilde{\sigma}_c) \in \mathcal{P}_i$) occurs is determined by the random variables $L, R, f_1(L, W, \sigma_w), g_1(R, C, \sigma_c), b_1$, and b_2 (respectively, $L, R, f_1(L, W, \sigma_w), g_1(R, \tilde{C}, \tilde{\sigma}_c), b_1$, and $\tilde{b}_2$).

Lemma 3.1

$$\Delta \left(C, \sigma_c; \tilde{C}, \tilde{\sigma}_c \mid L, R, f_1(L, W, \sigma_w), b_1 \right) \leq 4\varepsilon_1 + 2^{-\Omega(\kappa/\log \kappa)}.$$

Proof: By Lemma 2.14, we have that one of the shares of a split-state non-malleable code is $2\varepsilon_1$ -independent of the message. In other words, there is a random variable L^* independent of S, K_w, K_c such that

$$L, S, K_w, K_c \approx_{2\varepsilon_1} L^*, S, K_w, K_c,$$

Introducing fresh random variable W that is sampled independent of L, S, K_w, K_c , and observing that σ_w is a deterministic function of K_w and W , we have that

$$L, S, K_w, K_c, W, \sigma_w \approx_{2\varepsilon_1} L^*, S, K_w, K_c, W, \sigma_w. \quad (3.10)$$

Let b_1^* be a boolean random variable that is 1 if and only if $(f_2(L^*, W, \sigma_w), f_3(L^*, W, \sigma_w)) = (W, \sigma_w)$. By Lemma 2.1, the average min-entropy of W given $b_1^*, f_1(L^*, W, \sigma_w)$ is at least

$N - n - 1$, which is at least k . Thus, by Lemma 2.10, we have that

$$\Delta(\text{Ext}(W; S) ; U \mid b_1^*, f_1(L^*, W, \sigma_w), L^*, K_w, K_c, S) \leq \varepsilon_2 , \quad (3.11)$$

where U is uniform in $\{0, 1\}^{\kappa^*}$ and independent of all other random variables.

Applying Lemma 2.2 to Equation 3.10 twice, we get

$$\Delta(b_1^*, f_1(L^*, W, \sigma_w), L^* ; b_1, f_1(L, W, \sigma_w), L \mid \text{Ext}(W; S), K_w, K_c, S) \leq 2\varepsilon_1 , \quad (3.12)$$

and

$$\Delta(b_1^*, f_1(L^*, W, \sigma_w), L^* ; b_1, f_1(L, W, \sigma_w), L \mid U, K_w, K_c, S) \leq 2\varepsilon_1 . \quad (3.13)$$

By triangle inequality on inequalities 3.11, 3.12, 3.13, we get that

$$\Delta(\text{Ext}(W; S) ; U \mid b_1, f_1(L, W, \sigma_w), L, K_w, K_c, S) \leq \varepsilon_2 + 4\varepsilon_1 .$$

Notice that the distribution R is independent of W and is only correlated to (L, K_w, K_c, S) (as $(L, R) \equiv \text{NMEnc}(K_w, K_c, S)$). Hence by applying Lemma 2.2, we can include R to get that

$$\Delta(\text{Ext}(W; S) ; U \mid b_1, f_1(L, W, \sigma_w), L, R, K_w, K_c, S) \leq \varepsilon_2 + 4\varepsilon_1 .$$

Finally, observing that given $(b_1, f_1(L, W, \sigma_w), L, R, K_w, K_c, S)$, the random variable $\text{Ext}(W; S) \oplus m$ is distributed as C , and $U \oplus m$ is distributed as $\tilde{C}$, we get the desired result by another application of Lemma 2.2. $\square$

The following lemma bounds the probability that the codeword is in the partition $\mathcal{P}_2$ and doesn't decode to $\perp$.

Lemma 3.2

$$\Pr[\text{Tamper}_{f,g}^m \neq \perp \wedge (L, W, \sigma_w, R, C, \sigma_c) \in \mathcal{P}_2] \leq 4\varepsilon_1 + 2^{-\Omega(\kappa)} .$$

Proof: Since the codeword is in $\mathcal{P}_2$, we have that $\text{Dec}(L, R) = \text{Dec}(\underline{L}, \underline{R})$, and hence K_w, K_c remain unchanged after tampering. Moreover, one of the pairs (W, σ_w) or (C, σ_c) has been tampered with. Assume without loss of generality that (W, σ_w) has been changed after tampering. By the security of MAC, this tampering will be detected by our decoding algorithm as long as the tampering is independent of K_w . We argue using the secret sharing property of non-malleable codes that this is indeed the case since L is almost independent of K_w .

By Lemma 2.14, we have that one of the shares of a split-state non-malleable code is $2\varepsilon_1$ -

independent of the message. In other words, there is a random variable L^* independent of S, K_w, K_c such that

$$L, S, K_w, K_c \approx_{2\varepsilon_1} L^*, S, K_w, K_c ,$$

Introducing fresh random variable W that is sampled independent of L, S, K_w, K_c , and observing that σ_w is a deterministic function of K_w and W , we have that

$$L, S, K_w, K_c, W, \sigma_w \approx_{2\varepsilon_1} L^*, S, K_w, K_c, W, \sigma_w .$$

Also, by definition of one-time message authentication codes, we have that

$$\Pr[\text{Vrfy}_{K_w}(f_2(L^*, W, \sigma_w), f_3(L^*, W, \sigma_w)) = 1 \wedge (f_2(L^*, W, \sigma_w), f_3(L^*, W, \sigma_w)) \neq (W, \sigma_w)] = 2^{-\Omega(\kappa)} .$$

By triangle inequality, we get

$$\Pr[\text{Vrfy}_{K_w}(f_2(L, W, \sigma_w), f_3(L, W, \sigma_w)) = 1 \wedge (f_2(L, W, \sigma_w), f_3(L, W, \sigma_w)) \neq (W, \sigma_w)] \leq 2^{-\Omega(\kappa)} + 2\varepsilon_1 .$$

Similarly, there exists R^* independent of S, K_w, K_c such that

$$R, S, K_w, K_c \approx_{2\varepsilon_1} R^*, S, K_w, K_c ,$$

and observing that C is independent of R, K_w, K_c given S , we have that

$$R, S, K_w, K_c, C, \sigma_c \approx_{2\varepsilon_1} R^*, S, K_w, K_c, C, \sigma_c .$$

This implies, via an argument similar as above that

$$\Pr[\text{Vrfy}_{K_c}(g_2(R, C, \sigma_c), f_3(R, C, \sigma_c)) \wedge (g_2(R, C, \sigma_c), g_3(R, C, \sigma_c)) \neq (C, \sigma_c)] \leq 2^{-\Omega(\kappa)} + 2\varepsilon_1 .$$

The desired result then follows from a simple union bound by observing that if $\text{Tamper}_{f,g}^m \neq \perp$, and the codeword is in $\mathcal{P}_2$, i.e., K_w, K_c remain unchanged after tampering, while W, σ_w, C, σ_c are changed after tampering, then we must have that

- $\text{Vrfy}_{K_w}(f_2(L, W, \sigma_w), f_3(L, W, \sigma_w)) = 1$
- $\text{Vrfy}_{K_c}(g_2(R, C, \sigma_c), f_3(R, C, \sigma_c)) = 1,$
- At least one of $(f_2(L, W, \sigma_w), f_3(L, W, \sigma_w)) \neq (W, \sigma_w)$ or $(g_2(R, C, \sigma_c), g_3(R, C, \sigma_c)) \neq (C, \sigma_c).$

□

Lemma 3.3 For any fixed k_w, k_c , and any $\alpha > 2\varepsilon_1 + 2^{-d}$, if $\Pr[(L, W, \sigma_w, R, \tilde{C}, \tilde{\sigma}_c) \in \mathcal{P}_3] \geq \alpha$, then the statistical distance between

$$\text{Tamper}_{f,g}^m, C, R, \sigma_c \big|_{(L, W, \sigma_w, R, C, \sigma_c) \in \mathcal{P}_3}$$

and

$$\text{Dec}(f(L, W, \sigma_w), g(R, \tilde{C}, \tilde{\sigma}_c)), R, \tilde{C}, \tilde{\sigma}_c \big|_{(L, W, \sigma_w, R, \tilde{C}, \tilde{\sigma}_c) \in \mathcal{P}_3}$$

is at most $2^{-\Omega(\kappa/\log \kappa)} + \frac{4\varepsilon_1}{\alpha}$.

Proof:

In this lemma, we wish to argue that the decoding of the tampered codeword is independent of the message m . We begin by using that since the codeword is in $\mathcal{P}_3$, the output of the tampered codeword for the inner code $\text{NMEnc}(S, k_w, k_c)$ is independent of S which is the seed of the strong extractor.

We will use the non-malleability of the inner code, $(\text{NMEnc}, \text{NMDec})$. Consider the tampering functions f^*, g^* on (L, R) that sample $W, \tilde{C}$ uniformly and then compute $f^*(L) = f_1(L, W, \text{Tag}_{k_w}(W))$ and $g^*(R) = g_1(R, \tilde{C}, \text{Tag}_{k_c}(\tilde{C}))$. Let $\underline{S}, \underline{K}_w, \underline{K}_c = \text{NMDec}(f^*(L), g^*(R))$.¹⁷ By the ε_1 -augmented non-malleability of $(\text{NMEnc}, \text{NMDec})$, there exists a simulator NMSim such that

$$\underline{S}, \underline{K}_w, \underline{K}_c, L, S, W, \tilde{C} \approx_{\varepsilon_1} \text{Copy}(\mu, \text{NMSim}), S, W, \tilde{C}, \quad (3.14)$$

where the output of NMSim depends on $W, \tilde{C}$ and the functions f_1, g_1 . Recall (from definition 2.6) that NMSim outputs a joint distribution $(\text{NMSim}_1, L^{\text{Sim}})$. We further parse NMSim_1 as $(\underline{S}^{\text{Sim}}, \underline{K}_w^{\text{Sim}}, \underline{K}_c^{\text{Sim}})$, when NMSim_1 is conditioned to not output same^* or $\perp$. If $\text{NMSim}_1 = \perp$, set $(\underline{S}^{\text{Sim}}, \underline{K}_w^{\text{Sim}}, \underline{K}_c^{\text{Sim}}) = (\perp, \perp, \perp)$. The event that $(L, W, \sigma_w, R, \tilde{C}, \tilde{\sigma}_c) \in \mathcal{P}_3$ is the same as $(\underline{S}, \underline{K}_w, \underline{K}_c) \neq (S, k_w, k_c)$. By inequality (3.14) and the hypothesis of the lemma, we have that $\Pr[\text{NMSim}_1 \notin \{\text{same}^*, (S, k_w, k_c)\}] \geq \alpha - \varepsilon_1$. By Lemma 2.5, we have that

$$\begin{aligned} (\underline{S}, \underline{K}_w, \underline{K}_c, L, S, W, \tilde{C}) \big|_{(\underline{S}, \underline{K}_w, \underline{K}_c) \neq (S, k_w, k_c)} &\approx_{\frac{2\varepsilon_1}{\alpha}} \text{Copy}(\mu, \text{NMSim}), S, W, \tilde{C} \big|_{\text{NMSim}_1 \notin \{\text{same}^*, (S, k_w, k_c)\}} \\ (\underline{S}, \underline{K}_w, \underline{K}_c, L, S, W, \tilde{C}) \big|_{(\underline{S}, \underline{K}_w, \underline{K}_c) \neq (S, k_w, k_c)} &\approx_{\frac{2\varepsilon_1}{\alpha}} (\underline{S}^{\text{Sim}}, \underline{K}_w^{\text{Sim}}, \underline{K}_c^{\text{Sim}}, L^{\text{Sim}}, S, W, \tilde{C}) \big|_{\text{NMSim}_1 \notin \{\text{same}^*, (S, k_w, k_c)\}} \end{aligned} \quad (3.15)$$

¹⁷For the purpose of this proof alone, we slightly abuse notation and let $\underline{S}, \underline{K}_w, \underline{K}_c$ denote the tampered seed and the MAC keys when the codeword $((L, W, \text{Tag}_{k_w}(W)), (R, \tilde{C}, \text{Tag}_{k_c}(\tilde{C})))$ is tampered using functions f, g .

We have that for any w ,

$$\Pr[W = w | \text{NMSim}_1 \notin \{\text{same}^*, (S, k_w, k_c)\}] \leq \frac{\Pr[W = w]}{\Pr[\text{NMSim}_1 \notin \{\text{same}^*, (S, k_w, k_c)\}]} \leq \frac{2^{-N}}{\alpha - \varepsilon_1} \leq 2^{d-N},$$

where we use that $\alpha > 2\varepsilon_1 + 2^{-d}$. Thus $\mathbf{H}_\infty(W | \text{NMSim}_1 \notin \{\text{same}^*, (S, k_w, k_c)\}) \geq N - d$.

For the remainder of the proof, we assume $\text{NMSim}_1 \notin \{\text{same}^*, (S, k_w, k_c)\}$ (or, when appropriate, $(L, W, \sigma_w, R, \tilde{C}, \tilde{\sigma}_c) \in \mathcal{P}_3$) but don't state them explicitly.

Let $\text{Vrfy}_\ell^{\text{Sim}} := \text{Vrfy}_{\underline{K}_w^{\text{Sim}}}(f_2(L^{\text{Sim}}, W, \sigma_w), f_3(L^{\text{Sim}}, W, \sigma_w))$, where $\sigma_w = \text{Tag}_{k_w}(W)$. By Lemma 2.1, we have that

$$\tilde{\mathbf{H}}_\infty \left(W | \underline{S}^{\text{Sim}}, \underline{K}_w^{\text{Sim}}, \underline{K}_c^{\text{Sim}}, \text{Vrfy}_\ell^{\text{Sim}}, f_1(L^{\text{Sim}}, W, \sigma_w), \text{Ext}(f_2(L^{\text{Sim}}, W, \sigma_w); \underline{S}^{\text{Sim}}) \right)$$

is at least $N - d - \kappa^* - \kappa - n - 1 \geq \kappa^* + 4n$. Thus, by the strong extractor property of Ext ,

$$\text{Ext}(W; S) \approx_{\varepsilon_2} U | S, L^{\text{Sim}}, \tilde{C}, f_1(L^{\text{Sim}}, W, \sigma_w), \text{Ext}(f_2(L^{\text{Sim}}, W, \sigma_w); \underline{S}^{\text{Sim}}), \underline{S}^{\text{Sim}}, \underline{K}_w^{\text{Sim}}, \underline{K}_c^{\text{Sim}}, \text{Vrfy}_\ell^{\text{Sim}} \quad (3.16)$$

In order to show that the tampered codeword is independent of the message, we first need a statement similar to the inequality 3.16, but where the simulated output is replaced by the decoding of the tampering of $(L, W, \sigma_w, R, \tilde{C}, \tilde{\sigma}_c)$. We will obtain this using (3.15) twice and applying the triangle inequality. For the remainder of the proof, let $L, W, \sigma_w, R, \tilde{C}, \tilde{\sigma}_c$ be distributed as $L, W, \sigma_w, R, \tilde{C}, \tilde{\sigma}_c |_{(L, W, \sigma_w, R, \tilde{C}, \tilde{\sigma}_c) \in \mathcal{P}_3}$. Also, let $\underline{K}_w, \underline{K}_c, \underline{S}, \underline{L}, \underline{R}, \underline{W}$ be the corresponding random variables after tampering conditioned on the event that $(L, W, \sigma_w, R, \tilde{C}, \tilde{\sigma}_c) \in \mathcal{P}_3$. We shorthand the distribution in the LHS and RHS of 3.16 by A and B , respectively. Also, we define $\text{Vrfy}_\ell := \text{Vrfy}_{\underline{K}_w}(f_2(L, W, \sigma_w), f_3(L, W, \sigma_w))$. Applying Lemma 2.2 on inequality 3.15, we get that

$$A \approx_{\frac{2\varepsilon_1}{\alpha}} \text{Ext}(W; S), S, L, \tilde{C}, \underline{L}, \text{Ext}(\underline{W}; \underline{S}), \underline{S}, \underline{K}_w, \underline{K}_c, \text{Vrfy}_\ell, \quad (3.17)$$

and

$$B \approx_{\frac{2\varepsilon_1}{\alpha}} U, S, L, \tilde{C}, \underline{L}, \text{Ext}(\underline{W}; \underline{S}), \underline{S}, \underline{K}_w, \underline{K}_c, \text{Vrfy}_\ell. \quad (3.18)$$

By triangle inequality applied on the inequalities (3.17), (3.18), and (3.16), we have that

$$\text{Ext}(W; S) \approx_{\varepsilon_2 + \frac{4\varepsilon_1}{\alpha}} U | S, L, \tilde{C}, \underline{L}, \text{Ext}(\underline{W}; \underline{S}), \underline{S}, \underline{K}_w, \underline{K}_c, \text{Vrfy}_\ell.$$

Notice that given $(S, L, \tilde{C}, \underline{S}, \underline{K}_w, \underline{K}_c, \underline{L})$, the random variable R is independent of W , and hence

we obtain that

$$\text{Ext}(W; S) \approx_{\varepsilon_2 + \frac{4\varepsilon_1}{\alpha}} U|S, L, \tilde{C}, \underline{L}, \text{Ext}(\underline{W}; \underline{S}), \underline{S}, \underline{K_w}, \underline{K_c}, \text{Vrfy}_\ell, R, g_1(R, \tilde{C}, \tilde{\sigma}_c) .$$

Using Lemma 2.2, we can drop $\tilde{C}$ on both sides to get

$$\text{Ext}(W; S) \approx_{\varepsilon_2 + \frac{4\varepsilon_1}{\alpha}} U|S, L, \underline{L}, \text{Ext}(\underline{W}; \underline{S}), \underline{S}, \underline{K_w}, \underline{K_c}, \text{Vrfy}_\ell, R, \underline{R} \quad (3.19)$$

where $\underline{R} \stackrel{\text{def}}{=} g_1(R, \tilde{C}, \tilde{\sigma}_c)$.

We now observe that W is independent of $\tilde{C}$ given $S, L, \underline{L}, \text{Ext}(\underline{W}; \underline{S}), \underline{S}, \underline{K_w}, \underline{K_c}, \text{Vrfy}_\ell, R, \underline{R}$. To see this, we will use Lemma 2.9 with $V_0 = L, R, S, k_w, k_c$, and then $V_1 = \underline{L}$, $V_2 = \underline{R}$, $V_3 = \text{Ext}(\underline{W}; \underline{S}), \text{Vrfy}_\ell$, and observing that $\underline{S}, \underline{K_w}, \underline{K_c}$ is a deterministic function of $\underline{L}$ and $\underline{R}$. Thus, $\text{Ext}(W; S)$ is independent of $\tilde{C}$ given $S, L, \underline{L}, \text{Ext}(\underline{W}; \underline{S}), \underline{S}, \underline{K_w}, \underline{K_c}, \text{Vrfy}_\ell, R, \underline{R}$.

Now, applying Lemma 2.8 to equation 3.19, with $X = \text{Ext}(W, S)$, $Y = \tilde{C}$, $Z = (S, L, \underline{L}, \text{Ext}(\underline{W}; \underline{S}), \underline{S}, \underline{K_w}, \underline{K_c}, \text{Vrfy}_\ell, R, \underline{R})$, and using the above observation that $Y = \tilde{C}$ is independent of $\text{Ext}(W; S)$ given Z and of U given Z , we get that conditioned on $\tilde{C} \oplus \text{Ext}(W; S) = m$ on the LHS, and on $\tilde{C} \oplus U = m$ on the RHS:

$$C, S, L, \underline{L}, \text{Ext}(\underline{W}; \underline{S}), \underline{S}, \underline{K_w}, \underline{K_c}, \text{Vrfy}_\ell, R, \underline{R} \approx_{\varepsilon_2 + \frac{4\varepsilon_1}{\alpha}} \tilde{C}, S, L, \underline{L}, \text{Ext}(\underline{W}; \underline{S}), \underline{S}, \underline{K_w}, \underline{K_c}, \text{Vrfy}_\ell, R, \underline{R} ,$$

since $\tilde{C}$ conditioned on $\tilde{C} \oplus U = m$ is distributed identically as $\tilde{C}$ given all other random variables on the RHS in the above inequality.

The desired result follows by observing that the tampered codeword is a function of

$$\underline{L}, \underline{R}, \text{Ext}(\underline{W}; \underline{S}), C, R, \text{Vrfy}_\ell .$$

□

Proof: [Proof of Theorem 3.1] The simulator $\text{Sim}_{f,g}$ does the following. It samples $W, S, K_w, K_c, \sigma_w, L, R, \tilde{C}, \tilde{\sigma}_c$. Let $I^{\text{sim}} \in \{1, 2, 3\}$ be a random variable indicating the partition $(\mathcal{P}_{I^{\text{sim}}})$ in which the sampled codeword belongs. If $(L, W, \sigma_w, R, \tilde{C}, \tilde{\sigma}_c) \in \mathcal{P}_1$, then the simulator outputs $(\text{same}^*, R, \tilde{C}, \tilde{\sigma}_c)$ and sets $I^{\text{sim}} = 1$, if $(L, W, R, \tilde{C}) \in \mathcal{P}_2$ then the simulator outputs $(\perp, R, \tilde{C}, \tilde{\sigma}_c)$ and sets $I^{\text{sim}} = 2$, else the simulator outputs $(\text{Dec}((L, W, \sigma_w), (R, \tilde{C}, \tilde{\sigma}_c)), R, \tilde{C}, \tilde{\sigma}_c)$ and sets $I^{\text{sim}} = 3$.

Now, if $\Delta((\text{Tamper}_{f,g}^m, R, C, \sigma_c)|_{(L, W, \sigma_w, R, C, \sigma_c) \in \mathcal{P}_i} ; \text{Copy}(m, \text{Sim}_{f,g})|_{I^{\text{sim}}=i}) \leq \gamma_i$, by Lemma 2.6,

we will get:

$$\Delta((\text{Tamper}_{f,g}^m, R, C, \sigma_c); D_{f,g}^m) \leq \sum_{i=1}^3 \gamma_i \Pr[(L, W, \sigma_w, R, C, \sigma_c) \in \mathcal{P}_i], \quad (3.20)$$

where $D_{f,g}^m$ is a distribution on $(\{0, 1\}^{\kappa^*} \cup \{\text{same}^*, \perp\}) \times \{0, 1\}^{n+\kappa^*+t}$, such that $\Pr[D_{f,g}^m = d] = \sum_{i=1}^3 \Pr[(L, W, \sigma_w, R, C, \sigma_c) \in \mathcal{P}_i] \cdot \Pr[\text{Copy}(m, \text{Sim}_{f,g})|_{I^{\text{sim}=i}} = d]$. But, by Lemma 3.1, we get $(C, \sigma_c) \approx_{4\varepsilon_1 + 2^{-\Omega(\kappa/\log \kappa)}} (\tilde{C}, \tilde{\sigma}_c)|(L, R, \underline{L}, b_1)$, where b_1 is the bit indicating if $(\underline{W}, \underline{\sigma}_w) = (W, \sigma_w)$ or not, and the event that $(L, W, \sigma_w, R, C, \sigma_c) \in \mathcal{P}_i$ (or correspondingly $(L, W, \sigma_w, R, \tilde{C}, \tilde{\sigma}_c) \in \mathcal{P}_i$) can be determined by the random variables on the LHS and RHS of this inequality. Hence, together with the fact that $\Pr[\text{Copy}(m, \text{Sim}_{f,g}) = d] = \sum_{i=1}^3 \Pr[(L, W, \sigma_w, R, \tilde{C}, \tilde{\sigma}_c) \in \mathcal{P}_i] \cdot \Pr[\text{Copy}(m, \text{Sim}_{f,g})|_{I^{\text{sim}=i}} = d]$, for each $d \in (\{0, 1\}^{\kappa^*} \cup \{\text{same}^*, \perp\}) \times \{0, 1\}^{n+\kappa^*+t}$, we get:

$$\Delta(D_{f,g}^m; \text{Copy}(m, \text{Sim}_{f,g})) \leq 4\varepsilon_1 + 2^{-\Omega(\kappa/\log \kappa)} \quad (3.21)$$

Combining inequality 3.20 and 3.21, by triangle inequality, we get:

$$\Delta((\text{Tamper}_{f,g}^m, R, C, \sigma_c); \text{Copy}(m, \text{Sim}_{f,g})) \leq 4\varepsilon_1 + 2^{-\Omega(\kappa/\log \kappa)} + \sum_{i=1}^3 \gamma_i \Pr[(L, W, \sigma_w, R, C, \sigma_c) \in \mathcal{P}_i], \quad (3.22)$$

Hence, now we consider each partition and bound the corresponding term in the RHS of the summation in the inequality 3.22.

First, since we know that $(\text{Tamper}_{f,g}^m, R, C, \sigma_c)|_{(L, W, \sigma_w, R, C, \sigma_c) \in \mathcal{P}_1} = (m, R, C, \sigma_c)$ and $\text{Copy}(m, \text{Sim}_{f,g})|_{I^{\text{sim}=1}} = (m, R, \tilde{C}, \tilde{\sigma}_c)$, we use Lemma 3.1 to get:

$$\begin{aligned} \Delta((\text{Tamper}_{f,g}^m, R, C, \sigma_c)|_{(L, W, \sigma_w, R, C, \sigma_c) \in \mathcal{P}_1}; \text{Copy}(m, \text{Sim}_{f,g})|_{I^{\text{sim}=1}}) &\leq \Delta((C, \sigma_c); (\tilde{C}, \tilde{\sigma}_c)|(L, R, \underline{L}, b_1)) \\ &\leq 4\varepsilon_1 + 2^{-\Omega(\kappa/\log \kappa)} \end{aligned} \quad (3.23)$$

Hence, $\gamma_1 = 4\varepsilon_1 + 2^{-\Omega(\kappa/\log \kappa)}$, on the RHS of inequality 3.22. Similarly, since we know that by Lemma 3.2, $\Pr[(\text{Tamper}_{f,g}^m \neq \perp \wedge (L, W, \sigma_w, R, C, \sigma_c) \in \mathcal{P}_2)] \leq 4\varepsilon_1 + 2^{-\Omega(\kappa)}$, and $\text{Copy}(m, \text{Sim}_{f,g})|_{I^{\text{sim}=2}} = (\perp, R, \tilde{C}, \tilde{\sigma}_c)$, we use Lemma 3.1 to get:

$$\begin{aligned} &\Pr[(L, W, \sigma_w, R, C, \sigma_c) \in \mathcal{P}_2] \cdot \Delta((\text{Tamper}_{f,g}^m, R, C, \sigma_c)|_{(L, W, \sigma_w, R, C, \sigma_c) \in \mathcal{P}_2}; \text{Copy}(m, \text{Sim}_{f,g})|_{I^{\text{sim}=2}}) \\ &\leq \Pr[(\text{Tamper}_{f,g}^m \neq \perp \wedge (L, W, \sigma_w, R, C, \sigma_c) \in \mathcal{P}_2)] + \Delta((C, \sigma_c); (\tilde{C}, \tilde{\sigma}_c)|(L, R, \underline{L}, b_1)) \\ &\leq 8\varepsilon_1 + 2^{-\Omega(\kappa/\log \kappa)} \end{aligned} \quad (3.24)$$

Hence, $\gamma_2 \cdot \Pr[(L, W, \sigma_w, R, C, \sigma_c) \in \mathcal{P}_2] \leq 8\varepsilon_1 + 2^{-\Omega(\kappa/\log \kappa)}$, on the RHS of inequality 3.22.

Now, for the third partition, we set $\alpha = \sqrt{\varepsilon} + 2^{-d/2} (> 2\varepsilon + 2^{-d})$ in Lemma 3.3, and get that for any fixing a, b of K_w, K_c , if $\Pr[(L, W, \sigma_w, R, \tilde{C}, \tilde{\sigma}_c) \in \mathcal{P}_3 | K_w = a, K_c = b] \geq \alpha = \sqrt{\varepsilon} + 2^{-d/2}$, then

$$\begin{aligned} & \Delta((\text{Tamper}_{f,g}^m, R, C, \sigma_c) |_{(L, W, \sigma_w, R, C, \sigma_c) \in \mathcal{P}_3, K_w=a, K_c=b} ; \text{Copy}(m, \text{Sim}_{f,g}) |_{I^{\text{sim}}=3, K_w=a, K_c=b}) \\ & \leq 2^{-\Omega(\kappa/\log \kappa)} + \frac{4\varepsilon_1}{\alpha} \\ & \leq 2^{-\Omega(\kappa/\log \kappa)} + 4\sqrt{\varepsilon_1} \end{aligned}$$

Now, since $d = \Omega(\kappa/\log \kappa)$, and once again using Lemma 3.1, we get:

$$\begin{aligned} & \Pr[(L, W, \sigma_w, R, C, \sigma_c) \in \mathcal{P}_3] \cdot \Delta((\text{Tamper}_{f,g}^m, R, C, \sigma_c) |_{(L, W, \sigma_w, R, C, \sigma_c) \in \mathcal{P}_3} ; \text{Copy}(m, \text{Sim}_{f,g}) |_{I^{\text{sim}}=3}) \\ & \leq 4\varepsilon_1 + 2^{-\Omega(\kappa/\log \kappa)} + \Pr[I^{\text{sim}} = 3] \sum_{a,b} \Pr[K_w = a, K_c = b | I^{\text{sim}} = 3] \\ & \quad \cdot \Delta((\text{Tamper}_{f,g}^m, R, C, \sigma_c) |_{I^{\text{sim}}=3, K_w=a, K_c=b} ; \text{Copy}(m, \text{Sim}_{f,g}) |_{I^{\text{sim}}=3, K_w=a, K_c=b}) \\ & \leq (4\varepsilon_1 + 2^{-\Omega(\kappa/\log \kappa)}) + \sum_{a,b} \Pr[K_w = a, K_c = b] \cdot \Pr[I^{\text{sim}} = 3 | K_w = a, K_c = b] \\ & \quad \cdot \Delta((\text{Tamper}_{f,g}^m, R, C, \sigma_c) |_{I^{\text{sim}}=3, K_w=a, K_c=b} ; \text{Copy}(m, \text{Sim}_{f,g}) |_{I^{\text{sim}}=3, K_w=a, K_c=b}) \\ & \leq (4\varepsilon + 2^{-\Omega(\kappa/\log \kappa)}) + \sum_{a,b} \Pr[K_w = a, K_c = b] \cdot O(\sqrt{\varepsilon_1} + 2^{-\Omega(\kappa/\log \kappa)}) \\ & \leq O(\sqrt{\varepsilon_1} + 2^{-\Omega(\kappa/\log \kappa)}) \end{aligned} \tag{3.25}$$

In the above inequality 3.25, we make an abuse of notation, and use $(\text{Tamper}_{f,g}^m, R, C, \sigma_c) |_{I^{\text{sim}}=3, K_w=a, K_c=b}$ to denote the distribution where the partition is first picked using I^{sim} , and then the distribution $(\text{Tamper}_{f,g}^m, R, C, \sigma_c)$ is drawn conditioned on the partition. This distribution is clearly identical to $(\text{Tamper}_{f,g}^m, R, C, \sigma_c) |_{(L, W, \sigma_w, R, C, \sigma_c) \in \mathcal{P}_3}$, which is what we use above.

Hence, $\gamma_3 = O(\sqrt{\varepsilon_1} + 2^{-\Omega(\kappa/\log \kappa)})$, on the RHS of inequality 3.22. Finally, we can use inequalities 3.23, 3.24 and 3.25, in inequality 3.22 to get:

$$\Delta((\text{Tamper}_{f,g}^m, R, C, \sigma_c); \text{Copy}(m, \text{Sim}_{f,g})) \leq O(\sqrt{\varepsilon_1} + 2^{-\Omega(\kappa/\log \kappa)})$$

Rate. Clearly from the parameters set in Section 3.2.1, the size of our codeword is $|\ell| + |w| + |\sigma_w| + |r| + |c| + |\sigma_c| = n + N + t + n + \kappa^* + t = (3 + o(1))\kappa^*$, since we take $\kappa^* \in \omega(n)$, thus giving us the desired rate of $1/(3 + o(1))$. $\square$

3.3 t -Source Strong Non-malleable Extractors to t -State Augmented NMCs

In this section, we begin by generalizing the definition of augmented NMCs to the t -split state setting. Then, we generalize the connection known between seedless non-malleable extractors for t independent sources (Definition 2.2) and non-malleable codes for the t -split-state family from [40], to establish a connection between strong seedless non-malleable extractors for t independent sources and augmented non-malleable codes for t -split-state family. As a corollary, this helps us prove that the 2-split-state NMC of [81], which uses the [40]-compiler from non-malleable extractors to non-malleable codes, is indeed augmented secure.

We generalize the definition of 2-split-state augmented-NMC (Definition 2.6), to a j -augmented NMC for the t -split-state family, i.e., j of the t -states are also simulatable independent of the message (where $j < t$).

Definition 3.1 *A coding scheme (Enc, Dec) with message and codeword spaces as $\{0, 1\}^\alpha, (\{0, 1\}^\beta)^t$ respectively, is $[\varepsilon, j]$ - augmented non-malleable (where $j < t$) with respect to the function family $\mathcal{F} = \{(f_1, \dots, f_t) : f_i : \{0, 1\}^\beta \rightarrow \{0, 1\}^\beta\}$ if $\forall (f_1, \dots, f_t) \in \mathcal{F}, \exists$ a distribution $\text{Sim}_{f_1, \dots, f_t}$ over $(\{0, 1\}^\beta)^j \times (\{0, 1\}^\alpha \cup \{\text{same}^*, \perp\})$ such that $\forall m \in \{0, 1\}^\alpha$*

$$\text{Tamper}_{f_1, \dots, f_t}^m \approx_\varepsilon \text{Copy}_{\text{Sim}_{f_1, \dots, f_t}}^m$$

where $\text{Tamper}_{f,g}^m$ denotes the distribution $(X_{i_1}, \dots, X_{i_j}, \text{Dec}(f_1(X_1), \dots, f_t(X_t)))$, where $\text{Enc}(m) = (X_1, \dots, X_t)$ and $(X_{i_1}, \dots, X_{i_j})$ represents some j states of the total t states. $\text{Copy}_{\text{Sim}_{f_1, \dots, f_t}}^m$ is defined as

$$\begin{aligned} (X_{i_1}, \dots, X_{i_j}, \tilde{m}) &\leftarrow \text{Sim}_{f_1, \dots, f_t} \\ \text{Copy}_{\text{Sim}_{f_1, \dots, f_t}}^m &= \begin{cases} (X_{i_1}, \dots, X_{i_j}, m) & \text{if } (X_{i_1}, \dots, X_{i_j}, \tilde{m}) = (X_{i_1}, \dots, X_{i_j}, \text{same}^*) \\ (X_{i_1}, \dots, X_{i_j}, \tilde{m}) & \text{otherwise} \end{cases} \end{aligned}$$

$\text{Sim}_{f_1, \dots, f_t}$ should be efficiently samplable given oracle access to $(f_1, \dots, f_t)(\cdot)$.

To generalize the connection of non-malleable extractors and non-malleable codes given in [40], we begin by first formulating an alternate definition (of Definition 2.2) of a t -source strong seedless non-malleable extractor. This definition is obtained by generalizing the definition of seedless (relaxed) non-malleable extractors in [40] to the setting with t sources, which captures the property that the output of non-malleable extractor on the modified sources along with one of the source, is simulatable independent of the output of non-malleable extractor on original

sources.

Definition 3.2 A function $\text{nmExt} : (\{0, 1\}^n)^t \rightarrow \{0, 1\}^\ell$ is a (k, ε) -seedless (relaxed) strong non-malleable extractor for t independent sources w.r.t. family $\mathcal{F} = \{(f_1, \dots, f_t) : f_i : \{0, 1\}^n \rightarrow \{0, 1\}^n\}$, such that $\exists$ at least one j such that f_j has no fixed point, if it satisfies the following property: Let $X_1, \dots, X_t$ be t independent (n, k) -sources and $(f_1, \dots, f_t) \in \mathcal{F}$, then the following hold:

- nmExt is a t -source extractor for $(X_1, \dots, X_t)$, i.e., $\text{nmExt}(X_1, \dots, X_t) \approx_\varepsilon U_\ell$.
- There exists a distribution $\mathcal{D}$ over $\{0, 1\}^n \times (\{0, 1\}^\ell \cup \{\text{same}^*\})$ s.t. for an independent $(X_1, Y) \sim \mathcal{D}$,

$$(\text{nmExt}(X_1, \dots, X_t), X_1, \text{nmExt}(f_1(X_1), \dots, f_t(X_t))) \approx_\varepsilon (\text{nmExt}(X_1, \dots, X_t), \text{copy}((X_1, Y), (X_1, \text{nmExt}(X_1, \dots, X_t))))$$

Remark 1 It is clear that the non-malleability condition in Def 2.2 (for $i = 1$) is sufficient for the conditions in Def 3.2 to be satisfied.

But then, this relaxed notion of strong seedless non-malleable extractor is equivalent to the following general notion of strong non-malleable extractor (where, the tampering functions can have fixed points) upto a slight loss of parameters. (This proof follows from [40, Lemma 5.6]).

Definition 3.3 A function $\text{nmExt} : (\{0, 1\}^n)^t \rightarrow \{0, 1\}^\ell$ is a (k, ε) -seedless strong non-malleable extractor for t independent sources w.r.t. family $\mathcal{F} = \{(f_1, \dots, f_t) : f_i : \{0, 1\}^n \rightarrow \{0, 1\}^n\}$, if it satisfies the following property: Let $X_1, \dots, X_t$ be t independent (n, k) -sources and $(f_1, \dots, f_t) \in \mathcal{F}$, then the following hold:

- nmExt is a t -source extractor for $(X_1, \dots, X_t)$, i.e., $\text{nmExt}(X_1, \dots, X_t) \approx_\varepsilon U_\ell$.
- There exists a distribution $\mathcal{D}$ over $\{0, 1\}^n \times (\{0, 1\}^\ell \cup \{\text{same}^*\})$ s.t. for an independent $(X_1, Y) \sim \mathcal{D}$,

$$(\text{nmExt}(X_1, \dots, X_t), X_1, \text{nmExt}(f_1(X_1), \dots, f_t(X_t))) \approx_\varepsilon (\text{nmExt}(X_1, \dots, X_t), \text{copy}((X_1, Y), (X_1, \text{nmExt}(X_1, \dots, X_t))))$$

Hence, we take the above Definition 3.3 for strong non-malleable extractors and prove the following theorem.

Theorem 3.2 Let $\text{nmExt} : (\{0, 1\}^n)^t \rightarrow \{0, 1\}^k$ be a (n, ε) -seedless strong non-malleable extractor for t independent sources (by Definition 3.3). Define a coding scheme (Enc, Dec) with message length k and block length tn as follows. The decoder Dec is defined by

$$\text{Dec}(x_1, \dots, x_t) = \text{nmExt}(x_1, \dots, x_t)$$

The encoder Enc is defined as:

$$\text{Enc}(m) := \begin{cases} x_1, \dots, x_t \xleftarrow{\$} \text{nmExt}^{-1}(m) \\ o/p : (x_1, \dots, x_t) \end{cases}$$

Then, (Enc, Dec) is a $[\varepsilon', 1]$ -augmented non-malleable code (Definition 3.1) with error $\varepsilon' = \varepsilon(2^k + 1)$ for the t -split state family and with rate $= \frac{k}{tn}$.

Proof: Let $m \in \{0, 1\}^k$ and $f = (f_1, \dots, f_t) \in \mathcal{F}$, the t -split-state family be arbitrary. Since $\text{Dec} = \text{nmExt}$ is a strong non-malleable extractor, by Def 3.3, $\exists$ a distribution $\mathcal{D}$ s.t. for $(X_1, Y) \sim \mathcal{D}_{f_1, \dots, f_t}$, we have:

$$\begin{aligned} \text{nmExt}(X_1, \dots, X_t), X_1, \text{nmExt}(f_1(X_1), \dots, f_t(X_t)) &\approx_\varepsilon \\ \text{nmExt}(X_1, \dots, X_t), \text{copy}((X_1, Y), (X_1, \text{nmExt}(X_1, \dots, X_t))) &\quad (3.26) \end{aligned}$$

Claim. $\text{Enc}(U_k)$ is ε -close to uniform.

Proof: By extractor security, we have:

$$\text{Dec}(U_{tn}) \approx_\varepsilon U_k$$

Further, as $\text{Enc}(\cdot)$ samples uniformly random element of $\text{nmExt}^{-1}(\cdot)$, it follows that

$$\text{Enc}(\text{Dec}(U_{tn})) = U_{tn}$$

Hence, we get $\text{Enc}(U_k) \approx_\varepsilon \text{Enc}(\text{Dec}(U_{tn})) = U_{tn}$. □

Thus, at cost of ε increase in error, we assume codeword is of uniform distribution.

Let $(X_1, Y) \sim \mathcal{D}_{f_1, \dots, f_t}$. Now by Eq 3.26, just by substitution, we get:

$$(M, X_1, \text{Dec}(f(\text{Enc}(M)))) \approx_\varepsilon (M, \text{copy}((X_1, Y), (X_1, M)))$$

Now, for the arbitrary m that we chose, we get:

$$(m, X_1, \text{Dec}(f(\text{Enc}(m)))) \approx_{\varepsilon 2^k} (m, \text{copy}((X_1, Y), (X_1, m)))$$

which proves the theorem. $\square$

Augmented-non-malleability of 2-split-state construction in [81]:

Corollary 3.3 *There are constants $0 < c_1, c_2 < 1$ such that for any $\beta \in \mathbb{N}$ and $2^{-\frac{c_2 \beta}{\log \beta}} \leq \varepsilon \leq c_1$, there exists an explicit augmented non-malleable code in the 2-split-state model with block length 2β , rate $\Omega(\frac{\log \log \log(1/\varepsilon)}{\log \log(1/\varepsilon)})$ and error ε .*

Proof: It can be easily observed that the seedless 2 source non-malleable extractor constructed in [81] satisfies: For any (f, g) in 2-split-state family, such that atleast one of f or g has no fixed point, we have:

$$\text{nmExt}(X, Y), X, \text{nmExt}(f(X), g(Y)) \approx_{\varepsilon} U_{\ell}, X, \text{nmExt}(f(X), g(Y))$$

which, by Remark 1, is sufficient to imply the conditions in Definition 3.2. Hence, by Theorem 3.2, it is proved that the 2-split-state construction given in [81] is actually a 2-split-state augmented-non-malleable code.

Further, the specific non-malleable extractor of [81] gives error and rate parameters for the augmented-non-malleable code, exactly as obtained in Theorem 2.2. $\square$

Part II

**Leakage Resilient Secret Sharing
Schemes**

Chapter 4

Near Optimal/Optimal Leakage Resilient Secret Sharing Schemes

In this chapter, we will study the second object of interest in our quest to tackle with the other form of side-channel attacks, called the leakage attacks, namely *Leakage Resilient Secret Sharing*.

4.1 Introduction

Recall that a leakage resilient secret sharing (LRSS) scheme allows a dealer to share a secret amongst N parties in such a way that any authorized subset of the parties can recover the secret from their shares, while an adversary that obtains shares of any unauthorized subset of parties along with bounded leakage from the other shares (captured by a leakage model/family) learns no information about the secret. Ever since their introduction [55], leakage resilient secret sharing schemes have been studied in various leakage models [49, 84, 20, 65, 14, 96, 10, 59, 27, 79, 82, 38, 28], as we discussed in Section 1.2. We recall the broad categories of the leakage models along with the best known rates in each domain, as discussed in the introduction (Section 1.2). Amongst all prior works, in the information theoretic setting, either the leakage model is strong (adaptive and joint, allowing leakage queries dependent on prior responses and with each query dependent on multiple shares) with a sub-optimal rate of $\omega(\text{message length})$ (best known rate is $\mathcal{O}(1/N)$, where N is number of parties, by [38]), or the leakage model is weak (non-adaptive and independent, allowing only a single non-adaptive query, with each leakage query on a single share only), but with a near-optimal rate of $1/3$ (by [96]). In this work, our primary goal is to build an LRSS scheme in the joint and adaptive model, while still getting a constant rate. Furthermore, we improve on the LRSS scheme of [96] in the local leakage model, to get an

optimal rate of 1.

As a stepping stone towards building our joint and adaptive LRSS, we introduce an interesting new primitive called, *adaptive extractors*, which unlike traditional randomness extractors, guarantee security even when an adversary obtains leakage on the source *after* observing the extractor output. We believe that adaptive extractors would be of independent interest too. Formally, we define the notion of *adaptive extractors* with respect to an *after the fact* leakage family $\mathcal{F}$, and we say that an extractor is an adaptive extractor with respect to a function family $\mathcal{F}$, if for each $f \in \mathcal{F}$, an adversary cannot (statistically) distinguish $(S, f(W, \text{Ext}(W; S)), \text{Ext}(W; S))$ from $(S, f(W, U), U)$. Our notion of adaptive extractors can be seen as a generalization of exposure-resilient extractors introduced by Zimand [99] (Zimand’s extractors allow the adversary to adaptively learn up to n^δ bits of the source, for some $\delta < 1$ bits; the adversary can determine which bits to query based on an arbitrary function of the extractor output.), and of the notion of adaptive non-malleable extractors introduced by Aggarwal *et al.* in [4] (where adaptive non-malleability particularly guarantees that the adversary cannot distinguish between $(S, \text{Ext}(W; g(S, \text{Ext}(W; S))), \text{Ext}(W; S))$ and $(S, \text{Ext}(W; g(S, U)), U)$). We then observe that every randomness extractor is also an adaptive extractor with respect to a leakage family depending arbitrarily on the source and the output, with some loss in parameters. We note that this observation is similar to how the authors in [4, Lemma 3.5] show that every non-malleable extractor is adaptive non-malleable, with some loss in parameters. We demonstrate that, in spite of the loss in parameters that the adaptivity incurs, we use such extractors to build our constant-rate LRSS scheme tolerating adaptive leakage. We now describe these contributions in greater detail.

Our Results. We formalize the notion of adaptive extractors, and show that every randomness extractor is adaptive, with respect to an arbitrary leakage on the source, dependent on the output, with some loss in parameters. Further, we use such an adaptive extractor to build the following LRSS scheme, where N is the number of parties, t is the threshold, and ℓ is the message length.

Informal Theorem 1 (Main Result): *We build an LRSS scheme, tolerating ψ adaptive queries, each dependent on X shares (with $\psi \cdot X \leq N - t + 1$) and the reveal of the remaining $t - 1$ shares, such that it achieves a rate of $(X^{\Theta(\psi X/t)})^{-1}$, while allowing $\Theta(\ell)$ bits of leakage per query, for threshold access structures. In particular, for a constant X and $N = \Theta(t)$, this gives the first constant-rate adaptive LRSS scheme for the threshold access structure. We can also generalize our constructions to get the first constant-rate adaptive LRSS for general access structures.*

Furthermore, in the non-adaptive and independent (local leakage model) setting, we improve

the near-optimal construction of [96] to get the following optimal rate LRSS for general access structures.

Informal Theorem 2. *Given any secret sharing scheme for general monotone access structure $\mathcal{A}$ with share size ℓ/R , where ℓ is the message length and $R \leq 1$, one can construct an LRSS for the same access structure $\mathcal{A}$, against the local leakage model allowing μ bits of leakage per share, with a share size of $\ell/R + \mu + o(\ell/R + \mu)$.*

Since our main goal is to bridge the gap between the best known rate vs the best known leakage model, we first focus on giving a detailed technical overview of the construction ideas for Informal Theorem 1, followed by the formal construction and security proof.

Finally, in Appendix 4.5, we discuss the Informal Theorem 2 on the rate-optimal non-adaptive independent LRSS, and elaborate on the construction ideas for the same.

4.1.1 Technical Overview

We begin by describing the adaptive and joint leakage model for our LRSS and then give a technical overview of our scheme. For simplicity, we provide our technical overview for threshold access structures (which we can extend to general access structures as well). Let t denote the threshold and N , the number of parties.

Leakage Model. We allow the adversary to obtain adaptive leakage on $N - (t - 1)$ shares and then reveal the full shares of the remaining $t - 1$ shares. Each adaptive query can be on a set of at most X shares (where X is some value between 1 and $t - 1$), and different queries must be on sets that are disjoint from the prior queries. For the purposes of this exposition, we make the following restriction to our model: we assume that the adversary makes adaptive queries but only on a single share each time, i.e., it doesn't make any leakage query on multiple shares.

Warm-up Construction. To motivate our construction, we consider the following modification¹⁸ of a construction due to Srinivasan and Vasudevan in [96, Section 3.2.1]. Take any t -out-of- N secret sharing scheme (MShare, MRec) and then do as follows:

- Sample shares $(m_1, \dots, m_N)$ of the message m using MShare.
- Choose an extractor seed s and split s into $(sd_1, \dots, sd_N)$ using a t -out-of- n secret sharing scheme.
- Now, for every m_i , choose an extractor source w_i uniformly and compute $y_i = m_i \oplus \text{Ext}(w_i; s)$.

¹⁸We note that the original construction of [96] only aimed to achieve non-adaptive security, and we consider a modification, with the aim to expand to adaptive security.

- Finally, output the final shares $\{sh_i\}$ as $\{(w_i, y_i, sd_i)\}$.

For now, consider a weak model, where the adversary obtains only non-adaptive and independent leakage from a total of (say) $t - 1$ shares, in addition to $t - 1$ full shares. The hope is to show that the $t - 1$ leakage queries are independent of the message shares m_i , following which the privacy of MShare can be used to get the $t - 1$ full shares. One might hope to show this independence of leakage from the m_i 's, using the security of the extractor as follows: Pick sd_i uniformly at random and independent of s ; then the leakage function on $\{sh_i\}$, can be answered as an auxiliary leakage query on the source w_i . Once s is revealed in the extractor security game, the reduction can pick the other sd_j values in a consistent manner. However, this proof strategy has a flaw. For extractor security, it is important that the auxiliary leakage query on w is independent of s ; however, there is a dependence on s via y_i . In other words, it is unclear how to prove that this construction satisfies leakage resilience even in a weak model where the adversary obtains leakage only independently and non-adaptively.

Fortunately, with adaptive extractors, we can show that this construction is secure not only in this weak model but also in a stronger model where the adversary is allowed to leak from $t - 1$ shares adaptively, before receiving $t - 1$ full shares. Furthermore, this construction even has a constant rate! The high-level idea of security is as follows. We wish to reduce the adaptive leakage queries on the shares to an adaptive extractor leakage query. Since the adaptive leakage query on w_j cannot depend on the seed, we need to first show that the share sd_j in the corresponding query is independent of the seed s . Indeed, using the privacy of secret sharing¹⁹, we can show that for the first $t - 1$ queries, the shares sd_j in sh_j can be replaced with shares of 0 (hence removing the dependence on s). Then, using the adaptive extractor security, we can replace the y_j 's (for the first $t - 1$ queries) with uniform, where the leakage can be asked on the w_j 's. Now, the privacy of MShare can be invoked to get the $t - 1$ full shares.

Main Construction. Our next goal is to leverage adaptive extractors to go beyond leaking from just $t - 1$ shares. The main bottleneck is that for any subsequent leakage query (beyond $t - 1$), the sd_j 's will reveal s , and hence the adaptive leakage query on subsequent w_j 's will no longer remain independent of the seed s . Thus, extractor security fails. This is the challenge we must address to achieve our main result where the adversary is allowed to obtain adaptive leakage on $N - (t - 1)$ shares (in total) and reveal $t - 1$ of the remaining shares.

One approach to facilitating leakage from more than $t - 1$ shares could be to use independent extractor seeds to extract independent random masks. Consider the following modification of the above construction: mask the share of a message m_i not just with one extractor output

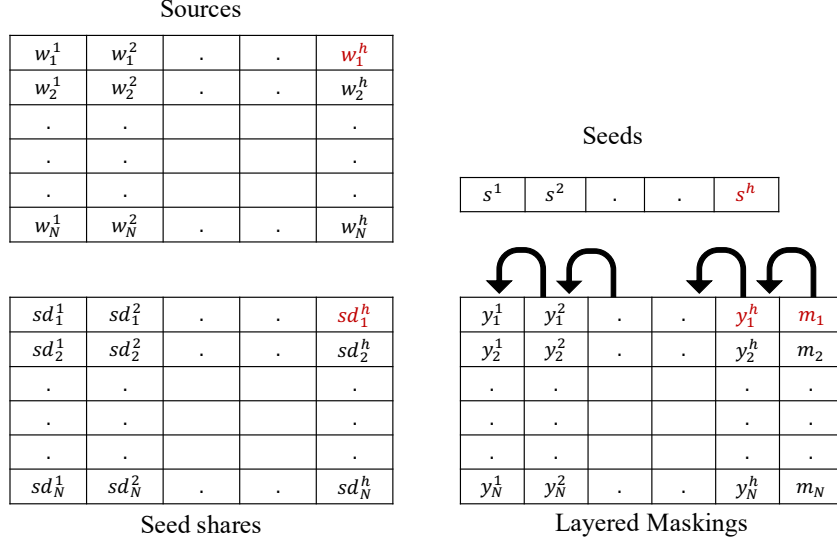
¹⁹Since the leakage queries are adaptive, we require adaptive privacy of the underlying secret sharing scheme, and we show instantiations of the same.

but with many. In particular, let $y_i = m_i \oplus \text{Ext}(w_i; s^1) \oplus \text{Ext}(w_i; s^2) \dots \oplus \text{Ext}(w_i; s^h)$, for some parameter h , where $s^1 \dots s^h$ are independent seeds. We might hope that because we are using h seeds, we could hope to leak from $h(t-1)$ shares and use the security of each seed per batch of $t-1$ shares. Unfortunately, this doesn't work for the following reason: reconstruction is only possible if we recover all h seeds. This means that we ultimately need to somehow share all the seeds in a manner where they can be reconstructed from $t-1$ shares. In other words, once we leak from $t-1$ shares, we can no longer argue security by leveraging any of the seeds because they can all be reconstructed from $t-1$ shares. We overcome this challenge by carefully using a multi-layered approach for both masking the message shares as well as for reconstructing the seeds.

Construction Overview:

1. Pick h extractor seeds $s^1, \dots, s^h$ and hN extractor sources $w_1^1, \dots, w_1^h, \dots, w_N^1, \dots, w_N^h$.
2. Secret share each of the h seeds using a t -out-of- N secret sharing scheme to obtain shares; let the share of s^j be $sd_1^j, \dots, sd_N^j$.
3. Each share m_j is masked using the h seeds in a layered manner as follows:
 - (a) In level $h+1$: Set $y_j^{h+1} = m_j$.
 - (b) For every subsequent lower level $i (i \geq 1)$, compute $x_j^i = y_j^{i+1} \oplus \text{Ext}^i(w_j^i; s_i)$ and set $y_j^i = (x_j^i || sd_j^i)$. [Note that we use a different extractor per-level since the length of the extractor outputs (and the length of y_j^i s they mask) increase with level.] Finally set $Sh_j = (w_j^1, \dots, w_j^h, y_j^1)$.
4. Output $(Sh_1, \dots, Sh_N)$

A pictorial representation of the construction can be found in Figure 4.1. In order to give an overview of the proof, we first recall that we are in a setting where each adaptive query of an adversary is a query on a single share – we can extend our results to the case of joint leakage but, for the sake of simplicity, we don't focus on that for now.



Each entry of the layered maskings matrix appropriately uses the corresponding entries of the sources, seeds and seed shares matrices. In addition, each entry y_i^j ($j \leq h$) also depends on the subsequent value i.e., y_i^{j+1} .
Example: $y_1^h = m_1 \oplus \text{Ext}^h(w_1^h; s^h) || sd_1^h$ (colored red)

Figure 4.1: The Main Construction

At a high-level, the idea of the security proof is that we view the leakage queries in batches of $t - 1$ queries. For the first set of $t - 1$ queries, we rely on the adaptive security of the extractor outputs evaluated using seed s^1 and, in particular, all of these outputs can be replaced by uniform. (This also relies on the *adaptive* privacy of the secret sharing scheme, a notion we define and instantiate.) For the second set of $t - 1$ queries, we can no longer assume that s^1 is hidden, since we can not use the privacy of the secret sharing scheme any more. However, two things come to our rescue: first, the second batch of queries helps unmask at most $t - 1$ shares of s^2 and therefore, adaptive extractor security on seed s^2 can be leveraged; second, the extractor outputs $\text{Ext}(w_j^1; s^1)$ (where j was a share that was leaked from in the first batch) continue to remain uniform. The reason for the latter is that all extractor sources are uniformly chosen, and our model requires a disjoint set of indices to be leaked from across batches. In short, for the first batch of queries, we use adaptive security of the extractor outputs evaluated on the first seed and, for every subsequent batch, we move to argue extractor security using the subsequent seed. Since we have h independent seeds, we can do this h times and therefore answer h batches of queries, i.e., we can obtain leakage on $h(t - 1)$ shares.

4.1.2 Organization of the Chapter

We begin by defining adaptive extractors, and giving an instantiation of the same, in Section 4.2. Then, we describe our adaptive and joint leakage model in Section 4.3. Finally, we give our

construction, prove its security and give a detailed rate analysis in Section 4.4. In Appendix 4.5, we give an overview of our rate-optimal LRSS scheme in the non-adaptive and independent leakage model.

4.2 Adaptive Extractors

The standard seeded randomness extractors (Definition 2.1), unfortunately, provide no guarantees on the extractor output being uniform when an adversary can obtain an ‘adaptive’ leakage on the source, that is *dependent on the extractor output* and the seed. This is not surprising, as if an adversary can obtain *arbitrary adaptive leakage* on the source, then we cannot hope for the extractor output to remain uniform. For example, given $y = \text{Ext}(w, s)$, an adversary can distinguish the extractor output from uniform with high probability by querying a single bit of auxiliary information that tells her whether $\text{Ext}(w, s) = y$. However, as we will see later, in many applications, the adaptive leakage that the adversary obtains comes from a specific function family. Hence, by carefully defining this function family, we show how to obtain useful notions of extractors that guarantee security even in the presence of an adaptive auxiliary information. We introduce and call this notion *adaptive extractors* and now proceed to formally define them.

Definition 4.1 *An $(n, \tau, d, \ell, \varepsilon)$ - extractor Ext is said to be an $(\mathcal{F}, \delta)$ -**adaptive extractor** if for all pairs of random variables (W, Z) such that W is an n -bit string satisfying $\tilde{\mathbf{H}}_\infty(W|Z) \geq \tau$, and any function f in the function family $\mathcal{F}$, it holds that*

$$Z, U_d, f(W, \text{Ext}(W; U_d), U_d), \text{Ext}(W; U_d) \approx_\delta Z, U_d, f(W, U_\ell, U_d), U_\ell$$

*We call δ , the **adaptive error** of the extractor.*

4.2.1 Construction

Generic Relation. We show that every extractor is in fact an adaptive extractor for the family of leakage functions where the adaptive leakage depends only on the source and the extractor output (i.e., it doesn’t depend on the seed except via the extractor output), with some loss in security. This loss, in fact, depends only on the number of bits of the extractor output that the adaptive leakage function depends on. For ease of exposition, we omit auxiliary information z that depends only on the source (but not on the extractor output or seed) from the notation below. We now explicitly define this family below:

$$\mathcal{F}_{a, \zeta} \subseteq \{f' : \{0, 1\}^n \times \{0, 1\}^\ell \rightarrow \{0, 1\}^\zeta\}$$

such that for every $f' \in \mathcal{F}_{a,\zeta}$ there exists two functions $f : \{0,1\}^\ell \rightarrow \{0,1\}^a$ and

$$g : \{0,1\}^{n+a} \rightarrow \{0,1\}^\zeta \text{ such that } \forall w, y, f'(w, y) = g(w, f(y))\}$$

Here, ‘ ζ ’ denotes the number of bits of adaptive leakage and ‘ a ’ denotes the number of bits of the extractor output (or the uniform string) that the adaptive leakage depends on. This is captured by requiring that every function f' has an equivalent representation in terms of some g and f such that $f'(w, y) = g(w, f(y))$ where f ’s output is only a bits long. w and y should be interpreted as the source and the extractor output (or the uniform string) respectively.

The following theorem shows that any $(n, \tau, d, \ell, \varepsilon)$ -seeded randomness extractor can be shown to be adaptive secure against the above family $\mathcal{F}_{a,\zeta}$, with an adaptive error of $2^{a+2}\varepsilon$. Informally, we can reduce the adaptive security to the extractor security (as in Definition 2.1) in the following way: to answer the adaptive leakage query, the reduction makes a guess, v , for the extractor challenge dependent value $f(y_b)$ (where, y_b is the extractor challenge), which is of a -bits, and gets the leakage $g(w, v)$ from the source. Now, it gets the challenge y_b from the extractor challenger and if $f(y_b)$ matches the guess v , then the reduction can successfully simulate the challenge and the adaptive leakage response, else it cannot proceed (and aborts). Hence, the winning probability in the extractor game is the probability of a correct guess (2^{-a}), multiplied with the winning probability of the adaptive extractor adversary. We formalize this proof in the theorem below.

Theorem 4.1 *Every $(n, \tau, d, \ell, \varepsilon)$ -average case extractor Ext , as in Definition 2.1, is an $(n, \tau + \zeta, d, \ell, \varepsilon)$ - extractor that is $(\mathcal{F}_{a,\zeta}, 2^{a+2}\varepsilon)$ -adaptive, for any $\tau + \zeta \leq n$ and $a \leq \ell$.*

Proof: For simplicity, we omit the auxiliary information Z , that depends only on the source (and not on the extractor output). Let W be the source of n bits, such that $\mathbf{H}_\infty(W) \geq \tau + \zeta$. Consider $f' \in \mathcal{F}_{a,\zeta}$, with the corresponding functions (f, g) (recall $f'(w, y) = g(w, f(y))$, where f outputs a bits and g outputs ζ bits). To prove adaptive security (definition 4.1), we need to show that:

$$U_d, f'(W, Y), Y \approx_{2^{a+2}\varepsilon} U_d, f'(W, U_\ell), U_\ell,$$

where Y is the random variable $\text{Ext}(W; U_d)$. Expanding the description of f' , this gives:

$$U_d, g(W, f(Y)), Y \approx_{2^{a+2}\varepsilon} U_d, g(W, f(U_\ell)), U_\ell$$

To prove this, we consider the following two sets $\mathcal{B} = \{b : \Pr[f(Y) = b] > 0\}$ and $\mathcal{A} = \{0, 1\}^{d+\zeta+\ell}$. For each $b \in \mathcal{B}$, we begin by using the statistical distance Lemma 2.3 with random variables A, B and events E, E' set as $(U_d, g(W, f(Y)), Y)$, $(U_d, g(W, f(U_\ell)), U_\ell)$, $f(Y) = b$ and

$f(U_\ell) = b$, respectively. By use of law of total probability and Lemma 2.3, we get:

$$\begin{aligned} & \Delta((U_d, g(W, f(Y)), Y); (U_d, g(W, f(U_\ell)), U_\ell)) \\ & \leq \Pr[f(U_\ell) \notin \mathcal{B}] + \sum_{b \in \mathcal{B}} \Delta(A \wedge E; B \wedge E') \\ & \leq \Pr[f(U_\ell) \notin \mathcal{B}] + \sum_{b \in \mathcal{B}} ((|\Pr[E] - \Pr[E']|) + \Pr[E'] \cdot \Delta(A|E; B|E')) \end{aligned}$$

But now, note that, by extractor security, since $Y \approx_\varepsilon U_\ell$, by applying Lemma 2.2, we have $f(Y) \approx_\varepsilon f(U_\ell)$. Further, by the definition of statistical distance, we have that, for each $b \in \mathcal{B}$, $|\Pr[f(Y) = b] - \Pr[f(U_\ell) = b]| \leq \varepsilon$ and $\Pr[f(U_\ell) \notin \mathcal{B}] \leq \varepsilon$ (since $\Pr[f(Y) \notin \mathcal{B}] = 0$). Applying this to above inequality, we get:

$$\begin{aligned} & \Delta((U_d, g(W, f(Y)), Y); (U_d, g(W, f(U_\ell)), U_\ell)) \\ & \leq \varepsilon + \sum_{b \in \mathcal{B}} (\varepsilon + \Pr[E'] \cdot \Delta(A|E; B|E')) \\ & = (|\mathcal{B}| + 1)\varepsilon + \sum_{b \in \mathcal{B}} \Pr[E'] \cdot \Delta(A|E; B|E') \end{aligned}$$

Finally, we apply the statistical distance lemma 2.5 on the random variables $(A, f(Y))$ and $(B, f(U_\ell))$ with set $S = \{b\}$. Note that, given events E and E' the value of $f(Y)$ and $f(U_\ell)$ are fixed to a b , which means the leakage $g(W, b)$ is only a leakage on W . Thus, we can use extractor security to get: $(U_d, g(W, b), Y) \approx_\varepsilon (U_d, g(W, b), U_\ell)$. Hence, applying this to the above inequality, we get:

$$\begin{aligned} & \Delta((U_d, g(W, f(Y)), Y); (U_d, g(W, f(U_\ell)), U_\ell)) \\ & \leq (|\mathcal{B}| + 1)\varepsilon + \sum_{b \in \mathcal{B}} \Pr[E'] \cdot \frac{2\varepsilon}{\Pr[f(U_\ell) = b]} \\ & \leq 4|\mathcal{B}|\varepsilon \leq 2^{a+2}\varepsilon \end{aligned}$$

□

Concrete Instantiation. We show that the seeded randomness extractor of [70] from Lemma 2.10 is an adaptive extractor even when the leakage depends on the entire extractor output.

Let $\text{Full}_\zeta (= \mathcal{F}_{\ell, \zeta})$, denote the leakage function family which computes leakage (of size ζ) dependent on the entire extractor output and the source. The following lemma shows that one can appropriately set the parameters of the extractor from [70] to get negligible error,

AdLeak^b :

- Let Tr and $\mathcal{S}$ be a null string and null set respectively.
- For upto k times
 - $(j, g_j) \leftarrow \mathcal{D}'(Tr)$ where $j \in [k] \setminus \mathcal{S}$ and $g_j : \{0, 1\}^{n+\ell} \rightarrow \{0, 1\}^\zeta$.
 - Append $(j, g_j, g_j(w_j, E_j^b), E_j^b)$ to Tr .
 - Add j to $\mathcal{S}$.
- Output Tr .

Figure 4.2: The AdLeak^b Game

while extracting a constant fraction of the bits from the source, and while adaptively leaking a constant fraction of bits from it.

Lemma 4.1 *For all positive integers ℓ, ζ , every constant $\nu > 1$ and $\varepsilon \geq 0$, there is an explicit (efficient) $(n, \tau + \zeta, d, \ell, \varepsilon)$ -extractor that is $(\text{Full}_\zeta, \delta)$ -adaptive with $d = \mathcal{O}(\log(\frac{n}{\varepsilon}))$, $\tau = \nu\ell + \mathcal{O}(\log(\frac{n}{\varepsilon}))$, any $n \geq \tau + \zeta$ and $\delta = \varepsilon \cdot 2^{\ell+2}$.*

On further implication, for any $c > 1$, there exists constants α, β such that $d \leq \alpha\ell$, $\tau \leq \beta\ell$, $n \geq \beta\ell + \zeta$, $\varepsilon = 2^{-cl}$ and $\delta = 2^{(1-c)\ell+2}$ when $\ell = \omega(\log n)$.

Proof: The proof of the first part of the lemma follows directly from Theorem 4.1 and Lemma 2.10 and the further implication can be obtained by simple substitution. $\square$

Further, we use the following generalization of adaptive extractors: for an adaptive extractor Ext , if we consider k independent sources $W_1, \dots, W_k$ and a single seed S , all the extractor outputs $(\text{Ext}(W_i; S))_{i \in [k]}$ look uniform, even given adaptive leakage on each W_i , dependent on not just $\text{Ext}(W_i; S)$ (or uniform), but also all the prior extractor outputs and adaptive leakages (queried before i). As the sources are independent, this lemma can be proved using a simple hybrid argument.

Lemma 4.2 *Let k be an arbitrary positive integer, $W_1, \dots, W_k$ be k independent $(n, \tau + \zeta)$ sources and S be the uniform distribution on $\{0, 1\}^d$. Let Ext be an $(n, \tau + \zeta, d, \ell, \varepsilon)$ -extractor that is $(\text{Full}_\zeta, \delta)$ -adaptive. For each $i \in [k]$, let E_i^0 denotes $\text{Ext}(W_i; S)$, E_i^1 denotes uniform distribution on $\{0, 1\}^\ell$. For $b \in \{0, 1\}$, we define AdLeak^b as in Figure 4.2 Then for any stateful distinguisher $\mathcal{D}'$ we have $\text{AdLeak}^0 \approx_{k\delta} \text{AdLeak}^1$.*

Proof: The proof follows a simple hybrid argument. From the first hybrid AdLeak^0 , where the extractor outputs E_i^0 's are used, we proceed through k hybrids, where we replace E_j^0 with E_j^1

in the j -th hybrid, Hyb_j , for each $j \in [k]$. The final hybrid Hyb_k is identical to AdLeak^1 . Now, if we denote AdLeak^0 by Hyb_0 , then for each $j \in [k]$, a direct application of adaptive security of Ext_j together with the fact that all the remaining sources $\{W_m : m \in [k] \setminus \{j\}\}$ are independent of W_j gives us that $\text{Hyb}_{j-1} \approx_\delta \text{Hyb}_j$. Hence, by triangle inequality, the lemma is proved. $\square$

We also show the following property of strong seeded extractors, and in particular of adaptive extractors as well, which essentially states that the same seed can be used to extract from multiple independently drawn sources.

Lemma 4.3 *If Ext is an $(n, \tau, d, \ell, \varepsilon)$ -strong seeded extractor (Defintion 2.1), then for any $q \geq 1$ and any $\delta > 0$, Ext_q is an $(qn, (q-1)n + \tau, d, q\ell, q\varepsilon)$ -strong seeded extractor, where Ext_q is defined as follows:*

- Parse w as $w_1 || w_2 || \dots || w_q$ (where w_i is n -bit long, for all $i \in [q]$)
- Output $\text{Ext}(w_1; s) || \text{Ext}(w_2; s) || \dots || \text{Ext}(w_q; s)$

Further when the source(w) has uniform distribution, Ext_q is an $(qn, qn, d, q\ell, q\varepsilon)$ -extractor that is $(\text{Full}_{n-\tau}, 2^{q\ell+2}q\varepsilon)$ -adaptive.

Proof: Let $W|Z$ be the $(qn, (q-1)n + \tau)$ -average source, (where W is parsed as $(W_1, \dots, W_q)$) and $S \equiv U_d$. Then, by Lemma 2.1, $\tilde{\mathbf{H}}_\infty(W_q | W_1, \dots, W_{q-1}, Z) \geq \tau$. Therefore, by the security of Ext , we have

$$Z, W_1, \dots, W_{q-1}, \text{Ext}(W_q; S), S \approx_\varepsilon Z, W_1, \dots, W_{q-1}, U_\ell, S$$

Then by Lemma 2.2 it follows that,

$$Z, W_1, \dots, W_{q-2}, \text{Ext}(W_{q-1}; S), \text{Ext}(W_q; S), S \approx_\varepsilon Z, W_1, \dots, W_{q-2}, \text{Ext}(W_{q-1}; S), U_\ell, S$$

We now aim to show $\text{Ext}(W_{q-1}; S)$ is close to uniform even given $Z, W_1, \dots, W_{q-2}, U_\ell$ and S . Also, S remains uniform given $Z, W_1, \dots, W_{q-2}, U_\ell$. Since U_ℓ is independent of $W_1, \dots, W_{q-2}, W_{q-1}, Z$, we have $\tilde{\mathbf{H}}_\infty(W_{q-1} | W_1, \dots, W_{q-2}, Z, U_\ell) = \tilde{\mathbf{H}}_\infty(W_{q-1} | W_1, \dots, W_{q-2}, Z)$. This implies that, we have $\tilde{\mathbf{H}}_\infty(W_{q-1} | W_1, \dots, W_{q-2}, Z) \geq \tilde{\mathbf{H}}_\infty(W_{q-1} | W_1, \dots, W_{q-2}, Z, W_q)$ which is atleast τ (by Lemma 2.1). Then by security of Ext , we have

$$Z, W_1, \dots, W_{q-2}, \text{Ext}(W_{q-1}; S), U_\ell, S \approx_\varepsilon Z, W_1, \dots, W_{q-2}, U'_\ell, U_\ell, S^{20}$$

²⁰ U'_ℓ is a uniform sample from $\{0, 1\}^l$ independent of U_ℓ

Thus, by triangle inequality,

$$Z, W_1, \dots, W_{q-2}, \text{Ext}(W_{q-1}; S), \text{Ext}(W_q; S), S \approx_{2\varepsilon} Z, W_1, \dots, W_{q-2}, U_\ell, U_\ell, S$$

Then by similar arguments, it is easy to see that

$$Z, (\text{Ext}(W_1; S), \dots, \text{Ext}(W_{q-2}; S), \text{Ext}(W_{q-1}; S), \text{Ext}(W_q; S)), S \approx_{q\varepsilon} Z, U_{q\ell}, S$$

The adaptivity property follows from application of Theorem 4.1 on Ext_q . $\square$

4.3 Leakage Model: Joint and Adaptive

We consider two leakage models in this paper. For now, we restrict our discussion to an (N, t) -threshold access structure.

- **Adaptive Leakage and Reveal Model:** The adversary can adaptively obtain leakage on individual shares for any $N - t + 1$ shares. After this, he can additionally even get all the remaining $t - 1$ shares in their entirety.
- **Joint Leakage and Reveal Model:** The adversary can ask any number of joint leakage queries on disjoint sets of size X (a parameter). After this, he can additionally get any (at most $t - 1$) of the remaining shares in their entirety. While this model completely subsumes the adaptive leakage and reveal model, the amount of leakage per share supported in the latter would be lesser.

We provide a formal description of the adaptive leakage and reveal model and the joint leakage and reveal model in Sections 4.3.1 and 4.3.2, respectively. We give a construction that is leakage resilient with respect to both these models in Section 4.4 with a complete proof. We also briefly discuss the extension to general access structures in Section 4.4.4.

4.3.1 Adaptive Leakage and Reveal Model $\mathcal{F}_{leak}^{\psi, \mu}$

The model allows for leakage on individual shares and then also reveals at most $t - 1$ of the remaining shares in clear. We have two parameters in the model μ and ψ where μ denotes the amount of leakage provided in each leakage query and ψ captures the maximum number of leakage queries allowed. We allow ψ ranging from 1 to $N - t + 1$. Though we allow ψ to be $N - t + 1$, we have it as an explicit parameter because lower ψ would imply a weaker leakage model and possibly have better constructions. In fact, our multi-layered construction in Section 4.4 becomes compact (and offers better rate) as ψ decreases.

Let $(\text{Share}, \text{Rec})$ (where $\text{Share} : \{0, 1\}^\ell \rightarrow (\{0, 1\}^{\ell'})^N$) be a t -out-of- N secret sharing scheme. We formalize leakage obtained in this model on shares of a message m as $\text{Leak}_{\text{Share}}^m$ in Figure 4.3, where an arbitrary stateful distinguisher $\mathcal{D}$ makes the queries. For any two messages m and m' , we require $\text{Leak}_{\text{Share}}^m \approx_{\varepsilon_{lr}} \text{Leak}_{\text{Share}}^{m'}$, for $(\text{Share}, \text{Rec})$ to be ε_{lr} -leakage resilient against the adaptive leakage and reveal model (as in Definition 2.8).

$\text{Leak}_{\text{Share}}^m$:

- Initialize Z to be a null string and $\mathcal{S}$ to be a null set.

- $(Sh_1, \dots, Sh_N) \leftarrow \text{Share}(m)$.

- **Leakage Phase:**

For upto ψ times

- $(j, f_j) \leftarrow \mathcal{D}(Z)$ where $f_j : \{0, 1\}^{\ell'} \rightarrow \{0, 1\}^\mu$.
- If $j \in [N] \setminus \mathcal{S}$, add j to $\mathcal{S}$ and append $(j, f_j, f_j(Sh_j))$ to Z .

- **Reveal phase**

For upto $t - 1$ times

- $j \leftarrow \mathcal{D}(Z)$.
- If $j \in [N] \setminus \mathcal{S}$, append (j, Sh_j) to Z .

- $\mathcal{D}$ updates Z with any relevant state information.
- Output Z .

Figure 4.3: Adaptive Leakage and Reveal Model: $\text{Leak}_{\text{Share}}^m$ Distribution

4.3.2 Joint Leakage and Reveal Model $\mathcal{J}^{X, \psi, \mu}$

The model allows for ψ number of joint leakage queries on disjoint sets where each query depends on X number of shares and additionally also reveals $t - 1$ of the remaining shares (on which leakage isn't queried) in clear. The parameter μ captures the amount of leakage provided in each leakage query.

Let $(\text{Share}, \text{Rec})$ (where $\text{Share} : \{0, 1\}^\ell \rightarrow (\{0, 1\}^{\ell'})^N$) be a secret sharing scheme for an (N, t) - threshold access structure. We formalize leakage obtained in this model on shares of a

message m as $\text{JLeak}_{\text{Share}}^m$ in Figure 4.4, where an arbitrary stateful distinguisher $\mathcal{D}$ makes the queries. For any two messages m and m' , we require $\text{JLeak}_{\text{Share}}^m \approx_{\varepsilon_{lr}} \text{JLeak}_{\text{Share}}^{m'}$, for $(\text{Share}, \text{Rec})$ to be ε_{lr} leakage resilient against this model (as in Definition 2.8).

$\text{JLeak}_{\text{Share}}^m$:

- Initialize Z be a null string and $\mathcal{S}$ to be a null set.
- $(Sh_1, \dots, Sh_N) \leftarrow \text{Share}(m)$.
- **Leakage Phase:**
For upto ψ times
 - $(Q_j, f_j) \leftarrow \mathcal{D}(Z)$ where $Q_j \subseteq [n]$ and $f_j : \{0, 1\}^{|Q_j|^{\ell'}} \rightarrow \{0, 1\}^\mu$.
 - If $Q_j \in [N] \setminus \mathcal{S}$ and $|Q_j| \leq X$,
add elements of Q_j to $\mathcal{S}$ and append $(Q_j, f_j, f_j(Sh_{Q_j}))$ to Z .
- **Reveal phase**
For upto $t - 1$ times
 - $j \leftarrow \mathcal{D}(Z)$.
 - If $j \in [N] \setminus \mathcal{S}$, append (j, Sh_j) to Z .
- $\mathcal{D}$ updates Z to include any relevant state information.
- Output Z .

Figure 4.4: Joint Leakage and Reveal Model: $\text{JLeak}_{\text{Share}}^m$ Distribution

4.4 Our Construction: Leakage Resilient Secret Sharing

We begin by formally describing our construction, which we gave a detailed overview of in Section 4.1.1, and proving that it is leakage resilient against the weaker adaptive leakage and reveal model, in Section 4.4.2. Then, we subsequently prove that our construction is also leakage resilient against the joint adaptive and reveal model in Section 4.4.3. We also discuss the extension of our construction to the general access structures in Section 4.4.4.

4.4.1 Building Blocks and Construction

Let N be the number of parties and t be the reconstruction threshold. Let $h > 0$ be a parameter guaranteed to be less than $\lceil N/(t-1) \rceil$.

- Let $(\text{MShare}, \text{MRec})$ be an (t, N, ε_p) -secret sharing scheme (Definition 2.7) which is ε_{adp} -adaptively private (Section 2.6.1), for messages in $\{0, 1\}^\ell$ and share space being $\{0, 1\}^{\ell'}$.
- For $i \in [h]$, let $(\text{SdShare}^i, \text{SdRec}^i)$ be an $((n, t), \varepsilon'_i, \varepsilon'_{\text{adp}_i})$ -adaptive secret sharing scheme for messages in $\{0, 1\}^{d_i}$ with share space being $\{0, 1\}^{d'_i}$.
- For $i \in [h]$, let Ext^i be an $(n_i, \tau_i + \mu, d_i, \ell_i, \varepsilon_{\text{ext}_i})$ -extractor that is $(\text{Full}_\mu, \delta_i)$ -adaptive. We set $\ell_1 = \ell'$, and for $i \in [h] \setminus \{1\}$, we set $\ell_i = \ell_{i-1} + d'_{i-1}$.

We now describe our construction, which uses the above building blocks.

$\text{Share}^h(m)$:

- $(m_1, \dots, m_N) \leftarrow \text{MShare}(m)$.
- For $i \in [h]$, pick seeds $s^i \in_R \{0, 1\}^{d_i}$ and compute their shares $(sd_1^i, \dots, sd_N^i) \leftarrow \text{SdShare}^i(s^i)$.
- For $i \in [h]$ and $j \in [N]$, pick sources $w_j^i \in_R \{0, 1\}^{n_i}$.
- For $j \in [N]$:
 - Define $y_j^{h+1} = m_j$.
 - For $i \leftarrow h$ to 1, compute $x_j^i = y_j^{i+1} \oplus \text{Ext}^i(w_j^i; s^i)$ and $y_j^i = (x_j^i || sd_j^i)$.
- For $j \in [N]$, define $Sh_j = (w_j^1, \dots, w_j^h, y_j^1)$.
- Output $(Sh_1, \dots, Sh_N)$.

$\text{Rec}^h(Sh_T)$: (where T is the reconstruction set)

- For $j \in T$, parse Sh_j as $(w_j^1, \dots, w_j^h, y_j^1)$, where $y_j^1 = x_j^1 || sd_j^1$.
- For $i \leftarrow 1$ to h :
 - $s^i = \text{SdRec}^i(sd_T^i)$.

- For each $j \in T$, $y_j^{i+1} = x_j^i \oplus \text{Ext}^i(w_j^i; s^i)$. For each $i \in [h-1]$, parse y_j^{i+1} as $x_j^{i+1} || sd_j^{i+1}$.
- Parse y_j^{h+1} as m_j . Recover $m = \text{MRec}(m_T)$.
- Output m .

Figure 4.5: Construction of our LRSS

4.4.2 Security Proof: Adaptive Leakage and Reveal Model

Although we show that our construction tolerates the stronger joint leakage and reveal model, for ease of exposition we begin by proving security in the adaptive leakage and reveal model, and then show how to extend to the joint security.

Theorem 4.2 *For any $\psi \leq N - t + 1$ and $\ell, \mu > 0$, $(\text{Share}^h, \text{Rec}^h)$ is an (t, N, ε_p) -secret sharing scheme for ℓ -bit messages and is $(2\varepsilon_{\text{adp}} + 2 \sum_{i \in [h]} ((t-1)\delta_i + \varepsilon'_{\text{adp}_i}))$ -leakage resilient in the Adaptive Leakage and Reveal model $\mathcal{F}_{\text{leak}}^{\psi, \mu}$ where $h = \lceil \psi / (t-1) \rceil$.*

Further, there exists an instantiation of the scheme, given in Section 4.4.2.1, with rate is $(2^{\Theta(h)} + h\mu/\ell)^{-1}$. When either $N = \Theta(t)$ or h is a constant, the scheme achieves a constant rate and tolerates leakage upto $\mu = \Theta(\ell)$ bits from each share, asymptotically.

Proof: The correctness of the scheme follows directly from the correctness of underlying extractors and secret sharing schemes. The (adaptive) privacy of the scheme is directly implied by the leakage resilience (against the adaptive leakage and reveal model).

Leakage Resilience. For any message m we define the following the sequence of hybrids. In these hybrids we assume that $\mathcal{D}$ always asks legitimate queries as per the model and won't write explicit checks for legitimacy (for example, we assume that $\mathcal{D}$ doesn't ask leakage on same share twice).

We analyze the leakage queries made by $\mathcal{D}$ as bunches of $(t-1)$ queries. We now introduce some useful notation. Let $\mathcal{S}_1, \dots, \mathcal{S}_h$ denote the sets of indices queried by $\mathcal{D}$, where $\mathcal{S}_i$ contains the indices queried by $\mathcal{D}$ from the $((i-1)(t-1) + 1)^{\text{th}}$ query to $i(t-1)^{\text{th}}$ leakage queries (i.e., $\mathcal{S}_1$ contains the first $t-1$ queries, $\mathcal{S}_2$ the next $t-1$ queries and so on). For $i \in [h]$, we use $\mathcal{S}_{[i]}$ to denote $\bigcup_{j=1}^i \mathcal{S}_j$, which captures the set of indices queried in the first $i(t-1)$ leakage queries. For $i \in [h]$, let $Z_{[i]}$ denotes the set of leakage queries and the corresponding responses to the first $i(t-1)$ leakage queries. $Z_{[h+1]}$ denotes $Z_{[h]}$ together with the final reveal queries as well as any relevant state information. We prove leakage resilience using a hybrid argument, with the

following sequence of hybrids, LeakB_0^m , $\{\text{LeakA}_q^m, \text{LeakB}_q^m\}_{q \in [h]}$ and LeakC^m . The order of the hybrids is $\text{LeakB}_0^m, \text{LeakA}_1^m, \text{LeakB}_1^m, \dots, \text{LeakA}_h^m, \text{LeakB}_h^m, \text{LeakC}^m$, where we will show that LeakC^m is independent of m , and LeakB_0^m will correspond to the distribution $\text{Leak}_{\text{Share}^h}^m$. This will allow us to show that $\text{Leak}_{\text{Share}^h}^m$ is indistinguishable from $\text{Leak}_{\text{Share}^h}^{m'}$. We begin by giving an informal description of these hybrids.

LeakA_q^m : We start with $q = 1$. LeakA_1^m follows the actual leakage game i.e., $\text{Leak}_{\text{Share}^h}^m (\equiv \text{LeakB}_0^m)$ except for the following change: we replace the shares sd_j^1 , for each $j \in \mathcal{S}_1$ (the shares of s^1 corresponding to the first $t - 1$ leakage queries), with shares of a dummy seed $\tilde{s}^1 = 0^d$. In general, for each $1 < q \leq h$, the only change we make in LeakA_q^m (in comparison to the previous hybrid LeakB_{q-1}^m) is that we replace the shares sd_j^q , for each $j \in \mathcal{S}_q$ (the shares of s^q corresponding to the q -th set of $t - 1$ leakage queries), with shares of a dummy seed $\tilde{s}^q$. After answering the leakage queries corresponding to $\mathcal{S}_q$, shares of s^q are re-sampled consistent with the dummy seed shares used so far. The hybrid is formally described in Figure 4.6.

LeakB_q^m : For $q = 1$, LeakB_1^m follows the hybrid LeakA_1^m except for the following change: in LeakB_1^m , we replace the values x_j^1 , for each $j \in \mathcal{S}_1$ with random, instead of evaluating the h layers of masking to get x_j^1 (and hence x_j^1 's for $j \in \mathcal{S}_1$ are independent of $m_{\mathcal{S}_1}$, s^i and the shares of s^i , for each $1 < i \leq h$). Note that in LeakA_1^m , the shares Sh_j corresponding to $\mathcal{S}_1$ no longer depend on the seed s^1 . We carefully use the adaptive extractor security of Ext^1 to move to LeakB_1^m . In general, for each $1 < q \leq h$, the only change we make in LeakB_q^m (in comparison to the previous hybrid LeakA_q^m) is that we replace the values x_j^q , for each $j \in \mathcal{S}_q$ with random, instead of evaluating the $h - (q - 1)$ layers of masking to get x_j^q (and hence, for these queries in $\mathcal{S}_q$, s^i and the shares of s^i , for each $q < i \leq h$, and the shares m are not used to evaluate x_j^q). Further, we continue the steps of masking to evaluate $x_j^{q-1}, x_j^{q-2}, \dots, x_j^1$, for each $j \in \mathcal{S}_q$ as in the previous hybrid. The hybrid is formally described in Figure 4.7.

LeakC^m : In the hybrid LeakB_h^m , all the shares used in the leakage phase are independent of the shares of the message m . Hence, the only part of the view of $\mathcal{D}$ that depends on the shares of m corresponds to the reveal phase. In the final hybrid LeakC^m , we replace the $t - 1$ shares of m used in the reveal phase by shares of 0^ℓ . This hybrid is formally described in Figure 4.8. The formal descriptions of all hybrids are given below with the change from the prior hybrid highlighted in red color.

LeakA_q^m :

1. Initialize Z to be a null string and $\mathcal{S}_1, \dots, \mathcal{S}_h$ to be null sets.

2. $(m_1, \dots, m_N) \leftarrow \text{MShare}(m)$.
3. For $i \in [h]$, choose $s^i \in_R \{0, 1\}^{d_i}$.
4. For $i \in [h]$ and $j \in [N]$, choose $w_j^i \in_R \{0, 1\}^{n_i}$.
5. For $i \in [h] \setminus [q]$, compute $(sd_1^i, \dots, sd_N^i) \leftarrow \text{SdShare}^i(s^i)$.
6. For $i \in [q]$, let $\tilde{s}^i = 0^d$.
7. For $j \in [N]$, define $y_j^{h+1} = m_j$.

8. Leakage Phase:

- (a) For $c \leftarrow 1$ to q :
 - i. $(\tilde{sd}_1^c, \dots, \tilde{sd}_N^c) \leftarrow \text{SdShare}^c(\tilde{s}^c)$.
 - ii. For up to $(t-1)$ times:
 - A. $(j, f_j) \leftarrow \mathcal{D}(Z)$.
 - B. If $c < q$,
 - * Choose $x_j^c \in_R \{0, 1\}^{\ell_c}$ and compute $y_j^c = (x_j^c || \tilde{sd}_j^c)$.
 - * For $i \leftarrow c-1$ down to 1,
 - compute $x_j^i = y_j^{i+1} \oplus \text{Ext}^i(w_j^i; s^i)$ and $y_j^i = (x_j^i || sd_j^i)$.
 - C. If $c = q$, for $i \leftarrow h$ down to 1 compute

$$\begin{cases} x_j^i = y_j^{i+1} \oplus \text{Ext}^i(w_j^i; s^i) \text{ and } y_j^i = (x_j^i || sd_j^i) & \text{when } i \neq q \\ x_j^i = y_j^{i+1} \oplus \text{Ext}^i(w_j^i; s^i) \text{ and } y_j^i = (x_j^i || \tilde{sd}_j^i) & \text{when } i = q \end{cases}$$
 - D. Define $Sh_j = (w_j^1, \dots, w_j^h, y_j^1)$.
 - E. Add j to $\mathcal{S}_c$ and append $(j, f_j, f_j(Sh_j))$ to Z .
 - iii. $(sd_1^c, \dots, sd_N^c) \leftarrow \text{SdShare}^c(s^c | \tilde{sd}_{\mathcal{S}_c}^c)$.
- (b) For $j \in [N] \setminus (\mathcal{S}_{[q]})$ and $i \leftarrow h$ down to 1,
 - compute $x_j^i = y_j^{i+1} \oplus \text{Ext}^i(w_j^i; s^i)$ and $y_j^i = (x_j^i || sd_j^i)$.
- (c) Define $Sh_j = (w_j^1, \dots, w_j^h, y_j^1)$.
- (d) For $c \leftarrow q+1$ to h :
 - i. For upto $t-1$ times:
 - A. $(j, f_j) \leftarrow \mathcal{D}(Z)$.

B. Add j to $\mathcal{S}_c$ and append $(j, f_j, f_j(Sh_j))$ to Z .

9. Reveal phase

(a) For upto $t - 1$ times:

- i. $j \leftarrow \mathcal{D}(Z)$.
- ii. Append (j, Sh_j) to Z .

10. $\mathcal{D}$ updates Z with any relevant state information.

11. Output Z .

Figure 4.6: Hybrid LeakA_q^m

LeakB_q^m

1. Initialize Z to be a null string and $\mathcal{S}_1, \dots, \mathcal{S}_h$ to be null sets.

2. $(m_1, \dots, m_N) \leftarrow \text{MShare}(m)$.

3. For $i \in [h]$, choose $s^i \in_R \{0, 1\}^{d_i}$.

4. For $i \in [h]$ and $j \in [N]$, choose $w_j^i \in_R \{0, 1\}^{n_i}$.

5. For $i \in [h] \setminus [q]$, compute $(sd_1^i, \dots, sd_N^i) \leftarrow \text{SdShare}^i(s^i)$.

6. For $i \in [q]$, let $\tilde{s}^i = 0^d$.

7. For $j \in [N]$, define $y_j^{h+1} = m_j$.

8. Leakage Phase:

(a) For $c \leftarrow 1$ to q :

i. $(\tilde{sd}_1^c, \dots, \tilde{sd}_N^c) \leftarrow \text{SdShare}^c(\tilde{s}^c)$.

ii. For upto $(t - 1)$ times:

A. $(j, f_j) \leftarrow \mathcal{D}(Z)$.

B. Choose $x_j^c \in_R \{0, 1\}^{\ell_c}$ and compute $y_j^c = (x_j^c || \tilde{sd}_j^c)$.

C. For $i \leftarrow c - 1$ down to 1:

compute $x_j^i = y_j^{i+1} \oplus \text{Ext}^i(w_j^i; s^i)$ and $y_j^i = (x_j^i || sd_j^i)$.

- D. Define $Sh_j = (w_j^1, \dots, w_j^h, y_j^1)$.
 - E. Add j to $\mathcal{S}_c$ and append $(j, f_j, f_j(Sh_j))$ to Z .
 - iii. $(sd_1^c, \dots, sd_N^c) \leftarrow \text{SdShare}^c(s^c | \tilde{sd}_{\mathcal{S}_c}^c)$.
 - (b) For $j \in [N] \setminus \mathcal{S}_{[q]}$ and $i \leftarrow h$ to 1, ($\mathcal{S}_{[q]}$ denotes a null set when $q = 0$) compute $x_j^i = y_j^{i+1} \oplus \text{Ext}^i(w_j^i; s^i)$ and $y_j^i = (x_j^i || sd_j^i)$.
 - (c) Define $Sh_j = (w_j^1, \dots, w_j^h, y_j^1)$.
 - (d) For $c \leftarrow q + 1$ to h :
 - i. For upto $t - 1$ times:
 - A. $(j, f_j) \leftarrow \mathcal{D}(Z)$.
 - B. Add j to $\mathcal{S}_c$ and append $(j, f_j, f_j(Sh_j))$ to Z .
9. **Reveal phase**
- (a) For upto $t - 1$ times:
 - i. $j \leftarrow \mathcal{D}(Z)$.
 - ii. Append (j, Sh_j) to Z .
10. $\mathcal{D}$ updates Z with any relevant state information.
11. Output Z .

Figure 4.7: Hybrid LeakB_q^m

LeakC^m

1. Initialize Z to be a null string and $\mathcal{S}_1, \dots, \mathcal{S}_h$ to be null sets.
2. Let $\tilde{m} = 0^\ell$ and $(\tilde{m}_1, \dots, \tilde{m}_N) \leftarrow \text{MShare}(\tilde{m})$.
3. For $i \in [h]$, choose $s^i \in_R \{0, 1\}^{d_i}$.
4. For $i \in [h]$, let $\tilde{s}^i = 0^d$.
5. For $i \in [h]$ and $j \in [N]$, choose $w_j^i \in_R \{0, 1\}^{n_i}$.
6. **Leakage Phase:**
 - (a) For $c \leftarrow 1$ to h :

- i. $(\tilde{sd}_1^c, \dots, \tilde{sd}_N^c) \leftarrow \text{SdShare}^c(\tilde{s}^c)$.
- ii. For upto $(t - 1)$ times:
 - A. $(j, f_j) \leftarrow \mathcal{D}(Z)$.
 - B. Choose $x_j^c \in_R \{0, 1\}^{\ell_c}$ and compute $y_j^c = (x_j^c || \tilde{sd}_j^c)$.
 - C. For $i \leftarrow c - 1$ down to 1:
 - compute $x_j^i = y_j^{i+1} \oplus \text{Ext}^i(w_j^i; s^i)$ and $y_j^i = (x_j^i || sd_j^i)$.
 - D. Define $Sh_j = (w_j^1, \dots, w_j^h, y_j^1)$.
 - E. Add j to $\mathcal{S}_c$ and append $(j, f_j, f_j(Sh_j))$ to Z .
- iii. $(sd_1^c, \dots, sd_N^c) \leftarrow \text{SdShare}^c(s^c | \tilde{sd}_{\mathcal{S}_c}^c)$.

7. Reveal phase

- (a) For upto $t - 1$ times:
 - i. $j \leftarrow \mathcal{D}(Z)$.
 - ii. Define $y_j^{h+1} = \tilde{m}_j$.
 - iii. For $i \leftarrow h$ to 1, compute $x_j^i = y_j^{i+1} \oplus \text{Ext}^i(w_j^i; s^i)$ and $y_j^i = (x_j^i || sd_j^i)$.
 - iv. Define $Sh_j = (w_j^1, \dots, w_j^h, y_j^1)$.
 - v. Append (j, Sh_j) to Z .

8. $\mathcal{D}$ updates Z with any relevant state information.

9. Output Z .

Figure 4.8: Hybrid LeakC^m

We begin by proving the statistical closeness of LeakA_q^m and LeakB_{q-1}^m , for each $q \in [h]$, which follows from adaptive privacy of SdShare^q , as atmost only $t - 1$ dummy seed shares are used.

Claim 3 For $q \in [h]$, if SdShare^q is $\varepsilon'_{\text{adp}_q}$ -adaptively private against (N, t) -threshold access structures, then $\text{LeakA}_q^m \approx_{\varepsilon'_{\text{adp}_q}} \text{LeakB}_{q-1}^m$.

Proof: **Answering the first $(q - 1)$ sets of leakage queries (when $q > 1$):** Observe that the hybrids are identical up to answering the first $(q - 1)(t - 1)$ leakage queries and differ in answering the remaining queries. For any $k \in [q - 1]$ and, $j \in \mathcal{S}_k$ the leakage response only depends on $\tilde{sd}_j^k, w_j^1, \dots, w_j^h$ and $\{s^i, sd_j^i\}_{1 \leq i < k}$ (as x_j^k is chosen uniformly). We let Pre denote the union of these random variables upon which the leakage responses to $j \in \mathcal{S}_{[q-1]}$ depend.

Answering the q^{th} set of leakage queries: Consider $j \in \mathcal{S}_q$. To answer this leakage

query, it suffices to compute $Sh_j = (w_j^1, \dots, w_j^h, y_j^1)$. The hybrids only differ in computation of y_j^1 (particularly in computation of y_j^q , which is used to compute y_j^1) and the distribution of extractor sources is identical in both. We highlight the differences here. $\mathbf{LeakA}_q^m$ (Step 8-(a)-ii-C), iteratively computes $y_j^h, \dots, y_j^q, \dots, y_j^1$ as follows.

- $(y_j^h, \dots, y_j^{q+1})$ are computed using y_j^{h+1} and $\{w_j^i, sd_j^i, s^i\}_{i \in [h] \setminus [q]}$. Note that the distribution of $y_j^h, \dots, y_j^{q+1}$ is identical in both hybrids.
- x_j^q is computed using y_j^{q+1}, w^q and s^q . x_j^q is also identical in both hybrids.
- y_j^q is computed as $x_j^q || \tilde{sd}_j^q$ (where $\tilde{sd}_{[N]}^q$ are shares of a dummy seed $\tilde{s}^q$ which are generated before answering any queries in $\mathcal{S}_q$ in Step 8-(a)-i (when $c = q$)). Whereas in $\mathbf{LeakB}_{q-1}^m$, $y_j^q = x_j^q || sd_j^q$ (where $sd_{[N]}^q$ are shares of s^q).
- $(y_j^{q-1}, \dots, y_j^1)$ are computed using y_j^q and $\{sd_j^i, w_j^i, s^i\}_{i \in [q-1]}$. The computation of $(y_j^{q-1}, \dots, y_j^1)$ given the later random variables is again identical to $\mathbf{LeakB}_{q-1}^m$.
- Now $\mathbf{LeakA}_q^m$ defines $Sh_j = (w_j^1, \dots, w_j^h, y_j^1)$

For convenience, in this proof we distinguish (whenever necessary) the random variables that have same literal in both the hybrids but are distributionally different with subscripts A and B respectively. For example, $y_{j,A}^q$ and $y_{j,B}^q$ denote the distributions of y_j^q in $\mathbf{LeakA}_q^m$ and, $\mathbf{LeakB}_{q-1}^m$ respectively.

Let $\mathbf{Pre}' = (\{w_j^q, \{sd_j^i, w_j^i, s^i\}_{i \in [h] \setminus \{q\}}\}_{j \in [N] \setminus \mathcal{S}_{[q-1]}})$. $\mathbf{Pre}'$ captures the information required to answer all queries after the first $q-1$ sets of leakage queries, except for any information regarding $s^q, \tilde{s}^q$ and their shares. Note that $\mathbf{Pre}'$ is identical in both hybrids²¹. Since, $|\mathcal{S}_q| \leq t-1$, with a reduction to adaptive privacy of $\mathbf{SdShare}^q$ we have

$$\mathbf{Pre}, \mathbf{Pre}', s^q, \tilde{s}^q, \{\tilde{sd}_j^q\}_{j \in \mathcal{S}_{q,A}} \approx_{\varepsilon'_{adp_q}} \mathbf{Pre}, \mathbf{Pre}', s^q, \tilde{s}^q, \{sd_j^q\}_{j \in \mathcal{S}_{q,B}}$$

as $(\mathbf{Pre}, \mathbf{Pre}')$ is independent of the randomness used to generate the shares of $\tilde{s}^q$ and s^q . Note that the information on LHS suffices to answer the first q sets of queries as per $\mathbf{LeakA}_q^m$. Similarly, RHS suffices to answer queries in $\mathcal{S}_{[q]}$ as per $\mathbf{LeakB}_{q-1}^m$. Therefore, we have,

$$\mathbf{Pre}, \mathbf{Pre}', s^q, \tilde{s}^q, \{\tilde{sd}_j^q\}_{j \in \mathcal{S}_{q,A}}, Z_{[q],A} \approx_{\varepsilon'_{adp_q}} \mathbf{Pre}, \mathbf{Pre}', s^q, \tilde{s}^q, \{sd_j^q\}_{j \in \mathcal{S}_{q,B}}, Z_{[q],B} \quad (4.1)$$

²¹ $\mathbf{Pre}'$ possibly repeats some information already there in $\mathbf{Pre}$. For example for $q = 2$, s^1 is there in both $\mathbf{Pre}$ and $\mathbf{Pre}'$. It is for the ease of exposition that we have this repetition.

Answering the leakage and reveal queries made after the q^{th} set of leakage queries:

After all the q^{th} set leakage queries are answered, LeakA_q^m computes:

$(sd_1^q, \dots, sd_N^q) \leftarrow \text{SdShare}^q(s^q | \tilde{sd}_{\mathcal{S}_{q,A}}^q)$. Given $(sd_1^q, \dots, sd_N^q), s^q, \text{Pre}$ and Pre' , for any $j \in [N] \setminus \mathcal{S}_q$, Sh_j is easily computed (Steps 8-(b) and 8-(c)). With this, any further queries can be correctly answered as per LeakA_q^m . Let $(\hat{sd}_1^q, \dots, \hat{sd}_N^q) \leftarrow \text{SdShare}^q(s^q | sd_{\mathcal{S}_{q,B}}^q)$. By Lemma 2.2, we have

$$\text{Pre}, \text{Pre}', s^q, \tilde{s}^q, Z_{[q],A}, sd_{[N],A}^q \approx_{\varepsilon'_{\text{adp}_q}} \text{Pre}, \text{Pre}', s^q, \tilde{s}^q, Z_{[q],B}, \hat{sd}_{[N],B}^q$$

Note that $\hat{sd}_{[N]}^q$ is identical to $sd_{[N],B}^q$ (of LeakB_{q-1}^m) even given s^q and $\{sd_j^q\}_{j \in \mathcal{S}_q}$ by the property of consistent resampling in Claim 2.17. Therefore, we have,

$$\text{Pre}, \text{Pre}', s^q, Z_{[q],A}, sd_{[N],A}^q \approx_{\varepsilon'_{\text{adp}_q}} \text{Pre}, \text{Pre}', s^q, Z_{[q],B}, sd_{[N],B}^q$$

Since the above LHS and RHS are sufficient to answer any further queries, we have

$$Z_{[h+1],A} \approx_{\varepsilon'_{\text{adp}_q}} Z_{[h+1],B}$$

which proves the claim. $\square$

Now, we prove the statistical closeness of LeakA_q^m and LeakB_q^m , for each $q \in [h]$ using the adaptive extractor security. The high-level idea behind the reduction is that in hybrid LeakA_q^m , the shares corresponding to the first $q(t-1)$ queries (i.e., $\mathcal{S}_{[q]}$) no longer depend on the seed s^q and hence, we can use the adaptive extractor security of Ext^q to move to LeakB_q^m .

Claim 4 For $q \in [h]$, if Ext^q is an $(n_q, \tau_q + \mu, d_q, \ell_q, \varepsilon_{\text{ext}q})$ - extractor that is $(\text{Full}_\mu, \delta_q)$ -adaptive, then $\text{LeakA}_q^m \approx_{(t-1)\delta_q} \text{LeakB}_q^m$.

Proof: Observe that the hybrids are identical up to answering the first $(q-1)(t-1)$ leakage queries and differ in answering the q^{th} set of queries. Further, after answering the q^{th} set of leakage queries, the responses to all remaining leakage/reveal queries are answered identically in both hybrids.

Answering the first $(q-1)$ sets of leakage queries (when $q > 1$):

For any $k \in [q-1]$ and $j \in \mathcal{S}_k$ the leakage response only depends on $\tilde{sd}_j^k, w_j^1, \dots, w_j^h, \{s^i, sd_j^i\}_{1 \leq i < k}$ and x_j^k , where the latter is uniformly chosen. We let Pre denote the leakage responses $Z_{[q-1]}$ and the union of these random variables upon which the leakage responses to $j \in \mathcal{S}_{[q-1]}$ depend.

Answering the q^{th} set of leakage queries:

Consider $j \in \mathcal{S}_q$ and f_j be the corresponding leakage function. To answer this leakage query, we require computing $f_j(Sh_j)$ where $Sh_j = (w_j^1, \dots, w_j^h, y_j^1)$. The hybrids only differ in computation of y_j^1 (particularly in computation of x_j^q , which is used to compute y_j^1) and the distribution of extractor sources is identical in both. The hybrids iteratively computes $y_j^q, \dots, y_j^1$ as follows.

- x_j^q is chosen uniformly from $\{0, 1\}^{\ell_q}$ in LeakB_q^m . In contrast, x_j^q of LeakA_q^m depended on $\text{Ext}^q(w_j^q; s^q)$ and y_j^{q+1} .
- $(y_j^q, \dots, y_j^1)$ is determined given $x_j^q, \tilde{sd}_j^q$ and $\{sd_j^i, w_j^i, s^i\}_{i \in [q-1]}$ in both the hybrids.
- Both hybrids define $Sh_j = (w_j^1, \dots, w_j^h, y_j^1)$.

Let $\text{Pre}' = \{w_j^i, sd_j^i, s^i, y_j^{h+1}, \tilde{sd}_j^q\}_{i \in [h] \setminus \{q\}, j \in [N] \setminus \mathcal{S}_{[q-1]}}$. We capture Pre' as the information which along with $\{w_j^q, s^q\}_{j \in \mathcal{S}_q}$ is sufficient to answer any leakage queries on $j \in \mathcal{S}_q$. Also, Pre' is identical in both hybrids.

Let $j_1, \dots, j_{t-1}$ be the order of indices in which leakage queries are made in $\mathcal{S}_q$. Firstly, we prove that $(\text{Pre}, \text{Pre}', f_{j_1}(Sh_{j_1}))$ of both hybrids are statistically close. After that we proceed to show that $(\text{Pre}, \text{Pre}', f_{j_1}(Sh_{j_1}), \dots, f_{j_{t-1}}(Sh_{j_{t-1}}))$ of both the hybrids are statistically close, which implies that the hybrids are statistically close up to answering first q sets of queries. For convenience, in this proof we distinguish (whenever necessary) the random variables that have same literal in the hybrids but are distributionally different with subscripts A and B respectively. For example, $x_{j,A}^q$ and $x_{j,B}^q$ denote the distributions of x_j^q in LeakA_q^m and LeakB_q^m respectively.

Firstly, in both hybrids the distribution of (j_1, f_{j_1}) only depends on $Z_{[q-1]}$ (and any internal randomness of $\mathcal{D}$) and hence are identical. Note that given $\text{Pre}', f_{j_1}(Sh_{j_1})$ in LeakA_q^m , can be captured as Full_μ -adaptive leakage on the extractor source $w_{j_1}^q$ and $(x_{j_1,A}^q =) \text{Ext}^q(w_{j_1}^q; s^q) \oplus y_{j_1}^{q+1}$. This is because $(y_{j_1}^{q+1}, \text{Pre}')$ are independent of $(w_{j_1}^q, s^q)$. Let g_1 be a function that takes $\text{Pre}', w_{j_1}^q$ and $x_{j_1,A}^q$ (or $x_{j_1,B}^q$) as input, computes $y_{j_1,A}^1$ (or $y_{j_1,B}^1$) and outputs $f_j(w_{j_1}^1, \dots, w_{j_1}^h, y_{j_1,A}^1)$ (or $f_j(w_{j_1}^1, \dots, w_{j_1}^h, y_{j_1,B}^1)$). With a reduction to adaptive security of Ext^q we have

$$\begin{aligned} \text{Pre}, \text{Pre}', s^q, g_1(\text{Pre}', w_{j_1}^q, \text{Ext}^q(w_{j_1}^q; s^q) \oplus y_{j_1}^{q+1}) &\approx_{\delta_q} \text{Pre}, \text{Pre}', s^q, g_1(\text{Pre}', w_{j_1}^q, U_{\ell_q} \oplus y_{j_1}^{q+1}) \\ &\equiv \text{Pre}, \text{Pre}', s^q, g_1(\text{Pre}', w_{j_1}^q, x_{j_1,B}^q) \end{aligned}$$

Therefore

$$\text{Pre}, \text{Pre}', s^q, f_{j_1}(Sh_{j_1,A}) \approx_{\delta_q} \text{Pre}, \text{Pre}', s^q, f_{j_1}(Sh_{j_1,B})$$

With this, we showed that the hybrids are statistically close up to responding to the first query in the q^{th} set. Although, superficially, it may seem that all the leakage responses corre-

sponding to $j \in \mathcal{S}_q$ can be captured as adaptive extractor leakage on the source w_j^q , but it's not the case because of the following subtlety. The extractor sources used in each query are independent of each other, but the seed is the same. For example, one cannot directly capture $f_{j_2}(Sh_{j_2})$ as Full_μ -adaptive leakage (as we did with $f_{j_1}(Sh_{j_1})$). This is because the choice of j_2 , f_{j_2} depends on $f_{j_1}(Sh_{j_1})$ which in turn depends on $\text{Ext}^q(w_j^q; s^q)$, and hence is not independent of the seed s^q . We observe in Lemma 4.2 that adaptive extractors allow us to handle even such (stronger) form of adaptive leakages across different sources with same seed.

Proceeding, with a reduction to Lemma 4.2 with $k = (t - 1)$, $\{W_i = W_{j_i}^q : i \in [k]\}$, $S = s^q$ and $\text{Ext} = \text{Ext}^q$ and the i^{th} leakage function being g_i such that g_i (hardwired with $\text{Pre}', y_{j_i}^{q+1}$) takes $w_{j_i}^q$ and $\text{Ext}^q(w_{j_i}^q; s^q)$ (resp. U_{ℓ_q}) as input, computes $y_{j_i,A}^1$ (resp. $y_{j_i,B}^1$) and outputs $f_{j_i}(w_{j_i}^1, \dots, w_{j_i}^h, y_{j_i,A}^1)$ (resp. $f_{j_i}(w_{j_i}^1, \dots, w_{j_i}^h, y_{j_i,B}^1)$).

$$\text{Pre}, \text{Pre}', s^q, \{f_{j_i}, f_{j_i}(Sh_{j_i,A})\}_{j_i \in \mathcal{S}_{q,A}}, \mathcal{S}_{q,A} \approx_{(t-1)\delta_q} \text{Pre}, \text{Pre}', s^q, \{f_{j_i}, f_{j_i}(Sh_{j_i,B})\}_{j_i \in \mathcal{S}_{q,B}}, \mathcal{S}_{q,B}$$

This shows that the hybrids are statistically close up to answering the first q sets of leakage queries.

Answering the leakage and reveal queries made after the q^{th} set of leakage queries:

After all the q^{th} set of leakage queries are answered, both hybrids compute $(sd_1^q, \dots, sd_N^q) \leftarrow \text{SdShare}(s^q | \tilde{sd}_{\mathcal{S}_q}^q)$. Let $\text{Pre}'' = \{w_j^q, sd_j^q, s^q\}_{j \in [N] \setminus \mathcal{S}_q}$. Note that Pre' in conjunction with Pre'' completely defines Sh_j for any $j \in [N] \setminus \mathcal{S}_{[q]}$. Since Pre'' corresponding to LeakA_q^m (resp. LeakB_q^m) is only correlated to $\mathcal{S}_q, s^q$ and $\tilde{sd}_{\mathcal{S}_q}^q$ (which is in Pre') of the respective hybrids, we have

$$\text{Pre}, \text{Pre}', \text{Pre}_A'', s^q, \{f_{j_i}, f_{j_i}(Sh_{j_i,A})\}_{j_i \in \mathcal{S}_{q,A}}, \mathcal{S}_{q,A} \approx_{(t-1)\delta_q} \text{Pre}, \text{Pre}', \text{Pre}_B'', s^q, \{f_{j_i}, f_{j_i}(Sh_{j_i,B})\}_{j_i \in \mathcal{S}_{q,B}}, \mathcal{S}_{q,B}$$

Since responses to leakage/reveal queries after the q^{th} set are can be derived from the LHS and RHS respectively depending on the hybrid, we have

$$Z_{[h+1],A} \approx_{(t-1)\delta_q} Z_{[h+1],B}$$

This proves the claim. $\square$

Finally, we use the adaptive privacy of MShare to show that LeakC^m is statistically close to LeakB_h^m .

Claim 5 *If MShare is ε_{adp} -adaptively private against (N, t) -threshold access structures, then $\text{LeakC}^m \approx_{\varepsilon_{\text{adp}}} \text{LeakB}_h^m$.*

Proof: The hybrids answer the leakage queries identically and differ only in answering the

reveal queries.

Answering the leakage queries:

For any $k \in [h]$ and $j \in \mathcal{S}_k$ the leakage response only depends on $\tilde{sd}_j^k, w_j^1, \dots, w_j^h, \{s^i, sd_j^i\}_{1 \leq i < k}$ and x_j^k , where the latter is uniformly chosen. We let Pre denote the leakage responses $Z_{[h]}$ and the union of these random variables upon which the leakage responses to $j \in \mathcal{S}_{[h]}$ depend.

Answering the reveal queries: Let $\text{Pre}' = \{w_j^i, sd_j^i, s^i\}_{i \in [h], j \in [N] \setminus \mathcal{S}_{[h]}}$. Note that given y_j^{h+1} for all j queried in the reveal phase, $(\text{Pre}, \text{Pre}')$ has sufficient information to answer all the reveal queries.

- LeakB_h^m samples $(m_1, \dots, m_N) \leftarrow \text{MShare}(m)$ and sets $y_j^{h+1} = m_j$ for all j queried in the reveal phase.
- LeakC^m samples $(\tilde{m}_1, \dots, \tilde{m}_N) \leftarrow \text{MShare}(\tilde{m})$ and sets $y_j^{h+1} = \tilde{m}_j$ for all j queried in the reveal phase.

Let RevealB and RevealC denote the sets of indices queried in the reveal phase of LeakB_h^m and LeakC^m respectively. As reveal queries are at most $t - 1$ in number, we now invoke adaptive privacy of MShare and get

$$\text{Pre}, \text{Pre}', \tilde{m}, m, \{m_j\}_{j \in \text{RevealB}} \approx_{\varepsilon_{\text{adp}}} \text{Pre}, \text{Pre}', \tilde{m}, m, \{\tilde{m}_j\}_{j \in \text{RevealC}}$$

Note that $(\text{Pre}, \text{Pre}')$ is independent of the randomness used in generating shares of m and $\tilde{m}$, therefore adaptive privacy of MShare can be invoked even given these random variables.

Since Sh_j for j queried in reveal phase of LeakB_h^m (resp. LeakC^m) is determined by the above LHS (resp. RHS) we have

$$\underbrace{Z_{[h+1]}}_{\text{of } \text{LeakB}_h^m} \approx_{\varepsilon_{\text{adp}}} \underbrace{Z_{[h+1]}}_{\text{of } \text{LeakC}^m}$$

□

Using the Claims 3, 4 and 5, and by triangle inequality we know that for any message m , $\text{Leak}_{\text{Share}^h}^m \approx_{\varepsilon_{\text{adp}} + \sum_{i \in [h]} ((t-1)\delta_i + \varepsilon'_{\text{adp}_i})} \text{LeakC}^m$. Note that the description of LeakC^m is independent of m . Hence, for any message $m \neq m'$, we have $\text{LeakC}^m \equiv \text{LeakC}^{m'}$. Since we also have, $\text{Leak}_{\text{Share}^h}^{m'} \approx_{\varepsilon_{\text{adp}} + \sum_{i \in [h]} ((t-1)\delta_i + \varepsilon'_{\text{adp}_i})} \text{LeakC}^{m'}$, we get:

$$\text{Leak}_{\text{Share}^h}^m \approx_{2\varepsilon_{\text{adp}} + 2\sum_{i \in [h]} ((t-1)\delta_i + \varepsilon'_{\text{adp}_i})} \text{Leak}_{\text{Share}^h}^{m'},$$

which proves the leakage resilience, and hence the theorem. □

4.4.2.1 Instantiation and Rate Analysis

For $i \in [h]$, we instantiate SdShare^i on seeds of length d_i with the (adaptively) private Shamir secret sharing scheme (Section 2.6.1), which results in individual seed share length being d_i . We instantiate MShare on messages of length ℓ with the (adaptively) private Shamir secret sharing scheme, which results in individual seed share length being ℓ .

Further, we instantiate the adaptive extractor with the extractor from Lemma 4.1, which states that for any $c > 1$, there exists constants α, β such that $d \leq \alpha\ell$, $\tau \leq \beta\ell$, $n \geq \beta\ell + \mu$, $\epsilon = 2^{-c\ell}$ and $\delta = 2^{-(c-1)\ell+2}$ when $\ell = \omega(\log n)$. Fix any $c > 1$, and constants α, β corresponding to this c given by Lemma 4.1. For each $i \in [h]$, we instantiate $(n_i, \tau_i + \mu, d_i, \ell_i, \varepsilon_{exti})$ -extractor Ext^i that is $(\text{Full}_\mu, \delta_i)$ -adaptive as per this lemma as follows.

- We set $\ell_1 = \ell$, $\varepsilon_{ext1} = 2^{-c\ell}$, $\delta_1 = 2^{-\Omega(\ell)}$, $d_1 \leq \alpha\ell_1$, $\tau_1 \leq \beta\ell_1$ and $n_1 = \beta\ell_1 + \mu$.
- For $i > 1$, we set $\ell_i = \ell_{i-1} + d_{i-1}$, $\varepsilon_{exti} = 2^{-c\ell_i}$, $\delta_i = 2^{-\Omega(\ell_i)}$, $d_i \leq \alpha\ell_i$, $\tau_i \leq \beta\ell_i$ and $n_i = \beta\ell_i + \mu$.

With this setting, individual share length of Share^h is $\ell_h + d_h + \sum_{i \in [h]} n_i = h\mu + \Theta((1+\alpha)^h\ell)$. Therefore, when either $N = \Theta(t)$ or h is a constant, Share^h achieves a constant rate and a leakage of upto $\mu = \mathcal{O}(\ell)$ bits from each share.

As our instantiations of SdShare^i 's and MShare are perfectly adaptively private, we have Share^h to be a perfectly adaptively private secret sharing scheme which is $t \cdot 2^{-\Omega(\ell)}$ -leakage resilient against the adaptive leakage and reveal model.

4.4.3 Security Proof: Joint Leakage and Reveal Model

Having shown the security of our construction in Figure 4.5 in the adaptive leakage and reveal model, we now show how to extend the security to the joint leakage and reveal model, defined in Section 4.3.2.

Theorem 4.3 *For any $\psi, X > 0$ such that $\psi \cdot X \leq N - t + 1$ and $\ell, \mu > 0$, $(\text{Share}^h, \text{Rec}^h)$ is an (t, N, ε_p) -secret sharing scheme for ℓ bit messages and is ε_{lr} -leakage resilient in the joint leakage and reveal model $\mathcal{J}^{X, \psi, \mu}$ where $h = \lceil \frac{\psi}{\lfloor (t-1)/X \rfloor} \rceil$ and $\varepsilon_{lr} = 2(\varepsilon_{adp} + \sum_{i \in [h]} (4(t-1)X2^{X\ell_i}\varepsilon_{exti} + \varepsilon'_{adp_i}))$. Further, there exists an instantiation of the scheme, given in Section 4.4.3.1, with rate $(X^{\Theta(h)} + h\mu/\ell)^{-1}$. When X is a constant and when either $N = \Theta(t)$ or h is a constant, the scheme achieves constant rate and tolerates leakage up to $\mu = \Theta(\ell)$ bits per query, asymptotically.*

Proof: The proof of the theorem is along the same lines as the proof of Theorem 4.2, for the adaptive leakage and reveal model. Hence, we only describe only the hybrids in words, instead

of repeating the full hybrids again.

Correctness and privacy are already discussed in Theorem 4.2. We discuss leakage resilience against the joint leakage and reveal model here. The proof idea is similar in spirit to the leakage resilience proof of the adaptive leakage and reveal model. Let $a = \lfloor (t-1)/X \rfloor$ and $h = \lceil \frac{\psi}{a} \rceil$. We group the query sets into sets $\mathcal{JS}_1, \dots, \mathcal{JS}_h$ as follows. For $i \in [h-1]$, $\mathcal{JS}_i = Q_{(i-1)a+1} \cup \dots \cup Q_{ia}$. $\mathcal{JS}_h = Q_{(h-1)a+1} \cup \dots \cup Q_\psi$.

For any message m , we define the following sequence of hybrids (which are similar to hybrids in proof of Theorem 4.2). Without loss of generality, we assume that $\mathcal{D}$ always asks legitimate queries as per the model. The sequence of hybrids is $\mathbf{JLeakB}_0^m$, $\{\mathbf{JLeakA}_q^m, \mathbf{JLeakB}_q^m\}_{q \in [h]}$ and $\mathbf{JLeakC}^m$. Let $\mathbf{JLeakB}_0^m$ be the distribution $\mathbf{JLeak}_{\text{Share}^h}^m$.

$\mathbf{JLeakA}_q^m$: For each $1 \leq q \leq h$, the only change we make in $\mathbf{JLeakA}_q^m$ (in comparison to the previous hybrid $\mathbf{JLeakB}_{q-1}^m$) is that we replace the shares sd_j^q , for each $j \in \mathcal{JS}_q$ (the shares of s^q corresponding to the q -th set joint leakage queries), with shares of a dummy seed $\tilde{s}^q$. After answering the joint leakage queries corresponding to $\mathcal{S}_q$, $sd_{[N]}^q$ is re-sampled as shares of s^q conditioned on $\tilde{s}^q$. The statistical closeness of hybrids $\mathbf{JLeakB}_{q-1}^m$ and $\mathbf{JLeakA}_q^m$ follows from adaptive privacy of $\mathbf{SdShare}^q$. This is because $|\mathcal{JS}_q| \leq t-1$ and leakage responses to all queries in $\mathcal{JS}_{[q-1]}$ would be independent of s^q and its shares. The proof of this closeness follows the same steps as the proof of Claim 3.

$\mathbf{JLeakB}_q^m$: For each $1 < q \leq h$, the only change we make in $\mathbf{JLeakB}_q^m$ (in comparison to the previous hybrid $\mathbf{JLeakA}_q^m$) is that we replace the values x_j^q , for each $j \in \mathcal{JS}_q$ with random, instead of evaluating the $h - (q-1)$ layers of masking to get x_j^q (and hence the query response for any $Q_k \subseteq \mathcal{JS}_q$ is independent of m_{Q_k} and s^i for each $q \leq i \leq h$). Further, we continue to evaluate $x_j^{q-1}, x_j^{q-2}, \dots, x_j^1, y_j^1$, for each $j \in \mathcal{JS}_q$ as in the previous hybrid.

The response to each joint leakage query in $\mathbf{JLeakA}_q^m$ on any set $Q_k \subseteq \mathcal{JS}_q$ depends on $\{w_j^q, \text{Ext}^q(w_j^q; s^q)\}_{j \in Q_k}$ along with $\{w_j^i, s^i, sd_j^i\}_{i \in [h] \setminus \{q\}, j \in Q_k}$ and shares (corresponding to $j \in Q_k$) of a dummy seed $\tilde{s}^q$ (the distribution of the latter two random variables is identical in both hybrids). Now, consider a mega-extractor EXT^q which takes $w_{Q_k}^q, s^q$ as input and outputs $\{\text{Ext}^q(w_j^q; s^q)\}_{j \in Q_k}$ (as in Lemma 4.3 with respect to Ext^q and $u = X$). By Lemma 4.3, we know EXT^q is $(\text{Full}_\mu, 4X\varepsilon_{\text{ext}^q} \cdot 2^{X\ell_q})$ -adaptive²². Using adaptivity of this extractor with Lemma 4.2, we can show that leakage responses for all queries in $\mathcal{S}_q$ are statistically close by $(4(t-1)X\varepsilon_{\text{ext}^q} \cdot 2^{X\ell_q})$ in both hybrids. After answering queries corresponding to $\mathcal{S}_q$, all further joint leakage/reveal queries are answered in identical manner in both hybrids. The proof of this closeness follows the same steps as the proof of Claim 4.

²²While setting parameters we set $n_q - \tau_q \geq \mu$

JLeakC^m: In the hybrid JLeakB_h^m, all the shares used in the leakage phase are independent of the shares of the message m . Hence, the only part of the view of $\mathcal{D}$ that depends on the shares of m corresponds to the reveal phase. In the final hybrid JLeakC^m, we replace the $t - 1$ shares of m used in the reveal phase by shares of 0^ℓ . These hybrids are close by the adaptive privacy of MShare, which follows the same steps as the proof of Claim 5.

The formal proof of the statistical closeness of hybrids follows exactly like the corresponding claims in the proof of Theorem 4.2, giving the leakage error of $\varepsilon_{lr} = 2(\varepsilon_{adp} + \sum_{i \in [h]} (4(t - 1)X2^{X\ell_i}\varepsilon_{exti} + \varepsilon'_{adp_i}))$. $\square$

4.4.3.1 Instantiation and Rate Analysis

For $i \in [h]$, we instantiate SdShare^{*i*} on seeds of length d_i with the (adaptively) private Shamir secret sharing scheme (Section 2.6.1), which results in individual seed share length being d_i . We instantiate MShare on messages of length ℓ with the (adaptively) private Shamir secret sharing scheme, which results in individual seed share length being ℓ .

Further, we instantiate the extractor with Lemma 4.1, which states that for any $c > 1$, there exists constants α, β such that $d \leq \alpha\ell$, $\tau \leq \beta\ell$, $n \geq \beta\ell + \mu$, $\epsilon = 2^{-c\ell}$ and $\delta = 2^{-(c-1)\ell+2}$ when $\ell = \omega(\log n)$. Fix $c = 2X$, and α, β be values corresponding to this c given by Lemma 4.1. For each $i \in [h]$, we instantiate $(n_i, \tau_i, d_i, \ell_i, \varepsilon_{exti})$ -extractor Ext^{*i*} using the above lemma as follows²³.

- We set $\ell_1 = \ell$, $\varepsilon_{ext1} = 2^{-c\ell}$, $d_1 \leq \alpha\ell_1$, $\tau_1 \leq \beta\ell_1$ and $n_1 = \beta\ell_1 + \mu$.
- For $i > 1$, we set $\ell_i = \ell_{i-1} + d_{i-1}$, $\varepsilon_{exti} = 2^{-c\ell_i}$, $d_i \leq \alpha\ell_i$, $\tau_i \leq \beta\ell_i$ and $n_i = \beta\ell_i + \mu$.

With this setting, individual share length of Share^{*h*} is $\ell_h + d_h + \sum_{i \in [h]} n_i = h\mu + \Theta((1 + \alpha)^h \ell) = X^{\Theta(h)}\ell + h\mu$ (as α is specific to $c = 2X$). Therefore, when either $N = \Theta(t)$ or h is a constant, Share^{*h*} achieves

- Constant rate and a leakage of up to $\mu = \mathcal{O}(\ell)$ bits per query, whenever X is constant.
- Inverse poly logarithmic rate when $X = \log N$.

As our instantiations of SdShare^{*i*}'s and MShare are perfectly adaptively private, we have Share^{*h*} to be a perfectly adaptively private secret sharing scheme which is $\sum_{i \in [h]} 4(t-1)X2^{X\ell_i}\varepsilon_{exti} = t2^{-\Omega(\ell)}$ -leakage resilient against the adaptive leakage and reveal model.

²³Adaptivity of Ext^{*i*} isn't important and hence we don't mention parameters of adaptivity.

4.4.4 LRSS for General Access Structures

Our construction $(\text{Share}^h, \text{Rec}^h)$ can be easily adapted to provide security against general access structures as well. For $(\text{Share}^h, \text{Rec}^h)$ to be secure against a general access structure $\mathcal{A}$, the only modification is to instantiate MShare and SdShare^i in the construction to be secret sharing schemes that are secure (adaptively private) against $\mathcal{A}$ (we can instantiate this with the [19]-scheme given in 2.6.1).

Although, the leakage models we discussed previously in this section are formalized with respect to threshold access structures, the models naturally extend for general monotone access structures too easily. We would like to note that $(\text{Share}^h, \text{Rec}^h)$ (with the above modification) is also leakage resilient against these models for general access structures. We do not repeat the whole proof, as it would follow exactly the same steps with respect to general access structures.

4.5 Rate-optimal Non-adaptive LRSS Scheme

Since the main focus of our chapter was to bridge the gap between the strength of the leakage model and the rate, which we did above, we only give an overview of the construction that improves on the near-optimal rate non-adaptive independent (local leakage resilient) LRSS of [96] and gives an optimal rate of 1. As we only saw a modified version of [96] in Section 4.1.1, we begin by looking at the actual scheme of [96] for threshold access structures, before heading into our construction. This construction uses a threshold secret sharing scheme (with no leakage resilience guarantees) and a seeded randomness extractor.

Overview of [96]. Secret sharing of m follows these steps: a) First m is threshold secret shared into $m_1, \dots, m_N$ (referred to as “simple shares”) using a threshold secret sharing scheme b) Next, each m_i is masked using an extractor output $\text{Ext}(w_i, s)$ where s and w_i ’s are uniformly chosen. Now, let $sh_1, \dots, sh_N$ denote these masked shares c) r is uniformly chosen to additionally mask each sh_i d) Finally, r and s are together secret shared using a 2-out-of- N secret sharing scheme into shares $(a_1, \dots, a_N)$ and the i^{th} share of the scheme is then set to be $(w_i, sh_i \oplus r, a_i)$. At a high level, m_i was “doubly masked” in order to cast the leakage on the i^{th} share as leakage on the extractor source w_i . In order to add leakage resilience on top of the simple shares, they needed to be masked twice, and thus, information of both these masks and the masked value (each roughly of size $|m_i|$) is given as part of the final share, resulting in its length being approximately $3|m_i|$.

A Simpler Problem. Our goal is to compile simple shares m_i into leakage resilient shares in a share size-preserving manner (i.e., the size of the leakage resilient share needs to be about the same as $|m_i|$). As a first step, we relax the problem in two ways a) consider an LRSS only

for the (N, N) access structure (i.e., where the set of all N parties is the only authorized set); and b) require that the sharing scheme only works for uniformly random messages.

We first construct an LRSS scheme with optimal share size under these two relaxations. For this, we choose extractor sources $w_1, \dots, w_N$ and a seed s uniformly, secret share s as $(s_1, \dots, s_N)$ and define the i -th share, $share_i$ to be (w_i, s_i) . Now, define reconstruct $share_i$'s as $m = \oplus_{i \in [N]} \text{Ext}(w_i; s)$, where s is obtained by reconstructing s_i 's. First, observe that, by extractor security, the reconstructed value m has (almost) uniform distribution. Also, each share supports local leakage resilience as s_i (a share) is devoid of information about s and hence any bounded leakage of the form $f(w_i, s_i)$ is only dependent on w_i and is (almost) independent of the extractor output and hence m too. This scheme infact has an optimal share size of $|m| + \mu$ (for μ bits of leakage per share) as there are explicit extractor constructions with good parameters such that $|s| \ll |w_i| \approx |\text{Ext}(w_i, s)|$.

Final Construction Overview. Unfortunately, the above construction does not extend to either support threshold access structures or for secret sharing a specific message m . In order to reconstruct to a message m , the extractor outputs $\text{Ext}(w_i, s)$'s ($i \in [N]$) would have to be correlated. However, the fact that the extractor outputs $\text{Ext}(w_i, s)$ are uncorrelated is what gives leakage resilience in the scheme above for sharing random messages. The main technical hurdle which we overcome in this work is to ensure correlation in the shares while retaining enough independence (via extractors) so that we can argue leakage resilience.

In our construction, we first generate simple shares of m , denoted $(m_1, \dots, m_N)$ using a standard secret sharing scheme. Next, we aim to cast each of these simple shares m_i as an extractor output. This, however, has two challenges a) the distribution of m_i could be arbitrary and need not have any entropy; and b) it is not clear how to express m_i 's as the output of an extractor. To address (a), we observe that many natural secret sharing schemes (for example, the Shamir secret sharing scheme) satisfy the property that each share individually has (an almost) uniform distribution. We formalize this property as “local uniformity” of a secret sharing scheme (Section 2.6.3) and generate simple shares of m using such a locally uniform secret sharing scheme. To solve the challenge (b), we make use of seeded extractors that are linear functions - i.e., where the extractor function is guaranteed to be a linear map (over the source) for any fixed value of the seed, called linear seeded extractors [97, 98, 91]. We showed that such extractors provide an efficient way to find (entropic sample of) an extractor source such that the extractor output on this source takes a given value under a given seed (Lemma 2.12). With this useful property, each of our simple shares can indeed be expressed as extractor outputs.

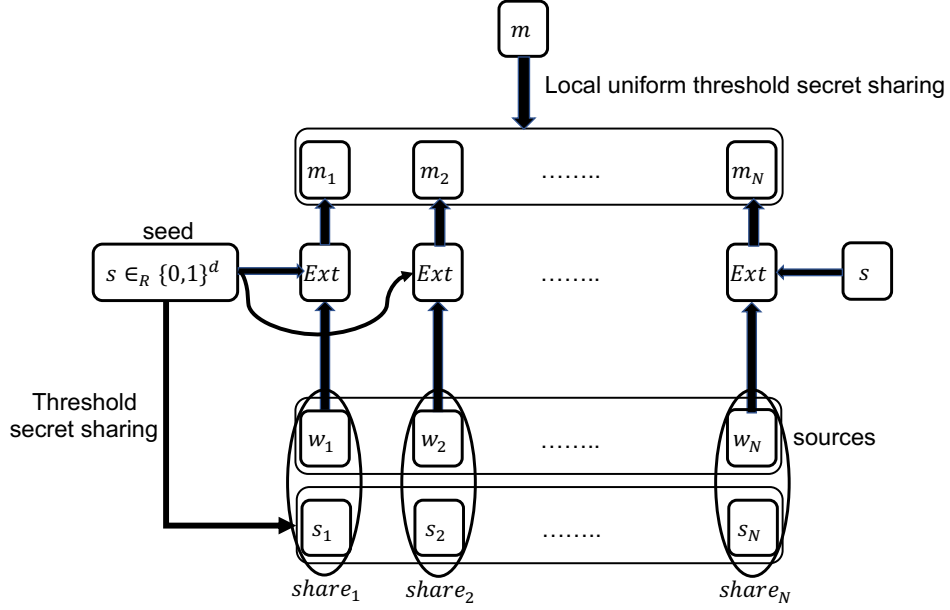


Figure 4.9: Construction of our Optimal Rate LRSS

To summarize our construction, described in Figure 4.9, we a) secret share m into simple shares $m_1, \dots, m_N$ using any locally uniform secret sharing scheme for the given general access structure; b) choose a seed s uniformly and generate its shares $s_1, \dots, s_N$ such that s can be reconstructed from any two shares s_i and s_j ; c) for each m_i , sample w_i such that $\text{Ext}(w_i, s) = m_i$; d) Finally, each share is set to be (w_i, s_i) for all $i \in [N]$.

Local leakage resilience of this scheme follows because: first, for each leakage query j , the source w_j can be simply picked at random, independent of the message, by using the linear extractor's invertibility property (Lemma 2.11) and the local uniformity (Section 2.6.3) of the underlying secret sharing schemes used on s and m ; second, the privacy of the sharing scheme corresponding to m can be used to show that the full shares given are also independent of the message m . In this scheme, the length of each share (w_i, s_i) is only negligibly larger than the length of m_i as there are explicit constructions of linear extractors that extract out almost all the entropy from the source while only using very short seeds (Lemma 2.11). Thus, we get a rate-1 LRSS scheme in the local leakage model.

Conclusion

We began with a quest to build optimal rate non-malleable codes and leakage resilient schemes, both of which are important primitives to handle two different kinds of side-channel attacks, namely, tampering attacks and leakage attacks, respectively. We summarize our results below:

1. In Chapter 3, we ventured into the study of NMCs and solved the long-standing open problem of rate-optimality by building a near-optimal rate-1/3 NMC in the 2-split-state model. In the process, we saw how relaxing the non-malleability requirement to just uniform random messages (which we called a non-malleable randomness encoder (NMRE)), helps us get a rate 1/2 NMRE in the 2-split-state model.
2. In Chapter 4, we looked at LRSS schemes, and bridged the existing gap in the literature between the strength of the leakage model and the rate of the scheme by building an LRSS scheme in the strong joint and adaptive leakage model, while still achieving a constant rate. Further, we also showed how one can get an optimal, rate-1 LRSS scheme in the non-adaptive and independent (local leakage) model.

Open Problems. While we resolved two major open problems in this thesis, we leave open some interesting problems. First, while a rate-1/3 scheme is the closest we have ever been to reaching the end goal, our quest ends only when we build the optimal, rate-1/2 NMCs in the 2-split-state model, which we leave open. We believe this would require some new techniques to achieve. Second, in our study of LRSS schemes, we introduced an interesting new primitive, called adaptive extractors, which is of independent interest. We leave open the study of adaptive extractors in other application-specific leakage models (e.g., polynomial depth circuits) and get non-trivial constructions of the same. It would be interesting to look at other applications of adaptive extractors in getting leakage resilient schemes. Third, while we did get a constant rate joint adaptive LRSS, we leave open the problem of building an optimal, constant rate LRSS in this strong model (as we did for the weaker local leakage model), potentially using some newer tools and techniques.

Bibliography

- [1] Divesh Aggarwal. Affine-evasive sets modulo a prime. *Inf. Process. Lett.*, 115(2):382–385, 2015. doi: 10.1016/j.ipl.2014.10.015. URL <https://doi.org/10.1016/j.ipl.2014.10.015>. 5, 20, 38
- [2] Divesh Aggarwal and Jop Briët. Revisiting the sanders-bogolyubov-ruzsa theorem in fp^n and its application to non-malleable codes. In *IEEE International Symposium on Information Theory, ISIT 2016, Barcelona, Spain, July 10-15, 2016*, pages 1322–1326. IEEE, 2016. doi: 10.1109/ISIT.2016.7541513. URL <https://doi.org/10.1109/ISIT.2016.7541513>. 5, 20
- [3] Divesh Aggarwal and Maciej Obremski. A constant rate non-malleable code in the split-state model. In *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 1285–1294. IEEE, 2020. doi: 10.1109/FOCS46700.2020.00122. URL <https://doi.org/10.1109/FOCS46700.2020.00122>. 3, 4, 5, 13, 29
- [4] Divesh Aggarwal, Yevgeniy Dodis, Zahra Jafargholi, Eric Miles, and Leonid Reyzin. Amplifying privacy in privacy amplification. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part II*, volume 8617 of *Lecture Notes in Computer Science*, pages 183–198. Springer, 2014. doi: 10.1007/978-3-662-44381-1_11. URL https://doi.org/10.1007/978-3-662-44381-1_11. 56
- [5] Divesh Aggarwal, Yevgeniy Dodis, and Shachar Lovett. Non-malleable codes from additive combinatorics. In *Symposium on Theory of Computing, STOC 2014*, 2014. doi: 10.1145/2591796.2591804. URL <http://doi.acm.org/10.1145/2591796.2591804>. 4, 5, 13, 20, 38
- [6] Divesh Aggarwal, Yevgeniy Dodis, Tomasz Kazana, and Maciej Obremski. Non-malleable

BIBLIOGRAPHY

- reductions and applications. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015*, 2015. doi: 10.1145/2746539.2746544. URL <http://doi.acm.org/10.1145/2746539.2746544>. 4, 5, 6, 19, 29
- [7] Divesh Aggarwal, Stefan Dziembowski, Tomasz Kazana, and Maciej Obremski. Leakage-resilient non-malleable codes. In *Theory of Cryptography Conference*, pages 398–426. Springer, 2015. 10, 14
- [8] Divesh Aggarwal, Tomasz Kazana, and Maciej Obremski. Inception makes non-malleable codes stronger. *IACR Cryptology ePrint Archive*, 2015:1013, 2015. URL <http://eprint.iacr.org/2015/1013>. 4, 5, 29
- [9] Divesh Aggarwal, Shashank Agrawal, Divya Gupta, Hemanta K. Maji, Omkant Pandey, and Manoj Prabhakaran. Optimal computational split-state non-malleable codes. In *Theory of Cryptography - 13th International Conference, TCC 2016-A, Tel Aviv, Israel, January 10-13, 2016, Proceedings, Part II*, pages 393–417, 2016. doi: 10.1007/978-3-662-49099-0_15. URL http://dx.doi.org/10.1007/978-3-662-49099-0_15. 4, 5, 19, 20, 29
- [10] Divesh Aggarwal, Ivan Damgård, Jesper Buus Nielsen, Maciej Obremski, Erick Purwanto, João Ribeiro, and Mark Simkin. Stronger leakage-resilient and non-malleable secret sharing schemes for general access structures. In *Advances in Cryptology - CRYPTO 2019*, 2019. doi: 10.1007/978-3-030-26951-7_18. URL https://doi.org/10.1007/978-3-030-26951-7_18. 3, 6, 9, 55
- [11] Dakshi Agrawal, Bruce Archambeault, Josyula R. Rao, and Pankaj Rohatgi. The EM side-channel(s). In B. Kaliski, C.K. Koc, and C. Paar, editors, *Cryptographic Hardware and Embedded Systems Conference (CHES '02)*, volume 2523 of *LNCS*. Springer-Verlag, 2002. 1
- [12] Shashank Agrawal, Divya Gupta, Hemanta K. Maji, Omkant Pandey, and Manoj Prabhakaran. A rate-optimizing compiler for non-malleable codes against bit-wise tampering and permutations. In *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part I*, pages 375–397, 2015. doi: 10.1007/978-3-662-46494-6_16. URL https://doi.org/10.1007/978-3-662-46494-6_16. 4, 5, 29
- [13] Adi Akavia, Shafi Goldwasser, and Vinod Vaikuntanathan. Simultaneous hardcore bits and cryptography against memory attacks. In Omer Reingold, editor, *Theory of Cryptography*,

BIBLIOGRAPHY

- 6th Theory of Cryptography Conference, TCC 2009, San Francisco, CA, USA, March 15-17, 2009. Proceedings*, volume 5444 of *Lecture Notes in Computer Science*, pages 474–495. Springer, 2009. doi: 10.1007/978-3-642-00457-5_28. URL https://doi.org/10.1007/978-3-642-00457-5_28. 1
- [14] Saikrishna Badrinarayanan and Akshayaram Srinivasan. Revisiting non-malleable secret sharing. In *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, 2019. doi: 10.1007/978-3-030-17653-2_20. URL https://doi.org/10.1007/978-3-030-17653-2_20. 3, 6, 55
- [15] Marshall Ball, Dana Dachman-Soled, Mukul Kulkarni, and Tal Malkin. Non-malleable codes for bounded depth, bounded fan-in circuits. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology - EUROCRYPT 2016*, pages 881–908, Berlin, Heidelberg, 2016. Springer Berlin Heidelberg. ISBN 978-3-662-49896-5. 5
- [16] Marshall Ball, Dana Dachman-Soled, Mukul Kulkarni, and Tal Malkin. Non-malleable codes from average-case hardness: Ac^0 , decision trees, and streaming space-bounded tampering. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2018*, pages 618–650, Cham, 2018. Springer International Publishing. 4, 5, 29
- [17] Mihir Bellare and Phillip Rogaway. Robust computational secret sharing and a unified account of classical secret-sharing goals. In *Proceedings of the 14th ACM Conference on Computer and Communications Security, CCS '07*. Association for Computing Machinery, 2007. ISBN 9781595937032. doi: 10.1145/1315245.1315268. URL <https://doi.org/10.1145/1315245.1315268>. 21
- [18] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In Janos Simon, editor, *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*, pages 1–10. ACM, 1988. doi: 10.1145/62212.62213. URL <https://doi.org/10.1145/62212.62213>. 2
- [19] Josh Cohen Benaloh and Jerry Leichter. Generalized secret sharing and monotone functions. In *Advances in Cryptology - CRYPTO '88, 8th Annual International Cryptology Conference*. Springer, 1988. doi: 10.1007/0-387-34799-2_3. URL https://doi.org/10.1007/0-387-34799-2_3. 21, 23, 84

BIBLIOGRAPHY

- [20] Fabrice Benhamouda, Akshay Degwekar, Yuval Ishai, and Tal Rabin. On the local leakage resilience of linear secret sharing schemes. In *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference*, e, 2018. doi: 10.1007/978-3-319-96884-1_18. URL https://doi.org/10.1007/978-3-319-96884-1_18. 6, 7, 55
- [21] Charles Bennett, Gilles Brassard, and Jean-Marc Robert. Privacy amplification by public discussion. *SIAM Journal on Computing*, 17(2):210–229, 1988. 1
- [22] Charles H. Bennett, Gilles Brassard, Claude Crépeau, and Ueli M. Maurer. Generalized privacy amplification. *IEEE Transactions on Information Theory*, 41(6):1915–1923, 1995. 1
- [23] Eli Biham and Adi Shamir. Differential fault analysis of secret key cryptosystems. In Burton S. Kaliski, Jr., editor, *Advances in Cryptology—CRYPTO '97*, volume 1294 of *LNCS*, pages 513–525. Springer-Verlag, 1997. 1
- [24] G.R. Blakley. Safeguarding cryptographic keys. In *Proceedings of the 1979 AFIPS National Computer Conference*. AFIPS Press, 1979. 2
- [25] D. Boneh, R. DeMillo, and R. Lipton. On the importance of checking cryptographic protocols for faults. In Walter Fumy, editor, *Advances in Cryptology—EUROCRYPT 97*, volume 1233 of *LNCS*, pages 37–51. Springer-Verlag, 1997. 1
- [26] Elette Boyle, Shafi Goldwasser, and Yael Tauman Kalai. Leakage-resilient coin tossing. In David Peleg, editor, *Distributed Computing - 25th International Symposium, DISC 2011, Rome, Italy, September 20-22, 2011. Proceedings*, volume 6950 of *Lecture Notes in Computer Science*, pages 181–196. Springer, 2011. doi: 10.1007/978-3-642-24100-0_16. URL https://doi.org/10.1007/978-3-642-24100-0_16. 6
- [27] Gianluca Brian, Antonio Faonio, and Daniele Venturi. Continuously non-malleable secret sharing for general access structures. In *Theory of Cryptography - 17th International Conference, TCC 2019*, 2019. doi: 10.1007/978-3-030-36033-7_8. URL https://doi.org/10.1007/978-3-030-36033-7_8. 6, 9, 55
- [28] Gianluca Brian, Antonio Faonio, Maciej Obremski, Mark Simkin, and Daniele Venturi. Non-malleable secret sharing against bounded joint-tampering attacks in the plain model. In *Advances in Cryptology - CRYPTO 2020*, 2020. doi: 10.1007/978-3-030-56877-1_5. URL https://doi.org/10.1007/978-3-030-56877-1_5. 6, 8, 9, 55

BIBLIOGRAPHY

- [29] Gianluca Brian, Antonio Faonio, Maciej Obremski, João L. Ribeiro, Mark Simkin, Maciej Skórski, and Daniele Venturi. The mother of all leakages: How to simulate noisy leakages via bounded leakage (almost) for free. In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology - EUROCRYPT 2021 - 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17-21, 2021, Proceedings, Part II*, volume 12697 of *Lecture Notes in Computer Science*, pages 408–437. Springer, 2021. doi: 10.1007/978-3-030-77886-6_14. URL https://doi.org/10.1007/978-3-030-77886-6_14. 9
- [30] Gianluca Brian, Antonio Faonio, and Daniele Venturi. Continuously non-malleable secret sharing: Joint tampering, plain model and capacity. In Kobbi Nissim and Brent Waters, editors, *Theory of Cryptography - 19th International Conference, TCC 2021, Raleigh, NC, USA, November 8-11, 2021, Proceedings, Part II*, volume 13043 of *Lecture Notes in Computer Science*, pages 333–364. Springer, 2021. doi: 10.1007/978-3-030-90453-1_12. URL https://doi.org/10.1007/978-3-030-90453-1_12. 8
- [31] Ran Canetti, Yevgeniy Dodis, Shai Halevi, Eyal Kushilevitz, and Amit Sahai. Exposure-resilient functions and all-or-nothing transforms. In Bart Preneel, editor, *Advances in Cryptology—EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 453–469. Springer-Verlag, 2000. ISBN 3-540-67517-5. 1
- [32] Nishanth Chandran, Vipul Goyal, Pratyay Mukherjee, Omkant Pandey, and Jalaj Upadhyay. Block-wise non-malleable codes. In *ICALP*, volume 55 of *LIPIcs*, pages 31:1–31:14. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016. 4, 5, 29
- [33] Nishanth Chandran, Bhavana Kanukurthi, and Srinivasan Raghuraman. Information-theoretic local non-malleable codes and their applications. In *Theory of Cryptography - 13th International Conference, TCC 2016-A, Tel Aviv, Israel, January 10-13, 2016, Proceedings, Part II*, pages 367–392, 2016. doi: 10.1007/978-3-662-49099-0_14. URL http://dx.doi.org/10.1007/978-3-662-49099-0_14. 4, 29
- [34] Eshan Chattopadhyay and Xin Li. Non-malleable codes and extractors for small-depth circuits, and affine functions. In *STOC*, pages 1171–1184. ACM, 2017. 4, 5, 29
- [35] Eshan Chattopadhyay and David Zuckerman. Non-malleable codes against constant split-state tampering. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 306–315, 2014. doi: 10.1109/FOCS.2014.40. URL <http://dx.doi.org/10.1109/FOCS.2014.40>. 4, 29

BIBLIOGRAPHY

- [36] Eshan Chattopadhyay, Vipul Goyal, and Xin Li. Non-malleable extractors and codes, with their many tampered extensions. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 285–298, 2016. doi: 10.1145/2897518.2897547. URL <http://doi.acm.org/10.1145/2897518.2897547>. 4, 5, 29
- [37] Eshan Chattopadhyay, Bhavana Kanukurthi, Sai Lakshmi Bhavana Obbattu, and Sruthi Sekar. Privacy amplification from non-malleable codes. In Feng Hao, Sushmita Ruj, and Sourav Sen Gupta, editors, *Progress in Cryptology - INDOCRYPT 2019 - 20th International Conference on Cryptology in India, Hyderabad, India, December 15-18, 2019, Proceedings*, volume 11898 of *Lecture Notes in Computer Science*, pages 318–337. Springer, 2019. doi: 10.1007/978-3-030-35423-7_16. URL https://doi.org/10.1007/978-3-030-35423-7_16. 3
- [38] Eshan Chattopadhyay, Jesse Goodman, Vipul Goyal, Ashutosh Kumar, Xin Li, Raghu Meka, and David Zuckerman. Extractors and secret sharing against bounded collusion protocols. In *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020*, 2020. doi: 10.1109/FOCS46700.2020.00117. URL <https://doi.org/10.1109/FOCS46700.2020.00117>. 6, 8, 9, 55
- [39] David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols (extended abstract). In Janos Simon, editor, *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*, pages 11–19. ACM, 1988. doi: 10.1145/62212.62214. URL <https://doi.org/10.1145/62212.62214>. 2
- [40] Mahdi Cheraghchi and Venkatesan Guruswami. Non-malleable coding against bit-wise and split-state tampering. In *Theory of Cryptography - 11th Theory of Cryptography Conference, TCC 2014, San Diego, CA, USA, February 24-26, 2014. Proceedings*, pages 440–464, 2014. doi: 10.1007/978-3-642-54242-8_19. URL http://dx.doi.org/10.1007/978-3-642-54242-8_19. 4, 15, 18, 29, 50, 51
- [41] Mahdi Cheraghchi and Venkatesan Guruswami. Capacity of non-malleable codes. In *Innovations in Theoretical Computer Science, ITCS'14, Princeton, NJ, USA, January 12-14, 2014*, pages 155–168, 2014. doi: 10.1145/2554797.2554814. URL <http://doi.acm.org/10.1145/2554797.2554814>. 3, 4, 5, 29

BIBLIOGRAPHY

- [42] Mahdi Cheraghchi, Frédéric Didier, and Amin Shokrollahi. Invertible extractors and wire-tap protocols. *CoRR*, abs/0901.2120, 2009. URL <http://arxiv.org/abs/0901.2120>. 16
- [43] Sandro Coretti, Ueli Maurer, Björn Tackmann, and Daniele Venturi. From single-bit to multi-bit public-key encryption via non-malleable codes. *IACR Cryptology ePrint Archive*, 2014:324, 2014. URL <http://eprint.iacr.org/2014/324>. 3
- [44] Sandro Coretti, Yevgeniy Dodis, Björn Tackmann, and Daniele Venturi. Non-malleable encryption: Simpler, shorter, stronger. In Eyal Kushilevitz and Tal Malkin, editors, *Theory of Cryptography*, pages 306–335, Berlin, Heidelberg, 2016. Springer Berlin Heidelberg. ISBN 978-3-662-49096-9. 3
- [45] Sandro Coretti, Antonio Faonio, and Daniele Venturi. Rate-optimizing compilers for continuously non-malleable codes. In Robert H. Deng, Valérie Gauthier-Umaña, Martín Ochoa, and Moti Yung, editors, *Applied Cryptography and Network Security - 17th International Conference, ACNS 2019, Bogota, Colombia, June 5-7, 2019, Proceedings*, volume 11464 of *Lecture Notes in Computer Science*, pages 3–23. Springer, 2019. doi: 10.1007/978-3-030-21568-2_1. URL https://doi.org/10.1007/978-3-030-21568-2_1. 5
- [46] Dana Dachman-Soled, Feng-Hao Liu, Elaine Shi, and Hong-Sheng Zhou. Locally decodable and updatable non-malleable codes and their applications. *IACR Cryptology ePrint Archive*, 2014:663, 2014. URL <http://eprint.iacr.org/2014/663>. 4, 29
- [47] Dana Dachman-Soled, Mukul Kulkarni, and Aria Shahverdi. Tight upper and lower bounds for leakage-resilient, locally decodable and updatable non-malleable codes. *IACR Cryptology ePrint Archive*, 2017:15, 2017. URL <http://eprint.iacr.org/2017/015>. 4, 29
- [48] Dana Dachman-Soled, Ilan Komargodski, and Rafael Pass. Non-malleable codes for bounded parallel-time tampering. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16-20, 2021, Proceedings, Part III*, volume 12827 of *Lecture Notes in Computer Science*, pages 535–565. Springer, 2021. doi: 10.1007/978-3-030-84252-9_18. URL https://doi.org/10.1007/978-3-030-84252-9_18. 5
- [49] Francesco Davì, Stefan Dziembowski, and Daniele Venturi. Leakage-resilient storage. In *Security and Cryptography for Networks, 7th International Conference, SCN 2010*, 2010. doi:

BIBLIOGRAPHY

- 10.1007/978-3-642-15317-4_9. URL https://doi.org/10.1007/978-3-642-15317-4_9. 6, 55
- [50] Yvo Desmedt and Yair Frankel. Threshold cryptosystems. In G. Brassard, editor, *Advances in Cryptology—CRYPTO ’89*, volume 435 of *LNCS*, pages 307–315. Springer-Verlag, 1990, 20–24 August 1989. 2
- [51] Yevgeniy Dodis, Amit Sahai, and Adam D. Smith. On perfect and adaptive security in exposure-resilient cryptography. In Birgit Pfitzmann, editor, *Advances in Cryptology - EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6-10, 2001, Proceeding*, volume 2045 of *Lecture Notes in Computer Science*, pages 301–324. Springer, 2001. doi: 10.1007/3-540-44987-6_19. URL https://doi.org/10.1007/3-540-44987-6_19. 1
- [52] Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM Journal on Computing*, 38(1):97–139, 2008. arXiv:cs/0602007. 12
- [53] Yevgeniy Dodis, Bhavana Kanukurthi, Jonathan Katz, Leonid Reyzin, and Adam Smith. Robust fuzzy extractors and authenticated key agreement from close secrets. *IEEE Transactions on Information Theory*, 2012. 18
- [54] Nico Döttling, Jesper Buus Nielsen, and Maciej Obremski. Information theoretic continuously non-malleable codes in the constant split-state model. *Electronic Colloquium on Computational Complexity (ECCC)*, 24:78, 2017. URL <https://eccc.weizmann.ac.il/report/2017/078>. 4, 5, 29
- [55] Stefan Dziembowski and Krzysztof Pietrzak. Intrusion-resilient secret sharing. In *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science, FOCS ’07*, 2007. ISBN 0-7695-3010-9. doi: 10.1109/FOCS.2007.35. URL <http://dx.doi.org/10.1109/FOCS.2007.35>. 1, 2, 14, 27, 55
- [56] Stefan Dziembowski and Krzysztof Pietrzak. Leakage-resilient cryptography. In *49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008, October 25-28, 2008, Philadelphia, PA, USA*, pages 293–302. IEEE Computer Society, 2008. doi: 10.1109/FOCS.2008.56. URL <https://doi.org/10.1109/FOCS.2008.56>. 8
- [57] Stefan Dziembowski, Krzysztof Pietrzak, and Daniel Wichs. Non-malleable codes. In *Innovations in Computer Science - ICS 2010, Tsinghua University, Beijing, China, January*

BIBLIOGRAPHY

- 5-7, 2010. *Proceedings*, pages 434–452, 2010. URL <http://conference.itcs.tsinghua.edu.cn/ICS2010/content/papers/34.html>. 1, 3, 4, 5, 18, 29
- [58] Stefan Dziembowski, Tomasz Kazana, and Maciej Obremski. Non-malleable codes from two-source extractors. In *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II*, pages 239–257, 2013. doi: 10.1007/978-3-642-40084-1_14. URL http://dx.doi.org/10.1007/978-3-642-40084-1_14. 4, 5, 9
- [59] Antonio Faonio and Daniele Venturi. Non-malleable secret sharing in the computational setting: Adaptive tampering, noisy-leakage resilience, and improved rate. In *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference*, 2019. doi: 10.1007/978-3-030-26951-7_16. URL https://doi.org/10.1007/978-3-030-26951-7_16. 6, 8, 9, 55
- [60] Sebastian Faust, Tal Rabin, Leonid Reyzin, Eran Tromer, and Vinod Vaikuntanathan. Protecting circuits from leakage: the computationally-bounded and noisy cases. In *Advances in Cryptology - EUROCRYPT 2010*, 2010. ISBN 978-3-642-13190-5. 2
- [61] Sebastian Faust, Pratyay Mukherjee, Jesper Buus Nielsen, and Daniele Venturi. Continuous non-malleable codes. In *Theory of Cryptography - 11th Theory of Cryptography Conference, TCC 2014, San Diego, CA, USA, February 24-26, 2014. Proceedings*, pages 465–488, 2014. doi: 10.1007/978-3-642-54242-8_20. URL http://dx.doi.org/10.1007/978-3-642-54242-8_20. 4, 5, 29
- [62] Sebastian Faust, Kristina Hostáková, Pratyay Mukherjee, and Daniele Venturi. Non-malleable codes for space-bounded tampering. In *CRYPTO (2)*, volume 10402 of *Lecture Notes in Computer Science*, pages 95–126. Springer, 2017. 4, 5, 29
- [63] Yair Frankel. A practical protocol for large group oriented networks. In Jean-Jacques Quisquater and Joos Vandewalle, editors, *Advances in Cryptology - EUROCRYPT '89, Workshop on the Theory and Application of Cryptographic Techniques, Houthalen, Belgium, April 10-13, 1989, Proceedings*, volume 434 of *Lecture Notes in Computer Science*, pages 56–61. Springer, 1989. doi: 10.1007/3-540-46885-4_8. URL https://doi.org/10.1007/3-540-46885-4_8. 2
- [64] Rosario Gennaro, Anna Lysyanskaya, Tal Malkin, Silvio Micali, and Tal Rabin. Tamper-proof security: Theoretical foundations for security against hardware tampering. In Naor [86]. 1

BIBLIOGRAPHY

- [65] Vipul Goyal and Ashutosh Kumar. Non-malleable secret sharing. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018*, 2018. doi: 10.1145/3188745.3188872. URL <https://doi.org/10.1145/3188745.3188872>. 3, 6, 9, 10, 55
- [66] Vipul Goyal, Yuval Ishai, Hemanta K. Maji, Amit Sahai, and Alexander A. Sherstov. Bounded-communication leakage resilience via parity-resilient circuits. In Irit Dinur, editor, *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 1–10. IEEE Computer Society, 2016. doi: 10.1109/FOCS.2016.10. URL <https://doi.org/10.1109/FOCS.2016.10>. 6
- [67] Vipul Goyal, Omkant Pandey, and Silas Richelson. Textbook non-malleable commitments. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 1128–1141, 2016. doi: 10.1145/2897518.2897657. URL <http://doi.acm.org/10.1145/2897518.2897657>. 2, 3
- [68] Divya Gupta, Hemanta K. Maji, and Mingyuan Wang. Non-malleable codes against look-ahead tampering. In Debrup Chakraborty and Tetsu Iwata, editors, *Progress in Cryptology - INDOCRYPT 2018 - 19th International Conference on Cryptology in India, New Delhi, India, December 9-12, 2018, Proceedings*, volume 11356 of *Lecture Notes in Computer Science*, pages 307–328. Springer, 2018. doi: 10.1007/978-3-030-05378-9_17. URL https://doi.org/10.1007/978-3-030-05378-9_17. 4, 5, 29
- [69] Venkatesan Guruswami and Mary Wootters. Repairing reed-solomon codes. In *Proceedings of the Forty-eighth Annual ACM Symposium on Theory of Computing, STOC '16*, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4132-5. doi: 10.1145/2897518.2897525. URL <http://doi.acm.org/10.1145/2897518.2897525>. 2
- [70] Venkatesan Guruswami, Christopher Umans, and Salil P. Vadhan. Unbalanced expanders and randomness extractors from Parvaresh-Vardy codes. In *IEEE Conference on Computational Complexity*, pages 96–108, 2007. 15, 63
- [71] Carmit Hazay, Muthuramakrishnan Venkitasubramaniam, and Mor Weiss. Zk-pcps from leakage-resilient secret sharing. *2nd Conference on Information-Theoretic Cryptography, ITC 2021*. URL <https://eprint.iacr.org/2021/606>. 6

BIBLIOGRAPHY

- [72] Yuval Ishai, Amit Sahai, and David Wagner. Private circuits: Securing hardware against probing attacks. In Dan Boneh, editor, *Advances in Cryptology—CRYPTO 2003*, volume 2729 of *LNCS*. Springer-Verlag, 2003. 2
- [73] Yuval Ishai, Manoj Prabhakaran, Amit Sahai, and David A. Wagner. Private circuits II: keeping secrets in tamperable circuits. In Serge Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings*, volume 4004 of *Lecture Notes in Computer Science*, pages 308–327. Springer, 2006. doi: 10.1007/11761679\19. URL https://doi.org/10.1007/11761679_19. 1
- [74] Zahra Jafargholi and Daniel Wichs. Tamper detection and continuous non-malleable codes. In *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part I*, pages 451–480, 2015. doi: 10.1007/978-3-662-46494-6_19. URL http://dx.doi.org/10.1007/978-3-662-46494-6_19. 4, 5, 29
- [75] Thomas Johansson, Gregory Kabatianskii, and Ben J. M. Smeets. On the relation between a-codes and codes correcting independent errors. In *Advances in Cryptology - EUROCRYPT '93, Workshop on the Theory and Application of Cryptographic Techniques, Lofthus, Norway, May 23-27, 1993, Proceedings*, pages 1–11, 1993. doi: 10.1007/3-540-48285-7_1. URL https://doi.org/10.1007/3-540-48285-7_1. 18
- [76] Bhavana Kanukurthi, Sai Lakshmi Bhavana Obbattu, and Sruthi Sekar. Four-state non-malleable codes with explicit constant rate. In *Theory of Cryptography - 15th International Conference, TCC 2017, Baltimore, MD, USA, November 12-15, 2017, Proceedings, Part II*, pages 344–375, 2017. doi: 10.1007/978-3-319-70503-3_11. URL https://doi.org/10.1007/978-3-319-70503-3_11. 4, 5, 13, 29, 32
- [77] Bhavana Kanukurthi, Sai Lakshmi Bhavana Obbattu, and Sruthi Sekar. Non-malleable randomness encoders and their applications. In *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part III*, pages 589–617, 2018. doi: 10.1007/978-3-319-78372-7_19. URL https://doi.org/10.1007/978-3-319-78372-7_19. 4, 5
- [78] Paul Kocher. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other

BIBLIOGRAPHY

- systems. In Neal Koblitz, editor, *Advances in Cryptology—CRYPTO ’96*, volume 1109 of *LNCS*. Springer-Verlag, 18–22 August 1996. [1](#), [2](#)
- [79] Ashutosh Kumar, Raghu Meka, and Amit Sahai. Leakage-resilient secret sharing against colluding parties. In *60th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2019*, 2019. doi: 10.1109/FOCS.2019.00045. URL <https://doi.org/10.1109/FOCS.2019.00045>. [6](#), [8](#), [9](#), [55](#)
- [80] Xin Li. Improved non-malleable extractors, non-malleable codes and independent source extractors. In *Symposium on Theory of Computing, STOC 2017, Montreal, Canada, June 19-23, 2017*, 2017. [4](#), [5](#), [15](#), [29](#)
- [81] Xin Li. Non-malleable extractors and non-malleable codes: Partially optimal constructions. In *Computational Complexity Conference, CCC 2019, New Brunswick, June 18-20, 2019*, 2019. [4](#), [5](#), [20](#), [29](#), [50](#), [53](#)
- [82] Fuchun Lin, Mahdi Cheraghchi, Venkatesan Guruswami, Reihaneh Safavi-Naini, and Huaxiong Wang. Non-malleable secret sharing against affine tampering. *CoRR*, abs/1902.06195, 2019. URL <http://arxiv.org/abs/1902.06195>. [6](#), [15](#), [16](#), [55](#)
- [83] Feng-Hao Liu and Anna Lysyanskaya. Algorithmic tamper-proof security under probing attacks. In Juan A. Garay and Roberto De Prisco, editors, *Security and Cryptography for Networks, 7th International Conference, SCN 2010, Amalfi, Italy, September 13-15, 2010. Proceedings*, volume 6280 of *Lecture Notes in Computer Science*, pages 106–120. Springer, 2010. doi: 10.1007/978-3-642-15317-4_8. URL https://doi.org/10.1007/978-3-642-15317-4_8. [1](#)
- [84] Feng-Hao Liu and Anna Lysyanskaya. Tamper and leakage resilience in the split-state model. In *Advances in Cryptology - CRYPTO 2012 - 32nd Annual Cryptology Conference*, 2012. doi: 10.1007/978-3-642-32009-5_30. URL https://doi.org/10.1007/978-3-642-32009-5_30. [1](#), [3](#), [6](#), [10](#), [55](#)
- [85] Silvio Micali and Leonid Reyzin. Physically observable cryptography (extended abstract). In Naor [\[86\]](#), pages 278–296. [1](#)
- [86] Moni Naor, editor. *First Theory of Cryptography Conference — TCC 2004*, volume 2951 of *LNCS*, February 19–21 2004. Springer-Verlag. [96](#), [99](#)

BIBLIOGRAPHY

- [87] Moni Naor and Gil Segev. Public-key cryptosystems resilient to key leakage. In Shai Halevi, editor, *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings*, volume 5677 of *Lecture Notes in Computer Science*, pages 18–35. Springer, 2009. doi: 10.1007/978-3-642-03356-8_2. URL https://doi.org/10.1007/978-3-642-03356-8_2. 9
- [88] Moni Naor and Gil Segev. Public-key cryptosystems resilient to key leakage. *SIAM J. Comput.*, 41(4):772–814, 2012. doi: 10.1137/100813464. URL <https://doi.org/10.1137/100813464>. 9
- [89] Jesper Buus Nielsen and Mark Simkin. Lower bounds for leakage-resilient secret sharing. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceedings, Part I*, volume 12105 of *Lecture Notes in Computer Science*, pages 556–577. Springer, 2020. doi: 10.1007/978-3-030-45721-1_20. URL https://doi.org/10.1007/978-3-030-45721-1_20. 7
- [90] Noam Nisan and David Zuckerman. Randomness is linear in space. *Journal of Computer and System Sciences*, 52(1):43–53, 1996. 14
- [91] Ran Raz, Omer Reingold, and Salil P. Vadhan. Extracting all the randomness and reducing the error in trevisan’s extractors. *J. Comput. Syst. Sci.*, 65(1):97–128, 2002. doi: 10.1006/jcss.2002.1824. URL <https://doi.org/10.1006/jcss.2002.1824>. 15, 85
- [92] Ronald L. Rivest. All-or-nothing encryption and the package transform. In Eli Biham, editor, *Fast Software Encryption, 4th International Workshop, FSE ’97, Haifa, Israel, January 20-22, 1997, Proceedings*, volume 1267 of *Lecture Notes in Computer Science*, pages 210–218. Springer, 1997. doi: 10.1007/BFb0052348. URL <https://doi.org/10.1007/BFb0052348>. 1
- [93] Guy N. Rothblum. How to compute under ac0 leakage without secure hardware. In *Proceedings of the 32Nd Annual Cryptology Conference on Advances in Cryptology — CRYPTO 2012*, 2012. doi: 10.1007/978-3-642-32009-5_32. URL https://doi.org/10.1007/978-3-642-32009-5_32. 2
- [94] Alfredo De Santis, Yvo Desmedt, Yair Frankel, and Moti Yung. How to share a function securely. In Frank Thomson Leighton and Michael T. Goodrich, editors, *Proceedings of*

BIBLIOGRAPHY

- the Twenty-Sixth Annual ACM Symposium on Theory of Computing, 23-25 May 1994, Montréal, Québec, Canada*, pages 522–533. ACM, 1994. doi: 10.1145/195058.195405. URL <https://doi.org/10.1145/195058.195405>. 2
- [95] Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979. 2, 6, 7, 21
- [96] Akshayaram Srinivasan and Prashant Nalini Vasudevan. Leakage resilient secret sharing and applications. In *Advances in Cryptology – CRYPTO 2019*. Springer International Publishing, 2019. ISBN 978-3-030-26951-7. 2, 3, 6, 7, 9, 55, 57, 84
- [97] Luca Trevisan. Construction of extractors using pseudo-random generators (extended abstract). In *STOC*, pages 141–148, 1999. 15, 85
- [98] Luca Trevisan. Extractors and pseudorandom generators. *J. ACM*, 48(4):860–879, 2001. doi: 10.1145/502090.502099. URL <https://doi.org/10.1145/502090.502099>. 85
- [99] Marius Zimand. Exposure-resilient extractors. In *21st Annual IEEE Conference on Computational Complexity (CCC 2006)*. IEEE Computer Society, 2006. doi: 10.1109/CCC.2006.19. URL <https://doi.org/10.1109/CCC.2006.19>. 56